Université de Mons
Faculté des Sciences

Neural Marked Temporal Point Processes for Probabilistic Predictive Modeling of Continuous-Time Event Data

Tanguy Bosser

A dissertation submitted in fulfillment of the requirements of the degree of
Docteur en Sciences

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Pr. Souhaib Ben Taieb, for his unwavering and invaluable support throughout this entire journey. Your insightful advice and our fruitful discussion sessions have been instrumental in my growth as both a researcher and a teacher. Under your guidance, I have learned so much, and your constant encouragement has helped shape me into a more capable and confident individual.

I am especially grateful for your belief in me, even though we had a challenging start. You gave me the opportunity to start on this path despite my very limited background in computer science, and for that, I am forever thankful. Looking back, I can only realize how much progress I have made during these past four years, and I owe a great deal of that to your mentorship.

Thank you for always taking the time to provide me critical feedback on my work, and for these last-minute revision sprints right before a deadline. You always pushed me to strive for my best self, and your timely words of encouragement brought me back on track during difficult moments—especially when I felt at my lowest. Today, I am deeply grateful that you always encouraged me to keep pushing ahead.

I would also like to extend my gratitude to my co-supervisor, Pr. Tom Mens. Even though we interacted to a lesser extent research-wise during this journey, you always found time to provide me with feedback and guidance on navigating the PhD process, particularly during the early stages of my thesis. Also, I would like to thank the members of my thesis committee, Pr. Stephane Dupont and Pr. Marco Saerens, for your support and insightful suggestions on how to further conduct my studies. I would also like to express my gratitude to Pr. James Taylor for accepting to be a member of my jury.

Additionally, I would like to thank the members of the BDML lab at the University of Mons for the great times we spent during group lunches and summer schools, as well as for your constant availability to help with various matters. Especially, I would like to thank Yorick for accepting to proofread an early version of this thesis, and Victor for taking the time to answer my questions on calibration.

I would also like to express my deepest gratitude to my family for always believing in me and for recognizing the value of my work even when I could not see it myself. Thank you for your constant words of reassurance and encouragement, and for your understanding when I could not make me available for family reunions and trips. I owe you everything, and I know that you are proud of me.

I would also like to thank all my friends who, one way or another, contributed significantly to the completion of this journey. Thank you for your encouragements and for all these fun trips and moments that we shared together, allowing me to take my mind off stress during the most difficult times. Your presence made this whole journey much more bearable and enjoyable. To your disappointment though, I didn't find *mon chiffre*, but I completed *mon gros devoir*, as you like to call it, the best that I could.

Last but not least, I'd like to extend my heartfelt thanks to you, Sophie, for your unwavering support and for all your little attentions that contributed to make my life easier during these last difficult months. You always found a way to revive my motivation and help me realize that I might be a little too hard on myself. Thank you for being such a joy and for making me laugh each passing day, I feel so lucky to have you in my life.

Abstract

Labeled event sequences occurring at irregular intervals in continuous time are common across a wide range of application domains. Typical examples include user interactions on social media platforms, online shopping activity, electronic health records, and earthquake occurrences in seismology. Marked Temporal Point Processes (MTPP) provide a principled framework for modeling these event sequences, enabling subsequent inferences such as predicting the arrival times of future events, as well as their associated labels, called marks. In this context, the main challenge consists in learning a MTPP model that accurately captures the cross-temporal dependencies between past observations and future ones. However, classical MTPP models are often constrained by strong assumptions, which practically limit their ability to capture the complex dynamics of real-world patterns. To overcome this limitation, neural MTPP models have emerged as a flexible alternative, leveraging neural network parametrizations to improve modeling capabilities. Since its introduction, the field of neural MTPP modeling witnessed rapid development, with the emergence of numerous neural architectures applied successfully to a diverse set of real-world problems.

While recent studies demonstrate the effectiveness of neural MTPP models, the field still faces a series of open challenges that warrant future exploration from the community. Specifically, evaluation setups for newly proposed models often lack consistency between studies, relying on different baselines, datasets, and experimental configurations. This context makes it challenging to identify key factors driving improvements in predictive accuracy, effectively hindering future research progress in the field. Another challenge relates to the common strategies adopted to learn these probabilistic models from event sequence data.

In practice, neural MTPP models are often trained by Maximum Likelihood Estimation (MLE), which reduces to minimizing the Negative Log-Likelihood (NLL) over observed sequences. This training procedure implicitly involves learning two predictive distributions: one for the arrival times of events, and another for their marks. Usually, neural MTPP parametrizations enforce these two distributions to be learned jointly, which can lead to challenges during optimization and degrade model performance. Finally, a third challenge relates to accurately quantifying the uncertainty in the predictions extracted from neural MTPP models. Indeed, due to model misspecification or lack of training data, these probabilistic models may provide a poor approximation of the true, unknown underlying process. Consequently, prediction regions extracted directly from the model may be unreliable, failing to faithfully reflect the true uncertainty.

In this thesis, we seek to address these concerns by introducing novel neural MTPP models for probabilistic modeling of continuous-time event data, designing new training strategies to enhance their predictive accuracy, and developing reliable, distribution-free methods to quantify the uncertainty in their predictions. To that end, we begin our discussion with a comprehensive large-scale experimental study that systematically evaluates the predictive accuracy of modern neural MTPP models. We thoroughly investigate the influence of major architectural components in modeling the time and mark predictive distributions, and shed light on specific design choices that lead to increased predictive accuracy. Furthermore, we delve into the less explored area of probabilistic calibration for neural MTPP models, and highlight that the mark predictive distribution is often miscalibrated. Our study aims to provide valuable insights into the performance and characteristics of neural MTPP models, contributing to a better understanding of their strengths and limitations.

We then show that learning a neural MTPP model with the NLL objective can be interpreted as a two-task learning problem, where both tasks share a common set of trainable parameters that are optimized jointly. We show that this common practice can lead to the emergence of conflicting gradients during training, where task-specific gradients are pointing in opposite directions. When such conflicts arise, following the average gradient can be detrimental to the learning of each individual task, resulting in overall degraded performance. To overcome this issue, we introduce novel parametrizations for neural MTPP models that allow for separate modeling and training of each task, effectively preventing the emergence of conflicts. Specifically, our framework allows to prevent conflicting gradients from the root while maintaining the flexibility of the original parametrizations. Our experiments on real-world event sequence datasets outline the advantages of our framework over the original model formulations.

Finally, we develop more reliable methods for uncertainty quantification in neural MTPP models via the framework of conformal prediction. A primary objective is to generate a distribution-free joint prediction region for an event's arrival time and mark, with finite-sample coverage guarantees. We first consider a simple but overly conservative approach that combines individual prediction regions for the event's arrival time and mark. Then, we introduce a more effective method based on bivariate highest density regions derived from the joint predictive density of arrival times and marks. By leveraging the dependencies between these two variables, this method excludes unlikely combinations of the two, resulting in sharper prediction regions while still attaining the nominal coverage level. Through extensive experimentation on both simulated and real-world datasets, we confirm the validity and efficiency of these methods.

Contents

Abbreviations

A-NH	Attentive Neural Hawkes
ACC	Accuracy
APS	Adaptive Prediction Sets
C-CONST	Conformal Constant
C-HDR	Conformal Highest Density Regions
C-IND	Conformal Independent
C-PROB	Conformal Probability
C-QR	Conformalized Quantile Regression
C-QRL	Conformalized Quantile Regression Left
CCE	Conditional Coverage Error
CD	Critical Distance
CDF	Cumulative Distribution Function
CHR	Conformalized Histogram Regression
CONCAT	Concatenate
CONS	Constant
CP	Conformal Prediction
CRPS	Continuous Ranked Probability Score
DDSM	Deep Discrete-time State Space Models
EC	Exponential Constant
ECE	Expected Calibration Error
EHR	Electronic Health Records
FNN	Fully Neural Network

GLUE	General Language Understanding Evaluation
GMS	Gradient Magnitude Similarity
GRU	Gated Recurrent Unit
H-APS	Heuristic Adaptive Prediction Sets
H-HDR	Heuristic Highest Density Regions
H-QR	Heuristic Quantile Regression
H-QRL	Heuristic Quantile Regression Left
H-RAPS	Heuristic Reguralized Adaptive Prediction Sets
HDR	Highest Density Region
HPD	Highest Predictive Density
HPP	Homogeneous Poisson Process
i.i.d.	Independent and Identically Distributed
IPP	Inhomogeneous Poisson Process
LCONCAT	Log-Concatenate
LE	Learnable
LEWL	Learnable With Labels
LN	LogNorm
LNM	LogNormMix
LRA	Long Range Arena
LSE	Least-Square Estimation
LSTM	Long-Short Term Memory
LTO	Log-Times Only
MAE	Mean Absolute Error
MC	Marginal Coverage
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MLP	Multi-Layer Perceptron
MLP/MC	Monte-Carlo Multi-Layer Perceptron
MOOC	Massive Open Online Course
MRR	Mean Reciprocal Rank
MSE	Mean Squared Error
MSL	Mean Sequence Length
MTPP	Marked Temporal Point Processes

NH	Neural Hawkes
NLL	Negative Log-Likelihood
NLP	Natural Language Processing
PCE	Probabilistic Calibration Error
PDF	Probability Density Function
PIT	Probability Integral Transform
PMF	Probability Mass Function
RAPS	Reguralized Adaptive Prediction Sets
RMTPP	Recurrent Marked Temporal Point Process
RNN	Recurrent Neural Network
SA	Self-Attention
SA/CM	Cumulative Self-Attention
SA/MC	Monte-Carlo Self-Attention
SAN	Survival Analysis
SGD	Stochastic Gradient Descent
STPP	Spatio-Temporal Point Processes
TEM	Temporal
TEMWL	Temporal With Labels
TO	Times Only
TPI	Time Priority Index
TPP	Temporal Point Processes
UQ	Uncertainty Quantification
WSC	Worst Slab Coverage

List of Symbols

$\mathcal{S}_l$ — Realization of a M(TPP) as a sequence of n_l events.

$\mathcal{S}_{\text{train}}$ — Dataset of L training sequences.

$\mathcal{S}_{\text{b,train}}$ — Mini-batch of b training sequences from $\mathcal{S}_{\text{train}}$.

$\mathcal{S}_{\text{val}}$ — Dataset of L validation sequences.

$\mathcal{S}_{\text{test}}$ — Dataset of L test sequences.

$\boldsymbol{e}_i = (t_i, k_i) = (\tau_i, k_i)$ — Event with (inter-)arrival time t_i (τ_i) and mark k_i.

$\mathbb{K}$ — Mark space.

K — Total number of categorical marks.

T — Observation window.

$\mathbb{R}_+$ — Positive real numbers.

$\mathbb{R}^d$ — d-dimensional space.

$N(t)$ — Counting process at time t.

$N_k(t)$ — Counting process for mark k at time t.

$\mathbb{1}(\cdot)$ — Indicator function.

$\mathcal{H}_t$ — Observed history at time t.

$f(\cdot)$ — Probability density function (PDF).

$f^*(\cdot) = f(\cdot|\mathcal{H}_t)$ — PDF conditional on $\mathcal{H}_t$.

$F(\cdot)$ — Cumulative distribution function (CDF).

$F^*(\cdot)$ — CDF conditional on $\mathcal{H}_t$.

$p(\cdot)$ — Probability Mass Function (PMF).

$p^*(\cdot)$ — PMF conditional on $\mathcal{H}_t$.

$\lambda^*(t)$ — Ground intensity function conditional on $\mathcal{H}_t$.

$\lambda(t)$ — Ground intensity function.

λ — Constant ground intensity function.

$\lambda_k^*(t)$ — Marked conditional intensity functions conditional on $\mathcal{H}_t$.

$\lambda_k(t)$ — Marked conditional intensity functions.

λ_k — Constant marked conditional intensity functions.

$\Lambda^*(t)$ — Cumulative ground intensity function conditional on $\mathcal{H}_t$.

$\Lambda(t)$ — Cumulative ground intensity function.

$\Lambda_k^*(t)$ — Cumulative marked intensity functions conditional on $\mathcal{H}_t$.

$\Lambda_k(t)$ — Cumulative marked intensity functions.

$\mathbb{P}(A)$ — Probability of outcome A.

$\mathbb{E}[A]$ — Expectation of the random variable A.

$n!$ — Factorial of n.

$\text{Log}\mathcal{N}(\mu, \sigma)$ — Log-normal distribution with mean μ and variance σ^2.

$\psi(\cdot)$ — Triggering kernel in (2.40) or (2.41) for the univariate Hawkes process.

$\psi_{k,k'}(\cdot)$ — Triggering kernel between marks k and k' in (2.43) for the multivariate Hawkes process.

β, γ, ω — Model parameters.

$|A|$ — Size of a set A.

$\boldsymbol{\theta} \in \Theta$ — Trainable parameters belonging to some parameter space Θ.

$\prod$ — Product operator.

$\sum$ — Sum operator.

$\mathcal{L}(\cdot)$ — Objective function.

$\nabla_{\boldsymbol{\theta}}$ — Differential operator with respect to $\boldsymbol{\theta}$.

η — Learning rate.

S^f, S^p, S^F — Consistent scoring rules for PDF, PMF, and CDF, respectively.

υ — Sample from a uniform distribution.

f^{-1} — Inverse transform of a function f.

π — Permutation operator.

$\mathcal{S}_Z = \{z_1, ..., z_n\}$ — Realization of a univariate unit rate HPP.

$(\Lambda^*)^{-1}$ — Inverse transformation of the ground compensator Λ^*.

ν — Sample from an exponential distribution with unit rate.

$m(t)$ — Upper bound on $\lambda^*(t; \boldsymbol{\theta})$ for Ogata's modified thinning algorithm.

$\mathcal{L}_T(\boldsymbol{\theta}; \mathcal{S})$ — Time component of the NLL evaluated on a sequence $\mathcal{S}$.

$\mathcal{L}_M(\boldsymbol{\theta}; \mathcal{S})$ — Mark component of the NLL evaluated on a sequence $\mathcal{S}$.

$\tilde{\tau}$ — Estimate of the inter-arrival time of the next event.

$\tilde{k}$ — Estimate of the mark of the next event.

$t^\star$ — Survival time in survival analysis.

$S(t)$ — Survival function.

$\mathcal{X}$, $\mathcal{Y}$ — Domains of definition of random variables X and Y, respectively.

$\mathcal{F}$ — Space of distributions.

$d(A, B)$ — Distance function between A and B, as in (2.79).

$\mathcal{D}$ — dataset.

$\mathcal{D}_{\text{train}} \subset \mathcal{D}$ — Training partition of $\mathcal{D}$.

$\mathcal{D}_{\text{val}} \subset \mathcal{D}$ — Validation partition of $\mathcal{D}$.

$\mathcal{D}_{\text{test}} \subset \mathcal{D}$ — Test partition of $\mathcal{D}$.

$\bar{F}$ — Empirical CDF of PITs in (2.80).

p — Probability/Confidence level.

P_{XY} — Joint distribution of random variables X and Y.

$R(X_{n+1})$ — Prediction region for a new test input X_{n+1}.

$\hat{R}_\tau(\boldsymbol{h}_{n+1})$ — Prediction region for τ_{n+1} on a new test sequence $\mathcal{S}_{n+1}$.

$\hat{R}_k(\boldsymbol{h}_{n+1})$ — Prediction region for k_{n+1} on a new test sequence $\mathcal{S}_{n+1}$.

$\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1})$ — Prediction region for (τ_{n+1}, k_{n+1}) on a new test sequence $\mathcal{S}_{n+1}$.

α — Miscoverage level.

$Q(\cdot)$ — Quantile function.

$s(X_i, Y_i) = s_i$ — Non-conformity score between inputs X_i and Y_i. See (2.88) for an example.

$\hat{q}$ — Adjusted $1 - \alpha$ empirical quantile of the calibration scores in (5.5).

$\lceil \cdot \rceil$ — Ceiling operator.

$u(\cdot)$ — Event encoder.

$\mathrm{ENC}(\cdot)$ — History encoder.

$\boldsymbol{l}_i \in \mathbb{R}^{d_l}$ — Event encoding of event $\boldsymbol{e}_i$.

$\boldsymbol{h}_i \in \mathbb{R}^{d_h}$ — History encoding of event $\boldsymbol{e}_i$. See (3.10) for an example.

$\boldsymbol{l}_i^t \in \mathbb{R}^{d_l^t}$ — Time encoding of t_i. See (D.3) for an example.

$\boldsymbol{l}_i^k \in \mathbb{R}^{d_l^k}$ — Mark encoding of k_i as in (3.5).

$\oplus$ — Concatenation operator.

σ — Activation function.

$\boldsymbol{w}$ — Weight vector.

$\mathbf{W}$ — Weight matrix.

$\boldsymbol{b}$ — Bias vector.

$\boldsymbol{q}(t)$ — query vector in (3.14).

$\mathbf{K}$ — Key matrix in (3.14).

$\mathbf{V}$ — Value matrix in (3.14).

$\mathcal{T}_T$ — Time prediction task.

$\mathcal{T}_M$ — Mark prediction task.

$\boldsymbol{g}_T$ — Vector of gradients of $\mathcal{T}_T$ with respect to parameters $\boldsymbol{\theta}$.

$\boldsymbol{g}_M$ — Vector of gradients of $\mathcal{T}_T$ with respect to parameters $\boldsymbol{\theta}$.

ϕ_{TM} — Angle between the gradients of tasks $\mathcal{T}_T$ and $\mathcal{T}_T$. See Definition 1.

$|| \cdot ||_2$ — The l_2 norm.

$\hat{g}(\cdot; \boldsymbol{\theta})$ — Estimate of a function $g(\cdot)$ with learnable parameters $\boldsymbol{\theta}$. This notation is only used in Chapter 5.

$r(k) = |\{ k' \in \mathbb{K} : \hat{p}(k' \mid \boldsymbol{h}) \geq \hat{p}(k|\boldsymbol{h}) \}|$ — Ranking of the observed mark k among the probabilities in $\hat{p}(\cdot|\boldsymbol{h})$.

$(x)^+$ — Positive part of x.

$z_{1-\alpha}$ — Threshold at level $1 - \alpha$ in (5.32).

$\emptyset$ — Empty set.

$\boldsymbol{v}$ — Slab in (5.51).

$\mathbb{S}^d$ — d-dimensional simplex.

ϵ — Small offset value.

$C_i \in R^d$ — Centroïd obtained from the k-means++ algorithm.

$\mathcal{A} \subseteq \mathbb{R}^d$ — Partition of $\mathbb{R}^d$.

CHAPTER 1

Introduction

Many of our daily activities can naturally be represented as sequences of events unfolding along a continuous timeline. As an illustrative example, consider the task of monitoring the activity of various individuals purchasing items on an online shopping platform. By recording the precise times at which each item is purchased, we can generate a sequence of *timestamps* that reflects the unique shopping behavior of each individual customer. Moreover, for each purchase, we may have access to additional information regarding each purchase, such as its price, or the category to which the item belongs. Such complementary information, usually referred to as *marks*, can either be discrete (e.g. the item's category) or continuous (e.g. the item's price), and provides a richer context for understanding individual purchasing patterns.

Given the purchase history of these multiple individuals, we may want to predict the likely timings and marks of future purchases for both existing and new costumers, allowing the platform to e.g. better manage stocks, or to provide personalized recommendations. Importantly, we do not know in advance how many items will be bought by a given individual, nor can we expect regular intervals between purchases, making this task challenging. Moreover, it is reasonable to assume that purchases exhibit complex inter-dependencies, implying that future transactions are directly influenced by an individual's past shopping behavior. For example, purchasing a bike may increase the likelihood of buying protective gear shortly after, while simultaneously lowering the likelihood of acquiring a brand new car. Hence, modeling the complex dynamics of events occurrences becomes crucial in predicting future purchases from past observations. Figure 1.1 shows an illustration of event sequences for different individuals in our online shopping example. Naturally, such sequences arise in a variety of scenarios that extend well beyond our online shopping example, such as the activity of an individual on social media, records of health conditions, or even earthquake occurrences.

Marked Temporal Point Processes (MTPP) (Daley & Vere-Jones, 2008; Rasmussen, 2018) provide a principled mathematical framework for modeling these sequences of marked events. In a MTPP, event occurrences are typically characterized by *conditional marked intensity functions*, which specify the instantaneous rate of marked events arrivals on the continuous timeline given past observations. Defining a MTPP model is thus usually achieved by specifying a parametric form for each of these conditional marked intensity functions, enabling future predictions on the evolution of the system.

Early developments in MTPP modeling can be traced back to the beginning of the 20th century, where they originally found applications in modeling the

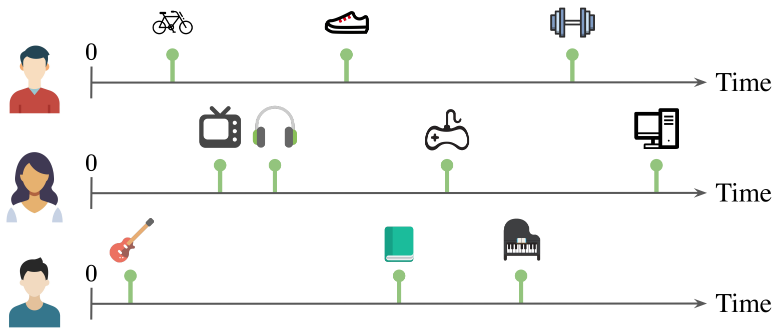


Figure 1.1: Illustration of event sequences for different individuals in our online shopping example.

random arrivals of telephone calls (Erlang, 1909) and the decay of radioactive materials (Bateman, 1910). Nevertheless, it is only during the period that followed the second world war that a substantial growth in the field of MTPP theory was observed (Feller, 1949; Cox, 1955, 1966; Lewis, 1972; Lewis & Shedler, 1979), with subsequent applications to various fields including inventory control (Morse, 1958), reliability engineering (Cox, 1962; Barlow & Proschan, 1965), neurobiology (Stein, 1965; Fatt & Katz, 1965), and seismology (Ogata, 1988, 1998). Among these precursor models, the Hawkes process (Hawkes, 1971) is a well-known example of MTPP model where past observations are assumed to "trigger" new events. Although first proposed in the 1970s, the Hawkes process still remains widely employed to model event dynamics in diverse domains, including finance (Bacry & Muzy, 2014), crime analysis (Mohler et al., 2011), user recommendations (Du et al., 2015), seismology (Ogata, 1988; Rotondi & Varini, 2019), and information diffusion on social media (Rizoiu et al., 2017b).

A key challenge in MTPP modeling amounts to specifying a parametric form of the model that is flexible enough to capture the complex cross-temporal dependencies between past event occurrences and future ones. However, early classical parametrizations often rely on strong modeling assumptions regarding the underlying process, which can limit their ability to capture general dynamics of events arrivals (Mei & Eisner, 2017). For instance, the Hawkes process assumes that past observations have an exciting influence on future occurrences, which prevents its effective deployment to scenarios that include complementary effects, e.g. when past events can also have an inhibiting influence on future ones. To address these limitations, deep learning techniques have been introduced into the MTPP literature, enabling the design of a more flexible class of *neural MTPP models*. In contrast to their classical counterparts that generally require tedious feature engineering, these neural MTPP models have the potential to learn any patterns in a purely data-driven manner (Shchur, 2022).

Since its introduction by the seminal work of Du et al. (2016), the field of neural MTPP modeling has experienced rapid development, with the emergence of numerous novel architectures (Shchur et al., 2020a; Zuo et al., 2020; Yang et al., 2022; Li et al., 2023) and applications (Upadhyay et al., 2018; Trivedi et al., 2019; Gupta et al., 2021; Shchur et al., 2021a). However, despite witnessing significant progress in recent years, neural MTPP modeling still faces a number of open challenges that warrant further exploration from the community. In the following section, we outline these challenges in terms of concrete research questions, and provide an overview of our contributions towards addressing them.

1.1. Challenges and Aims

To capture complex dependencies between event occurrences, neural MTPP models typically involve a combination of different architectural components, each tailored to model different aspects of a sequence. Improvements with respect to existing baselines are usually obtained by proposing alternatives to either of these components, often drawing inspiration from advances in deep learning techniques. For instance, one can replace a Recurrent Neural Network (RNN) architecture (Du et al., 2016; Mei & Eisner, 2017) with a self-attentive one (Zuo et al., 2020; Zhang et al., 2020), or choose to parametrize a certain MTPP function that leads to useful properties, such as reduced computational costs (Omi et al., 2019) or closed-form sampling (Shchur et al., 2020a,b).

However, as pointed out by Shchur et al. (2021b), "*new architectures often change all these components at once, which makes it hard to pinpoint the source of empirical gains*". Moreover, the baselines against which a newly proposed architecture is compared, as well as the datasets employed and the experimental setups, often differ between studies, which renders a fair comparison even harder. Identifying these sources of improvements in predictive accuracy is further challenged by the use of metrics that do not always clearly isolate where progress is being made. Implicitly, fitting marked event sequence data involves learning joint distributions of arrival times and marks, conditional on the observed history. These joint distributions can be factorized into the product of a time predictive distribution and a mark predictive distribution. In practice, the Negative Log-Likelihood (NLL) computed over test sequences is often used to evaluate the predictive performance of neural MTPP models. However, reporting a single NLL value encompasses the contributions of both of these predictive distributions simultaneously, which makes it difficult to assess model performance with respect to each task separately.

Besides evaluation, another challenge concerns the way neural MTPP models are learned from data. As mentioned, learning a neural MTPP model with the NLL objective function can be interpreted as a two-task learning problem, where the time prediction task involve learning a time predictive distribution, while the mark prediction task involves learning a mark predictive distribution. As common practice in the neural MTPP literature, these two tasks are optimized jointly on a common set of shared parameters. While parameter sharing between tasks can sometimes enhance training efficiency (Standley et al., 2020), it may also result in performance degradation when compared to training each task independently. A major challenge in the simultaneous optimization of multi-task objectives is the issue of conflicting gradients (Liu et al., 2021a). This term describes situations where task-specific gradients point in opposite directions. When such conflicts arise, gradient updates tend to favor tasks with larger gradient magnitudes, thus hindering the learning process of other concurrent tasks and adversely affecting their performance. Although the phenomenon of conflicting gradients has been studied in various fields (Chen et al., 2018, 2020; Yu et al., 2020; Javaloy & Valera, 2022; Shi et al., 2023), its impact on the training of neural MTPP models remains unexplored.

Additionally, while the NLL is a proper scoring rule that enables comparison across different baselines, it is difficult to interpret as a standalone metric (Shchur et al., 2021b). Probabilistic calibration, in the context of forecasting theory, refers to the statistical consistency between the predictive distributions and the realized outcomes. It is a desirable property that any competent or ideal forecaster should possess, fostering trust in the predictions returned by the model (Guo et al., 2017; Dheur & Ben Taieb, 2023). Nonetheless, with a few exceptions exemplified by the works of Xiao et al. (2017a); Zhang et al. (2020); Lin et al. (2022), the probabilistic calibration of both time and mark predictive distributions has been generally overlooked in the context of neural MTPP models. These considerations collectively highlight the necessity of carefully designed experimental studies to accurately evaluate the true benefits and limitations of current neural MTPP models.

A final challenge relates to the reliability of the predictions extracted from the trained MTPP models. Consider a critical application domain, such as diagnostic medicine, where the goal is to estimate the arrival time and mark of a future physical condition based on the patient's medical history. In such context, returning a single-valued prediction may be deemed unsatisfactory for safe decision-making, and we may strive instead to provide a high probability prediction region for the next arrival time, mark, or both, that faithfully reflects the uncertainty in the predictions. This region should typically include a subset of potential values that are highly likely to occur, aligned with a nominal probability coverage level. However, due to model misspecification or lack of training data, the MTPP model may provide a poor approximation of the unknown underlying process. Consequently, prediction regions derived solely from the model's estimates may be unreliable, failing to accurately reflect the true underlying uncertainty. To address this challenge, we build on the statistically sound framework of Conformal Prediction (CP) (Vovk et al., 2005) to construct reliable prediction regions for arrival times and marks of future events.

CP is a principled framework that enables the construction of distribution-free prediction regions, offering a finite-sample coverage guarantee even when the base model is unreliable. CP has found successful applications in a broad spectrum of machine learning fields, such as computer vision (Angelopoulos et al., 2021), natural language processing (Campos et al., 2024), or time series forecasting (Gibbs & Candès, 2021; Zaffran et al., 2022). Nevertheless, to the best of our knowledge, this thesis is the first work to connect the fields of CP with neural MTPP modeling. It should be noted that Candès et al. (2023) and Gui et al. (2023) explored CP in the closely related field of survival analysis. However, these works have primarily focused on univariate survival times, whereas MTPPs also involve categorical marks, making their settings not directly transferable to ours.

1.2. Objectives and Research Questions

To address the challenges discussed in the previous section, we make three main contributions in this thesis. First, we introduce novel neural MTPP models for probabilistic predictive modeling of continuous-time event data. Second, we present new training strategies for neural MTPP models designed to improve their predictive accuracy on the time and mark prediction tasks. Third, we develop reliable and distribution-free conformal methods for quantifying the uncertainty in the predictions extracted from neural MTPP models.

A first step towards these contributions involves identifying the limitations of existing neural MTPP models in terms of predictive accuracy across a wide range of real-world scenarios. After an overview of the fundamental concepts in MTPP modeling and uncertainty quantification in Chapter 2, we address our first research question.

Research Question 1: *What architectural design choices lead to significant improvements in predictive accuracy for neural MTPP models?*

In Chapter 3, we present how neural architectures can be combined with the core principles of MTPP theory to develop flexible neural MTPP models. To assess their predictive accuracy across a wide range of real-world scenarios, we then perform a large-scale experimental study on 15 real-world event sequence datasets in a carefully designed and unified setup. For thoroughness, our study also includes classical MTPP models as well as synthetic datasets. Specifically, we study the influence of each major architectural component on model performance, from the perspectives of both time and mark prediction tasks. Finally, we employ standard metrics and evaluation tools borrowed from the forecasting literature to evaluate the calibration of neural MTPP models. Our extensive set of experiments reveals that modern neural MTPP models frequently demonstrate limitations both in terms of predictive accuracy and calibration. In Chapter 4, we outline an underlying cause of this problem: common parametrizations of neural MTPP models frequently lead to the emergence of conflicting gradients when trained on the NLL, which negatively impact predictive performance on both time and mark prediction tasks. Investigating deeper into this challenge leads to our second research question.

Research Question 2: *How can conflicting gradients be effectively mitigated during the training of neural MTPP models?*

To prevent the emergence of conflicting gradients, we introduce novel parametrizations for existing neural MTPP models, allowing for separate modeling and training of time and mark prediction tasks. Additionally, we present a simple, yet effective parametrization for the mark predictive distribution that relaxes the assumption of conditional independence between arrival times and marks made in Du et al. (2016) and Shchur et al. (2020a). Through extensive experiments, we demonstrate that our framework effectively prevents the emergence of conflicting gradients during training, thereby enhancing the predictive accuracy of the models. Unfortunately, improved predictive accuracy does not guarantee reliability of the estimates, meaning that prediction regions derived directly from the model may still poorly reflect the true uncertainty. This concern brings us to our third and final research question.

Research Question 3: *How can we generate informative and distribution-free prediction regions for future event times and marks with finite-sample coverage guarantees even when the neural MTPP model is unreliable?*

In Chapter 5, we develop more reliable methods for uncertainty quantification in neural MTPP models via the framework of conformal prediction. A primary objective is to generate distribution-free joint prediction regions for events' arrival times and marks, with a finite-sample marginal coverage guarantee. In this context, a key challenge is to handle a bivariate response composed of a continuous arrival time and a discrete mark, without making distributional assumptions. We first consider a simple but overly conservative approach that combines individual prediction regions for the event's arrival time and mark. Then, we introduce a more effective method based on bivariate highest density regions derived from the joint predictive density of arrival times and marks. By leveraging the dependencies between these two variables, this method excludes unlikely combinations of the two, resulting in sharper prediction regions while still attaining the pre-specified coverage level. We also explore the generation of individual univariate prediction regions for events' arrival times and marks through conformal regression and classification techniques. Moreover, we evaluate the stronger notion of conditional coverage. Finally, through extensive experimentation on both simulated and real-world datasets, we assess the validity and efficiency of these methods.

1.3. Contributions

The contents of Chapters 3 to 5 are primarily based on research papers submitted or published in international peer-reviewed conferences and journals. For each publication, we include a link to a public repository that contains the personal codebase allowing to reproduce all experiments presented in this thesis.

- **Chapter 3**: On the Predictive Accuracy of Neural MTPP Models.
    1. **Tanguy Bosser** & Souhaib Ben Taieb (2023a). On the Predictive Accuracy of Neural Temporal Point Process Models for Continuous-time Event Data. In *Transactions of Machine Learning Research (TMLR)*. Survey Certification.
        - **Repository:** https://github.com/tanguybosser/ntpp-tmlr2023.
- **Chapter 4**: Preventing Conflicting Gradients in Neural MTPP Models.
    1. **Tanguy Bosser** & Souhaib Ben Taieb (2024b). Preventing Conflicting Gradients in Neural Marked Temporal Point Processes. *Under Review for Transactions in Machine Learning Research (TMLR)*.
        - **Repository:** https://github.com/tanguybosser/grad_tpp.
    2. **Tanguy Bosser** & Souhaib Ben Taieb (2023b). Revisiting the Mark Conditional Independence Assumption in Neural Marked Temporal Point Processes. In *Proceedings of the 31st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*.
        - **Repository:** https://github.com/tanguybosser/grad_tpp.
- **Chapter 5**: Distribution-free Conformal Prediction Regions for Neural MTPP Models.
    1. Victor Dheur, **Tanguy Bosser**[1], Rafael Izbicki and Souhaib Ben Taieb (2024a). Distribution-Free Conformal Joint Prediction Regions for Neural Marked Temporal Point Processes. In *Machine Learning, 113, 7055–7102*.
        - **Repository:** https://github.com/tanguybosser/conf_tpp.

[1]Victor Dheur and Tanguy Bosser contributed equally to this work.

CHAPTER 2

Background on MTPPs and Uncertainty Quantification

The questions addressed in this thesis are at the intersection of Temporal Point Processes (TPP) and approaches to quantify the uncertainty of probabilistic models. This chapter therefore aims to provide the reader with a working understanding of the foundational concepts from both domains that will be built upon in subsequent chapters.

Specifically, we begin our discussion with TPP, a framework for modeling systems that generate variable-length event sequences in continuous-time. We present how the evolution of such systems can be fully characterized by adopting an auto-regressive perspective, together with the definitions of classical TPP models. We then show how we can learn these TPP models from event sequence data, and how the learned model can then be used for downstream prediction tasks on new sequences.

In the second part of the chapter, we pursue our discussion with approaches to quantify the uncertainty in the predictions made by any arbitrary probabilistic model. Uncertainty Quantification (UQ) is a broad and active field of research, with numerous methods and applications extending beyond the scope of this dissertation. Consequently, for the sake of conciseness, we will narrow our discussion to the concepts of probabilistic calibration and conformal prediction.

For additional material on TPP, the reference books of Daley & Vere-Jones (2003) and Daley & Vere-Jones (2008) provide a thorough overview of the theory of point processes, while Laub et al. (2015); Rasmussen (2018); De et al. (2019); Shchur et al. (2021b) and Shchur (2022) offer a more gentle introduction. For further references on uncertainty quantification, the book of Ghanem et al. (2017) proposes a broad overview of the key concepts pertaining to the field, while the recent surveys of Abdar et al. (2021) and Gawlikowski et al. (2023) focus on the application of uncertainty quantification to deep neural networks.

2.1. Marked Temporal Point Processes

A **Marked Temporal Point Process** (MTPP) is a random process whose realization is a sequence of n events $\mathcal{S} = \{\boldsymbol{e}_i = (t_i, k_i)\}_{i=1}^{n}$, where the number n of events is random. Each event $\boldsymbol{e}_i \in \mathcal{S}$ is an ordered pair with an *arrival time* $t_i \in [0, T]$ (with $t_0 = 0$) and a *mark* k_i belonging to some mark space $\mathbb{K}$. Depending on the problem considered, $\mathbb{K}$ can either be continuous (e.g. the price of an item), or discrete (e.g. the category of an item). In this thesis, we focus exclusively on categorical marks, and assume that $k_i \in \mathbb{K} = \{1, ..., K\}$

with $K = |\mathbb{K}|$ the size of the mark space $\mathbb{K}$.

The arrival times form a sequence of strictly increasing random values observed within a specified time interval $[0, T]$, i.e. $0 \leq t_1 < t_2 < \ldots < t_n \leq T$. Note that we will frequently use the equivalent representation $\boldsymbol{e}_i = (\tau_i, k_i)$, where $\tau_i = t_i - t_{i-1} \in \mathbb{R}_+$ is an event *inter-arrival time*, i.e. the elapsed time between two events. Alternatively, a realization of a MTPP can be represented by a counting process $N(t) = \sum_{k=1}^{K} N_k(t)$ with

$$N_k(t) = \sum_{i=1}^{n} \mathbb{1}(t \geq t_i \ \cap \ k_j = k) \quad \text{for } t \in [0, T], \tag{2.1}$$

where $\mathbb{1}(\cdot)$ is the indicator function that equals 1 if its argument is true, 0 otherwise. Essentially, $N_k(t)$ counts the number of events of mark k in $\mathcal{S}$ that occurred prior to t, while $N(t)$ counts the total number of events that occurred prior to t. Note that throughout this dissertation, we make the usual assumption that the MTPP is *simple*, i.e. two arrival times coincide with probability 0, meaning that the events are strictly ordered in time.

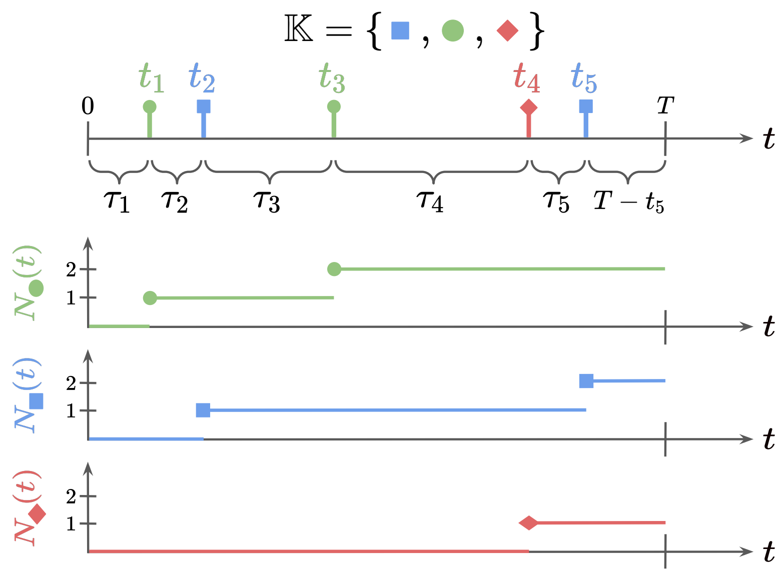
Figure 2.1 illustrates the realization of a MTPP in $[0, T]$, both from the perspectives of the sequence and counting process representations. This realization can, for instance, represent the activity of an individual on an online shopping platform. In such scenario, t_1 to $t_5 \in \mathbb{R}_+$ would be the times at which the purchases are made, while k_1 to $k_5 \in \mathbb{K}$ would correspond to the categories of the items bought with, e.g. $\mathbb{K} = \{$■ $=$ toy, ● $=$ shoe, ◆ $=$ book$\}$. Each time a new item is purchased, the counting process associated to its category is simply incremented by one unit.

2.1.1 Characterizing a MTPP

There are several ways to fully characterize the occurrences of arrival times and marks in an MTPP. As a first approach, we can characterize the process by specifying the distributions of events conditional on their pasts. Let $f_i(\boldsymbol{e}|\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1})$ denote the Probability Density Function (PDF) of an event $\boldsymbol{e}_i$ conditional on all preceding occurrences $\{\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1}\}$. By the chain rule, the PDF $f(\mathcal{S})$ of a realization $\mathcal{S}$ writes

$$f(\mathcal{S}) = f_1(\boldsymbol{e}) f_2(\boldsymbol{e}|\boldsymbol{e}_1) ... f_n(\boldsymbol{e}|\boldsymbol{e}_1, ...\boldsymbol{e}_{n-1}), \tag{2.2}$$

which essentially reveals that a MTPP can be treated as an auto-regressive process, where events are generated one by one conditional on all previous observations. Consequently, by specifying a PDF $f_i(\boldsymbol{e}|\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1})$ for each



$\mathbb{K} = \{ ■, ●, ◆ \}$
t_1 t_2 t_3 t_4 t_5
0
T
t
τ_1 τ_2 τ_3 τ_4 τ_5 $T - t_5$
$N_●(t)$
$N_■(t)$
$N_◆(t)$
1
2

Figure 2.1: Realization of a MTPP with $K = 3$ in $[0, T]$ viewed as a sequence $\mathcal{S} = \{(t_i, k_i)\}$ for $i = 1, .., 5$ (top), and as K counting processes $N_k(t)$ for $t \in [0, T]$ (bottom).

event $\boldsymbol{e}_i \in \mathcal{S}$, and by stitching these conditional PDFs together, the sequence $\{f_1(\boldsymbol{e}), f_2(\boldsymbol{e}|\boldsymbol{e}_1), ..., f_n(\boldsymbol{e}|\boldsymbol{e}_1, ..., \boldsymbol{e}_{n-1})\}$ fully characterizes event occurrences in a MTPP.

Let $\mathcal{H}_t = \{(t_j, k_j) \in \mathcal{S} \mid t_j < t\}$ denote the *history* of the process at time t, i.e. all events in $\mathcal{S}$ that occurred prior to t, meaning that $f_i(\boldsymbol{e}|\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1}) = f_i(\boldsymbol{e}|\mathcal{H}_t)$. As $f_i(\boldsymbol{e}|\mathcal{H}_t)$ defines a joint PDF for the bivariate event $\boldsymbol{e} = (t, k)$ conditional on the history, it will be referred to as the *joint PDF of arrival times and marks* that defines

$$f_i(t, k|\mathcal{H}_t)dt = \mathbb{P}[T_i \in [t, t + dt], K_i = k|\mathcal{H}_t], \tag{2.3}$$

where dt refers to an infinitesimal change in t, and T_i and K_i are the random variables associated to the arrival time and mark of the i^{th} event, respectively. In other terms, the joint PDF specifies the instantaneous probability to observe an event of mark k at time t, conditional on $\mathcal{H}_t$. For clarity, we will use the notation "$*$" of Daley & Vere-Jones (2008) to indicate dependence on $\mathcal{H}_t$, i.e. $f_i^*(t, k) = f_i(t, k|\mathcal{H}_t)$. Additionally, we will omit the suffix "i" from the notations, implying that a function will be systematically defined for the i^{th}

event in $\mathcal{S}$, i.e. $f^*(t,k) = f_i^*(t,k)$. We can further factorize the joint PDF as

$$f^*(t,k) = f^*(t)p^*(k|t), \tag{2.4}$$

where $f^*(t)$ is the *PDF of arrival times*, and $p^*(k|t)$ is *the conditional Probability Mass Function (PMF) of marks*, and are both conditional on the history $\mathcal{H}_t$. Equivalently, we have

$$f^*(t,k) = f^*(t|k)p^*(k), \tag{2.5}$$

where $f^*(t|k)$ is the *conditional PDF of arrival times*, while $p^*(k)$ is the *marginal PMF of marks.* From $f^*(t,k)$, we can also define the *joint Cumulative Distribution Function (CDF) of arrival times and marks* as

$$F^*(t,k) = \int_{t_{i-1}}^{t} f^*(s,k)ds = \mathbb{P}\left[T_i \in [t_{i-1},t], K_i = k|\mathcal{H}_t\right], \tag{2.6}$$

which corresponds to the probability of observing an event of mark k in the interval $[t_{i-1},t]$, conditional on $\mathcal{H}_t$. As previously mentioned, other functions than the joint PDF or CDF of arrival times and marks can be employed to fully characterize future event occurrences in a MTPP from past observations. Specifically, an alternative consists in specifying K *marked intensity functions* $\lambda_k^*(t)$, which are formally defined for $t > t_{i-1}$ as

$$\lambda_k^*(t) = \lim_{\Delta t \downarrow 0} \frac{\mathbb{E}[N_k(t+\Delta t) - N_k(t)|\mathcal{H}_t]}{\Delta t} = \frac{f^*(t,k)}{1 - F^*(t)}, \tag{2.7}$$

where $F^*(t) = \sum_{k=1}^{K} F^*(t,k)$ is the *CDF of arrival times.* In essence, the marked intensity $\lambda_k^*(t)$ gives the expected occurrence *rate* of events of mark k per unit of time, conditional on $\mathcal{H}_t$. In addition, $\lambda_k^*(t)$ possess the following heuristic interpretation

$$\lambda_k^*(t)dt \simeq \mathbb{P}\big[T_i \in [t,t+dt], K_i = k|\mathcal{H}_t, T_i \notin [t_{i-1},t]\big]. \tag{2.8}$$

In other words, $\lambda_k^*(t)$ gives the instantaneous probability of observing the next event of mark k in an infinitesimal interval around t, conditional on both $\mathcal{H}_t$ and on the fact that the event has not been observed in $[t_{i-1},t]$. Similarly to (2.4), $\lambda_k^*(t)$ can be factorized as

$$\lambda_k^*(t) = \frac{f^*(t)p^*(k|t)}{1 - F^*(t)} = \lambda^*(t)p^*(k|t), \tag{2.9}$$

$$\begin{tikzcd}
F^*(t,k) \arrow[r, shift right, "\frac{dF^*(t,k)}{dt}"'] \arrow[rd, dash] & f^*(t,k) \arrow[l, shift right, "\int_{t_{i-1}}^{t} f^*(s,k)ds"'] \arrow[d, shift right, "\frac{f^*(t,k)}{1-F^*(t)}"'] \\
\Lambda_k^*(t) \arrow[r, shift right, "\frac{d\Lambda_k^*(t)}{dt}"'] & \lambda_k^*(t) \arrow[l, shift right, "\int_{t_{i-1}}^{t} \lambda_k^*(s)ds"'] \arrow[u, shift right, "\lambda_k^*(t)\exp\left(-\sum_{k=1}^{K}\Lambda_k^*(t)\right)"']
\end{tikzcd}$$

Figure 2.2: Relations between $f^*(t,k), \lambda_k^*(t), F^*(t,k)$ and $\Lambda_k^*(t)$. Drawing inspired by Yan (2019b).

where

$$\lambda^*(t) = \sum_{k=1}^{K} \lambda_k^*(t) = \sum_{k=1}^{K} \frac{f^*(t,k)}{1-F^*(t)} = \frac{f^*(t)}{1-F^*(t)}, \tag{2.10}$$

is the *ground intensity* of the process, i.e. the expected occurrence rate of events of *any* mark conditional on $\mathcal{H}_t$. Finally, from $\lambda_k^*(t)$, we can define K *cumulative marked intensity functions* for $t \geq t_{i-1}$ as

$$\Lambda_k^*(t) = \int_{t_{i-1}}^{t} \lambda_k^*(s)ds, \tag{2.11}$$

which bears the following relation to $f^*(t,k)$ and $\lambda_k^*(t)$:

$$f^*(t,k) = \lambda_k^*(t)\exp\left(-\Lambda^*(t)\right), \tag{2.12}$$

where $\Lambda^*(t) = \sum_{k=1}^{K} \Lambda_k^*(t)$ is the *cumulative ground intensity functions* of the process. We provide the proof for (2.12) in Appendix B.

We want to stress that any of the functions $f^*(t,k)$, $F^*(t,k)$, $\lambda_k^*(t)$, and $\Lambda_k^*(t)$ can be uniquely retrieved from the others, and hence, they all fully characterize a MTPP. A graphical representation of the relations between these functions is presented on Figure 2.2, while Figure 2.3 shows an illustration of their typical evolutions with time for a MTPP with $K = 3$ marks.

As any of the functions $f^*(t,k)$, $\lambda_k^*(t)$, $F^*(t,k)$ and $\Lambda_k^*(t)$ uniquely characterize a MTPP, it also implies that they implicitly define a valid joint distribution over arrival times and marks. This translates into the following properties (Rasmussen, 2018; Enguehard et al., 2020):

Property 1. $\forall k \in \mathbb{K}$, *the joint PDF of arrival times and marks* $f^*(t,k)$ *verifies*

R_1^f: $f^*(t,k) \geq 0.$ (2.13)

R_2^f: $\int_{t_{i-1}}^{\infty} \sum_{k=1}^{K} f^*(s,k)ds = 1.$ (2.14)

Property 2. $\forall k \in \mathbb{K}$, *the joint CDF of arrival times and marks* $F^*(t,k)$ *verifies*

R_1^F: $F^*(t,k) \in [0,1].$ (2.15)

R_2^F: $F^*(t_{i-1},k) = 0.$ (2.16)

R_3^F: $\lim\limits_{t\to\infty} \sum_{k=1}^{K} F^*(t,k) = 1.$ (2.17)

R_4^F: $dF^*(t,k)/dt \geq 0.$ (2.18)

Property 3. $\forall k \in \mathbb{K}$, *the marked intensity functions* $\lambda_k^*(t)$ *verify*

R_1^λ: $\lambda_k^*(t) \geq 0.$ (2.19)

R_2^λ: $\lim\limits_{t\to\infty} \int_{t_{i-1}}^{t} \lambda_k^*(s)ds = \infty.$ (2.20)

Property 4. $\forall k \in \mathbb{K}$, *the cumulative marked intensity functions* $\Lambda_k^*(t)$ *verify*

R_1^Λ: $\Lambda_k^*(t) > 0.$ (2.21)

R_2^Λ: $\Lambda_k^*(t_{i-1}) = 0.$ (2.22)

R_3^Λ: $\lim\limits_{t\to\infty} \Lambda_k^*(t) = \infty.$ (2.23)

R_4^Λ: $d\Lambda_k^*(t)/dt \geq 0.$ (2.24)

Properties R_1^f, R_2^f and R_1^F to R_4^F are implied by the definitions of valid PDFs and CDFs, respectively. Property R_1^λ results from the definition of $\lambda_k^*(t)$ in (2.8), while R_2^λ directly follows from R_3^F. Finally, R_1^Λ and R_2^Λ can be deduced from R_1^λ and R_2^λ, respectively, while R_3^Λ and R_4^Λ directly results from R_3^F and R_4^F. As we will see in the next sections, keeping these properties in mind is important when specifying a MTPP model: the chosen parametrization must define a valid joint distribution over arrival times and marks, meaning that we must ensure that the above properties are always satisfied.

Remark 1. By stating Properties 1-4, we consider that the MTPP is *non-terminating*, meaning that a new event will arrive eventually in the future with probability 1. This consideration is formally captured in R_2^f, R_3^F, R_2^λ, and R_3^Λ.

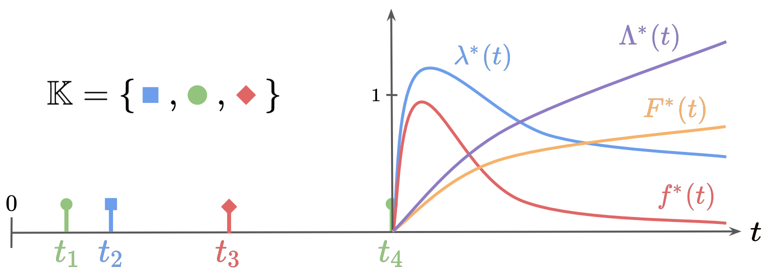


Figure 2.3: Illustration of the evolution of $f^*(t)$, $\lambda^*(t)$, $F^*(t)$, and $\Lambda^*(t)$ as a function of t for a MTPP with $K = 3$ marks. Here, t_4 corresponds to the arrival time of the last observed event.

Conversely, for a *terminating* MTPP, we would instead assume that the process eventually stops at some point in the future with probability $p < 1$, and that no more events are to be recorded. For such terminating MTPP, R_2^f, R_3^F, R_2^λ, and R_3^Λ as defined in the above must be adapted to

R_2^f: $\int_{t_{i-1}}^{\infty} \sum_{k=1}^{K} f^*(s,k)ds = 1 - p.$ R_2^λ: $\lim\limits_{t \to \infty} \int_{t_{i-1}}^{t} \lambda_k^*(s)ds = -\log p.$

R_3^F: $\lim\limits_{t \to \infty} \sum_{k=1}^{K} F^*(t,k) = 1 - p.$ R_3^Λ: $\lim\limits_{t \to \infty} \Lambda_k^*(t) = -\log p.$

Without loss of generality, we consider exclusively the setting of non-terminating MTPP in this dissertation, which implies that Properties 1-4 are systematically satisfied.

2.1.2 Classical Parametrizations of MTPP models

Early developments in MTPP can be traced back to the beginning of the 20th century, where they originally found applications in modeling the arrivals of telephone calls (Erlang, 1909) and the decay of radioactive materials (Bateman, 1910). Nevertheless, it is only during the period that followed the second world war that the field of MTPP theory witnessed substantial growth (Daley & Vere-Jones, 2008), with subsequent applications to various fields including inventory control (Morse, 1958), reliability engineering (Cox, 1962; Barlow & Proschan, 1965), neurobiology (Stein, 1965; Fatt & Katz, 1965), and seismology (Ogata, 1988, 1998).

In this section, we review several instances of these early models, including their basic properties, and how they can be used as building blocks to define more complex models. The term "classical" is used here to distinguish these seminal models from their neural counterparts, which will be introduced later in Chapter 3. For each process, we first consider the unmarked case, i.e. when the process does not include marks, meaning that a realization is characterized only by a sequence of arrival times. Their extensions to a marked scenarios directly follows in the discussion.

Homogeneous Poisson Process (HPP) (Kingman, 1992) is the earliest and simplest TPP model, characterized by a constant intensity function independent of the history:

$$\lambda^*(t) = \lambda, \tag{2.25}$$

where $\lambda \in \mathbb{R}^+$. Given the simple expression of the intensity in (2.25), the following important properties can be derived for the HPP:

1. The inter-arrival times $\tau \in \mathbb{R}_+$ are Independent and Identically Distributed (i.i.d.) random variables which follow an exponential distribution with mean $1/\lambda$, i.e.

$$f^*(\tau) = f(\tau) = \lambda \exp(-\lambda \tau). \tag{2.26}$$

2. The number of events $n_{ab} = N(b) - N(a)$ in any interval $[a, b]$ with $a < b \leq T$ follows a Poisson distribution with rate $\lambda(b-a)$, i.e.

$$\mathbb{P}\left(N = n_{ab}\right) = \frac{\lambda(b-a)^{n_{ab}}}{n_{ab}!} e^{-\lambda(b-a)}, \tag{2.27}$$

where $n_{ab}! = n_{ab}(n_{ab}-1)(n_{ab}-2)...1$ is the factorial of n_{ab}.

3. The number of events in two disjoints intervals are independent, also known as the *independent increment property.*

4. The HPP is *memoryless*, meaning that the distribution of inter-arrival times depends only on the current time, and not on any past information. Suppose that we are waiting for the arrival of a new event, and that τ' has already elapsed since the last event t_{i-1}, i.e. no event has been observed in $[t_{i-1}, t_{i-1} + \tau']$. For an additional waiting time τ'', the memoryless property of the HPP states that

$$\mathbb{P}(\tau > \tau' + \tau'' | \tau > \tau') = \mathbb{P}(\tau > \tau''), \tag{2.28}$$

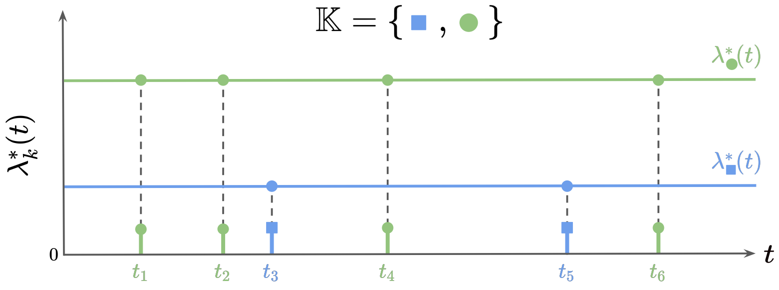


Figure 2.4: Illustration of the realization of a HPP with $K = 2$ marks and its marked intensity functions over time.

where τ refers to the random variable associated to the next inter-arrival time. In other words, the probability to wait an additional τ'' for the event to arrive when having already waited τ' is the same as the probability of waiting τ'' from the start.

These properties motivate the HPP being sometimes referred to as a *purely or completely random process* (Daley & Vere-Jones, 2008). Intuitively, events materialize at constant rate λ, creating sequences of arrival times evenly scattered along the continuous timeline. Extending the HPP to a marked scenario is easily achieved by defining K marked intensity functions that are constant over time, i.e.

$$\lambda^*_k(t) = \lambda_k, \tag{2.29}$$

where $k \in \mathbb{K}$ and $\lambda_k \in \mathbb{R}_+$. This leads to the following expression for the joint PDF of inter-arrival times and marks:

$$f^*(\tau, k) = f(\tau, k) = \lambda_k \exp\left(-\lambda\tau\right), \tag{2.30}$$

where $\lambda = \sum_{k=1}^{K} \lambda_k$. Figure 2.4 illustrates a realization of a HPP with $K = 2$ marks.

Inhomogeneous Poisson Process (IPP) (Daley & Vere-Jones, 2008) can be interpreted as a generalization of the HPP, where the intensity function now evolves over time:

$$\lambda^*(t) = \lambda(t), \tag{2.31}$$

for some function $\lambda(t) \geq 0 \; \forall t$. Note that, as in (2.25), the intensity function in (2.31) remains independent of the history $\mathcal{H}_t$. However, by allowing $\lambda(t)$ to vary with time, the IPP can capture changes in the rate of arrivals of events. For instance, when recording check-ins to restaurant open all day, one rightfully

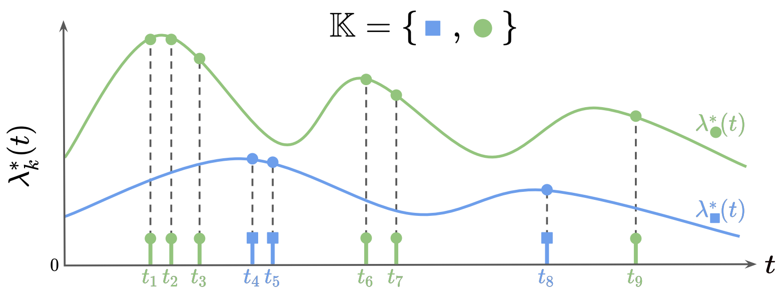


Figure 2.5: Illustration of a realization of a IPP with $K = 2$ marks and evolution of its marked intensity functions $\lambda_k^*(t)$ over time.

expects a greater number of events to occur during the day than during the night. Contrasting with the HPP, the IPP is able to capture such periodic fluctuations in the evolution of a system of interest.

Given the temporal evolution of the conditional intensity functions, the first two properties of the HPP are no longer valid, and need to be adjusted accordingly. Specifically, for a IPP, these properties become:

1. The inter-arrival times τ are independent random variables, but they are no longer identically distributed. In particular, if t_{i-1} is the arrival time of the last observed event, then the PDF of the next inter-arrival time τ_i is given by

$$f^*(\tau) = f(\tau) = \lambda(t_{i-1} + \tau)\exp\left(-\int_0^\tau \lambda(t_{i-1} + s)ds\right), \tag{2.32}$$

which depends on where t_{i-1} materialized on the continuous timeline.

2. The number of events $n_{ab} = N(b) - N(a)$ in any interval $[a, b]$ with $a < b \leq T$ follows a Poisson distribution with rate $\int_a^b \lambda(s)ds$, i.e.

$$\mathbb{P}\left(N = n_{ab}\right) = \frac{\left(\int_a^b \lambda(s)ds\right)^{n_{ab}}}{n_{ab}!} e^{-\int_a^b \lambda(s)ds}. \tag{2.33}$$

The time-varying intensity in (2.34) also causes the IPP to lose the memoryless property of the HPP. However, the independent increment property is retained by the process, meaning that the number of points in two disjoint intervals remain independent random variables.

Similarly to the HPP, we can define a marked version of the IPP by specifying K marked intensity functions:

$$\lambda_k^*(t) = \lambda_k(t), \tag{2.34}$$

for some functions $\lambda_k(t) \geq 0$ with $t \in \mathbb{R}_+$ and $k \in \mathbb{K}$. Figure (2.5) illustrates a realization of a marked IPP with $K = 2$ marks.

Renewal process (Cox, 1962). Recall that for a HPP, the inter-arrival times τ are i.i.d. random variables following an exponential distribution. The renewal process generalizes this concept to any arbitrary distribution independent of $\mathcal{H}_t$, i.e.

$$f^*(\tau) = f(\tau), \tag{2.35}$$

for some PDF $f(\cdot)$ defined on $\mathbb{R}_+$, such as the log-normal, Weibull or Gompertz distributions. Naturally, the exponential distribution in (2.26) retrieves the definition of the HPP. Given $F(\tau) = \int_0^\tau f(s)ds$, and assuming that t_{i-1} is the last observed event, the ground intensity function of a renewal process can be retrieved as

$$\lambda(t) = \lambda(t_{i-1} + \tau) = \frac{f(\tau)}{1 - F(\tau)}. \tag{2.36}$$

In a marked scenario, we assume that the bivariate events (τ, k) follow some arbitrary joint distribution independent of the history, with joint PDF and marked intensities expressed as

$$f^*(\tau, k) = f(\tau, k) = f(\tau)p(k|\tau), \tag{2.37}$$

$$\lambda_k^*(t) = \lambda_k(t_{i-1} + \tau) = \frac{f(\tau)p(k|\tau)}{1 - F(\tau)}, \tag{2.38}$$

where (2.37) and (2.38) are obtained from (2.4) and (2.9), respectively. Note that in the general case, while the pairs (τ, k) are assumed to be i.i.d. for a renewal process, $p(k|\tau)$ indicates that the existence of a dependency between inter-arrival times and marks is allowed. However, for simplicity, it is often assumed that these two quantities are independent of each other, yielding $p(k|\tau) = p(k)$ (Pandey & van der Weide, 2017). Figure 2.6 illustrates the evolution of $\lambda_k(t)$ as a function of time for a renewal process with $K = 2$ marks, where $f(\tau)$ is defined as a Log-$\mathcal{N}(0, 1)$ distribution, i.e. a log-normal distribution with mean 0 and variance 1.

In a broad sense, the term "renewal" underlines that the process *resets* itself every time a new event occurs. Typical examples of application of renewal processes appear in reliability engineering, where one may be interested in

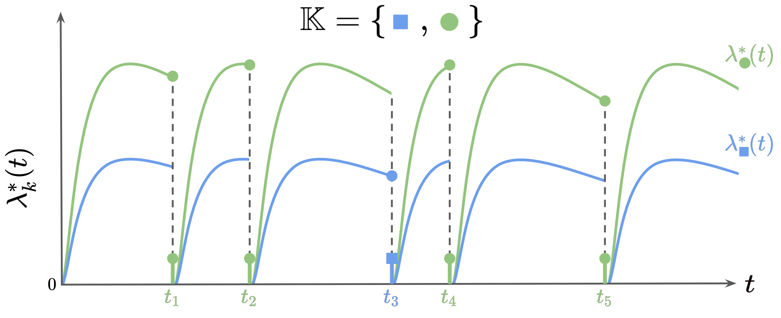


Figure 2.6: Illustration of a realization of a renewal process with $K = 2$ marks and evolution of the corresponding marked intensity functions $\lambda^*_k(t)$. We assume that $f(\tau)$ is defined as a Log-$\mathcal{N}(0,1)$ distribution with $p(k = ●) = 0.7$ and $p(k = ■) = 0.3$.

modeling the time between failures of industrial equipment. For instance, suppose that an industrial machine is prone to K different types of failures. If the time between two failures of any types follows the same distribution with density $f(\tau)$, and if the machine is integrally replaced any time a failure occurs (i.e. times to failures are independent), then the sequence of times and types of failures follows a renewal process.

Hawkes process (Hawkes, 1971; Liniger, 2009). In the processes considered thus far, events have been systematically assumed to arise independently of one another. However, in many real-world scenarios, we may reasonably assume that past observations directly influence the evolution of a system. For instance, we may expect that observed events increase the likelihood of future occurrences. In particular, earthquakes are known to generate aftershocks that can trigger the arrival of new earthquakes. The Hawkes process, also sometimes referred to as *self-exciting* process, explicitly models such behavior by defining an intensity function that jumps abruptly for each new observation in $\mathcal{H}_t$, i.e.

$$\lambda^*(t) = \lambda + \sum_{t_j \in \mathcal{H}_t} \psi(t - t_j), \tag{2.39}$$

where $\lambda \in \mathbb{R}_+$ denotes a *baseline* intensity rate, and $\psi : \mathbb{R}^+ \to \mathbb{R}^+$ is a general formulation of the so-called *triggering kernel*. A typical choice for ψ is the exponential triggering kernel (Hawkes, 1971; Ogata, 1988; Embrechts et al., 2011) defined as

$$\psi(t - t_j) = \gamma \exp\left(-\beta(t - t_j)\right), \tag{2.40}$$

where $\beta \in \mathbb{R}_+$ and $\gamma \in \mathbb{R}_+$. Under such kernel parametrization, the intensity function initially starts at λ, and jumps by the quantity γ each time a new event occurs, leading to the "self-exciting" denomination of the Hawkes process. Then, the excitation induced at time t_j reduces exponentially to the initial baseline rate λ as time rises, with the reduction controlled by the parameter β. Another popular choice for ψ is the power-law kernel, which may be defined as (Rizoiu et al., 2017a)

$$\psi(t - t_j) = \frac{\gamma}{(t - t_j + \omega)^{\beta+1}}, \tag{2.41}$$

where $\alpha, \beta, \omega > 0$. The power-law kernel found numerous practical use cases, notably for modeling earthquake aftershocks in seismology (Ogata, 1988), or for modeling information diffusion on social media (Rizoiu et al., 2017b).

The term $b = \int_0^\infty \psi(s)ds$ denotes the *branching ratio* of the Hawkes process, and determines the expected number of events triggered by a single event. If $b < 1$ (i.e. each event triggers less than one event on average), the Hawkes process is called *stationary*, meaning that the total number of events remains bounded over time. In contrast, when $b > 1$, the Hawkes process becomes *non-stationary*, potentially leading to an infinite growth of the intensity in (2.42) and to an explosion of events.

When marks are available, the conditional intensities of the marked Hawkes process would take the form

$$\lambda_k^*(t) = \lambda_k + \sum_{k'=1}^{K} \sum_{(t_j,k_j)\in\mathcal{H}_t^{k'}} \psi_{k,k'}(t - t_j), \tag{2.42}$$

where now $\lambda_k \in \mathbb{R}_+$ and $\mathcal{H}_t^k = \{(t_j, k_j) \in \mathcal{S} \mid t_j < t, k_j = k\}$ refers to the set of events of mark k that occurred prior to t. Here $\lambda_k \in \mathbb{R}_+$ is a baseline intensity rate for each mark k, and $\psi_{k,k'}$ denotes a marked extension of the triggering kernel. For instance, the exponential kernel in (2.40) would now write

$$\psi_{k,k'}(t - t_{i-1}) = \gamma_{k,k'}\exp\left(-\beta_{k,k'}(t - t_{i-1})\right), \tag{2.43}$$

with

$$\boldsymbol{\gamma} = \begin{bmatrix} \gamma_{1,1} & \cdots & \gamma_{1,K} \\ \vdots & \ddots & \vdots \\ \gamma_{K,1} & \cdots & \gamma_{K,K} \end{bmatrix} \in \mathbb{R}_+^{K\times K}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_{1,1} & \cdots & \beta_{1,K} \\ \vdots & \ddots & \vdots \\ \beta_{K,1} & \cdots & \beta_{K,K} \end{bmatrix} \in \mathbb{R}_+^{K\times K}. \tag{2.44}$$

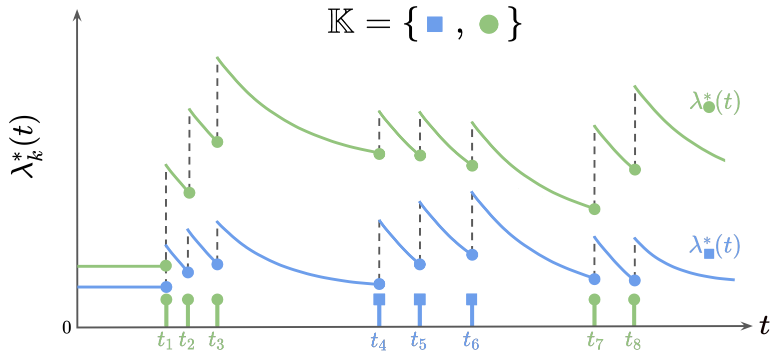


Figure 2.7: Illustration of a realization of a Hawkes process with $K = 2$ marks and evolution of the corresponding marked intensity functions $\lambda_k^*(t)$.

The parameters $\gamma_{k,k'}$ and $\beta_{k,k'}$ inherit the same interpretation as for the unmarked case, with the difference that they now capture the influence (i.e. excitation and decay) of an event of mark k on another event of mark k'. Figure 2.7 illustrates a typical realization of a Hawkes process with $K = 2$ marks. As observed, the Hawkes process results in clustered points patterns, where the occurrence of an event immediately increases the likelihood of observing a new one soon after.

On a final note, although both the exponential and power-law kernels enable to model self-exciting behaviors, there exists a key difference between the two. Indeed, a Hawkes process with exponential kernels verifies the Markov property (Oakes, 1975), meaning that the intensity $\lambda_k^*(t+\Delta t)$ at time $t+\Delta t$ only depends on the current intensity $\lambda_k^*(t)$, and a function of the elapsed time since the last event $t + \Delta t - t_{i-1}$. On the contrary, a Hawkes process with power-law kernels cannot be mapped in such a way, and hence lacks the Markov property (Bacry & Muzy, 2014).

Self-correcting process (Isham & Westcott, 1979) In a Hawkes process, the occurrence of an event increases the likelihood of subsequent events occurring shortly thereafter. What if we were instead interested in modeling the opposite behavior, i.e. observations that decrease the likelihood of new events rather than increase it? This desideratum can be precisely achieved by defining a self-correcting process, whose intensity function takes the form

$$\lambda^*(t) = \exp\left(\lambda t - \sum_{t_j \in \mathcal{H}_t} \gamma\right), \tag{2.45}$$

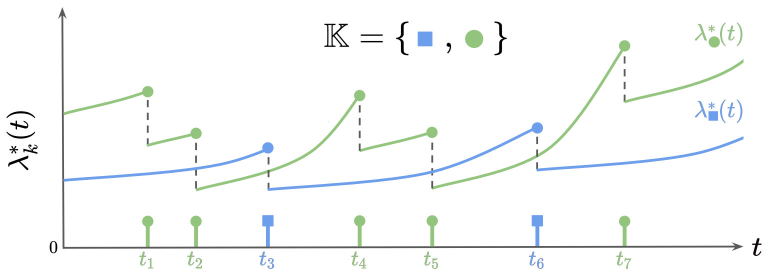


Figure 2.8: Illustration of a realization of a self-correcting process with $K = 2$ marks and evolution of the corresponding marked intensity functions $\lambda^*_k(t)$. For the sake of simplicity, we assume independence between events of different marks.

where $\lambda, \gamma > 0$. The name *self-correcting* relates to the process balancing two complementary effects that regulate event occurrences: in the absence of observations, the intensity rises exponentially over time, increasing the likelihood of new events. Then, when a new event materializes, the intensity is instantly dropped by the amount $\exp(-\gamma)$, ultimately decreasing their respective likelihood. The combinations of these two effects enable the process to *correct* itself and maintain a certain regularity.

In a marked scenario, the marked intensity functions would write

$$\lambda^*_k(t) = \exp\left(\lambda_k t - \sum_{k'=1}^{K} \sum_{(t_j,k_j)\in\mathcal{H}_t^{k'}} \gamma_{k'k}\right), \tag{2.46}$$

where $\lambda_k > 0$ and $\gamma_{k,k'} \in \mathbb{R}_+$. Drawing a parallel with the Hawkes process, the parameter $\gamma_{k,k'}$ now captures the *correcting* influence of an event of mark k' on the intensity of mark k. A realization of a self-correcting process with $K = 2$ marks is illustrated on Figure 2.8. As observed, the regulating behavior of a self-correcting process leads to regular spacing between events.

The self-correcting process has been used in situations where events tend to have an inhibiting influence on future realizations. For instance, the stress release of a large magnitude earthquake is expected to decrease the likelihood of observing a new occurrence soon after (Ogata & Vere-Jones, 1984), or the chances of criminals re-offending diminish once they have been arrested (Altieri et al., 2023).

Remark 2. (Superposition of independent processes) Suppose that we have access to two independent processes $N^1(t)$ and $N^2(t)$, whose intensity functions are $\lambda^{*,1}(t)$ and $\lambda^{*,2}(t)$, respectively. It turns out that the superposition of these two processes yields a new process $N(t) = N^1(t) + N^2(t)$ whose intensity function is given by the sum of the intensity of each individual process, i.e. $\lambda^*(t) = \lambda^{*,1}(t) + \lambda^{*,2}(t)$ (De et al., 2019). This implies that we can seamlessly design new models from the superposition of any of the processes defined previously, allowing to capture more complex dynamics. For instance, Schoenberg & Bolt (2000); Rotondi & Varini (2019) combined self-correcting and self-exciting processes to jointly model large-magnitude earthquakes and their aftershocks, while Xu et al. (2017) superposed Hawkes processes to solve the cold-start problem in recommendation systems.

2.1.3 Learning a MTPP Model from Event Sequence Data

Suppose that we observe a dataset of training sequences $\mathcal{S}_{\text{train}} = \{\mathcal{S}_1, ..., \mathcal{S}_L\}$ where each sequence $\mathcal{S}_l = \{(t_{i,l}, k_{i,l})\}_{i=1}^{n_l}$ comprises n_l events with arrival times observed within the interval $[0, T]$, and $l = 1, ..., L$. Equivalently, $\tau_{i,l} = t_{i,l} - t_{i-1,l}$ for $i = 2, ..., n_l$ and $\tau_{1,l} = t_{1,l}$. We assume that each of these sequences was generated independently from some ground-truth MTPP, e.g. a marked Hawkes process from Section 2.1.2. To enable inference on new sequences, we want to learn a model $\lambda_k^*(t; \boldsymbol{\theta})$, $\Lambda_k^*(t; \boldsymbol{\theta})$, $f^*(\tau, k; \boldsymbol{\theta})$ or $F^*(\tau; \boldsymbol{\theta})$ with learnable parameters $\boldsymbol{\theta} \in \Theta$ from $\mathcal{S}_{\text{train}}$, where Θ denotes the parameter space. For instance, if we were to parametrize a marked Hawkes process with exponential kernels in (2.42)-(2.43), then $\boldsymbol{\theta} = \{\boldsymbol{\lambda}, \boldsymbol{\gamma}, \boldsymbol{\beta}\}$ with $\boldsymbol{\lambda} \in \mathbb{R}_+^K$, $\boldsymbol{\gamma} \in \mathbb{R}_+^{K \times K}$ and $\boldsymbol{\beta} \in \mathbb{R}_+^{K \times K}$. As previously discussed, we have to keep in mind that the chosen parametrization must define a valid joint distribution over arrival times and marks—the properties enumerated in Section 2.1.1 must always be satisfied.

Given a valid parametrization of a MTPP model, we often use MLE to learn the parameters $\boldsymbol{\theta}$. For instance, consider a parametrization of $f^*(\tau, k; \boldsymbol{\theta})$. Given the training sequences $\mathcal{S}_{\text{train}}$, the *likelihood function* is given by

$$L(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = \prod_{l=1}^{L} \left[\prod_{i=1}^{n_l} f^*(\tau_{l,i}, k_{l,i}; \boldsymbol{\theta}) \right] (1 - F^*(T; \boldsymbol{\theta})). \tag{2.47}$$

where $F^*(T; \boldsymbol{\theta}) = \sum_{k=1}^{K} F^*(T, k; \boldsymbol{\theta})$ accounts for the absence of events in each interval $(t_{l,n}, T]$.

Taking the negative log of (2.47) finally yields the *Negative Log-Likelihood (NLL) objective function*:

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = -\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log f^*(\tau_{l,i}, k_{l,i}; \boldsymbol{\theta}) - \log(1 - F^*(T; \boldsymbol{\theta}))\right], \tag{2.48}$$

$$= -\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log f^*(\tau_{l,i}; \boldsymbol{\theta}) + \log p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta}) - \log(1 - F^*(T; \boldsymbol{\theta}))\right], \tag{2.49}$$

where we used the factorization in (2.4) between lines (2.48) and (2.49). This objective can be equivalently expressed by means of other functions characterizing the MTPP. If one chooses instead to parametrize $\lambda^*_k(t; \boldsymbol{\theta})$ or $\Lambda^*_k(t; \boldsymbol{\theta})$, then the NLL objective can be rewritten as

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = -\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\left[\log \lambda^*_{k_{l,i}}(t_{l,i}; \boldsymbol{\theta}) - \Lambda^*(t_{l,i}; \boldsymbol{\theta})\right] - \Lambda^*(T; \boldsymbol{\theta})\right] \tag{2.50}$$

$$= -\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\left[\log \lambda^*(t_{l,i}; \boldsymbol{\theta}) + \log p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta})) - \Lambda^*(t_{l,i}; \boldsymbol{\theta})\right] - \Lambda^*(T; \boldsymbol{\theta})\right], \tag{2.51}$$

where the absence of observations in the intervals $(t_{l,n}, T]$ is now accounted for by $\Lambda^*(T; \boldsymbol{\theta}) = \sum_{k=1}^{K}\Lambda^*_k(T; \boldsymbol{\theta})$. Finally, learning a MTPP model involves finding the set of parameters $\hat{\boldsymbol{\theta}}$ that minimizes the objectives in (2.49), (2.50) or (2.51), i.e.

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmin}}\ \mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}). \tag{2.52}$$

Unfortunately, solving the optimization problem in (2.52) is only achievable analytically for simple processes, such as the HPP. Hence, for more complex MTPP models, optimization must be carried out using numerical optimization methods, like mini-batch Stochastic Gradient Descent (SGD) (Ruder, 2017). Let $\mathcal{S}_{b,\text{train}} \subset \mathcal{S}_{\text{train}}$ denote a batch of b training sequences sampled uniformly at random from $\mathcal{S}_{\text{train}}$, and let $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{b,\text{train}})$ denote the gradient of $\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{b,train}})$, i.e. the vector of partial derivatives of $\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{b,train}})$ with respect to all parameters $\boldsymbol{\theta}$. Starting from an initial state $\boldsymbol{\theta}^{(0)}$, mini-batch SGD minimizes (2.52) by iteratively updating the parameters $\boldsymbol{\theta}$ in the negative direction of the gradient, i.e.

$$\boldsymbol{\theta}^{(s+1)} = \boldsymbol{\theta}^{(s)} - \eta \, \nabla_{\boldsymbol{\theta}^{(s)}} \mathcal{L}\left(\boldsymbol{\theta}^{(s)}; \mathcal{S}_{b,\text{train}}\right), \tag{2.53}$$

where $s \in \mathbb{N}$ and $\eta > 0$ is called the *learning rate*, which controls the magnitude of the update step taken. To solve the optimization problem in (2.52), the update step in (2.53) is repeated multiple times for $\mathcal{S}_{b,\text{train}}$ sampled from $\mathcal{S}_{\text{train}}$ until a minima of the objective function is reached. Provided that η is sufficiently small (e.g. 10^{-3}), mini-batch SGD is guaranteed to converge to at least a local minima of $\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})$[1]. In practice, extensions of mini-batch SGD that promote more stable and faster convergence are frequently used, such as Adam (Kingma & Ba, 2014), Adagrad (Duchi et al., 2011), Adadelta (Zeiler, 2012), or RMSprop (Hinton, 2018).

Once trained, the MTPP model $f(t, k; \hat{\boldsymbol{\theta}})/F^*(t, k; \hat{\boldsymbol{\theta}})/ \ \lambda_k^*(t; \hat{\boldsymbol{\theta}})/\Lambda_k^*(t; \hat{\boldsymbol{\theta}})$ with learned parameters $\hat{\boldsymbol{\theta}}$ can be used for prediction tasks on new sequences, addressing queries such as “When is the next event likely to occur?”, “What will be the type of the next event, given that it occurs at a certain time t?” or “How long until an event of type k occurs?” We will discuss such prediction tasks in more details in Section 2.1.5.

Alternatives to MLE. The NLL in (2.49), (2.50) or (2.51) are based on the logarithmic scoring rule (LogScore), and have been largely adopted in the literature as the default objective for learning MTPP models (Shchur et al., 2021b). However, Brehmer et al. (2021) showed that alternatives to the LogScore can be used to derive a variety of consistent scoring rules for MTPP models. Let S^f, S^p and S^F be (strictly) consistent scoring rules for $f^*(\tau; \boldsymbol{\theta})$, $p^*(k|\tau; \boldsymbol{\theta})$ and $1 - F^*(T; \boldsymbol{\theta})$, respectively. Given a a set of training sequences $\mathcal{S}_{\text{train}}$, the scoring rule

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = -\frac{1}{L}\sum_{l=1}^{L}\Bigg[\sum_{i=1}^{n_l}\Big[S_{l,i}^f(f^*(\tau_{l,i}; \boldsymbol{\theta})) + S_{l,i}^p(p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta}))\Big] \\ - S_l^F\left(1 - F^*(T; \boldsymbol{\theta})\right)\Bigg],\end{aligned} \tag{2.54}$$

is (strictly) consistent for the joint PDF $f^*(\tau, k; \boldsymbol{\theta})$ restricted to the interval $[0, T]$ (Brehmer et al., 2021). This expression implies that one can choose

[1]The local minima becomes a global minima in case of a convex problem.

among a broader class of scoring rules than the LogScore to evaluate point process forecasts. For instance, one can choose to use the Continuous Ranked Probability Score (CRPS) (Gneiting et al., 2007) for S^f, or the Brier score (Brier, 1950) for S^p. In this context, Ben Taieb (2022), proposed to use the CRPS to learn the quantile function of inter-arrival times. In contrast to the *local* property of the LogScore, both the CRPS and the Brier score are *sensitive to distance*, in the sense that they reward predictive distributions that assign probability mass close to the observed realization (Gebetsbergera et al., 2018). Nonetheless, the choice between local and non-local proper scoring rules has been generally subjective in the literature. Finally, note that using the LogScore for all $S^f_{l,i}$, $S^p_{l,i}$ and $S^F_{l,i}$ in (2.54) reduces to the NLL in (2.49).

Another popular class of learning procedures for MTPP models falls under the category of Least-Square Estimation (LSE) (Wang et al., 2016; Eichler et al., 2017; Xu et al., 2017). Based on the observation that

$$\mathbb{E}[N_k(t)|\mathcal{H}_t] = \int_0^t \lambda_k^*(s)ds, \tag{2.55}$$

least square objective functions essentially aim to minimize the squared error between the observed counting process and the integral of the intensity function , i.e.

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{S}_{\text{train}}) = -\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_l}\sum_{k=1}^{K}\left[N_{k_{l,i}}(t_{l,i}) - \int_0^{t_{l,i}} \lambda^*_{k_{l,i}}(s)ds\right]^2, \tag{2.56}$$

Finally, other forms of training procedures have been considered to learn the parameters of the model, such as reinforcement learning (Upadhyay et al., 2018; Li et al., 2018), adversarial training (Xiao et al., 2018; Zhu et al., 2020), variational inference (Boyd et al., 2020; Chen et al., 2021a), noise contrasting estimation (Guo et al., 2018b; Mei et al., 2020), or score-matching objectives (Sahani et al., 2016; Li et al., 2023).

Remark 3. Instead of a single mark, assume that events can be associated to multiple marks simultaneously. Such a scenario can naturally occur in various applications, such as in the context of Electronic Health Records (EHR), where a single consultation often leads to multiple diagnoses and treatments being recorded simultaneously (Enguehard et al., 2020). In this multi-mark setting, the NLL writes as a form of binary cross-entropy loss (Bhave & Perotte, 2021) that accounts for the multiple marks that events can take simultaneously, i.e.

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \mathcal{S}_{\text{train}}) = &-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\sum_{k=1}^{K}\Big[\mathbb{1}_{l,i,k}\log f^*(\tau_{l,i}, k_{l,i}; \boldsymbol{\theta})\right.\\ &+ (1-\mathbb{1}_{l,i,k})\log\left(f^*(\tau_{l,i};\boldsymbol{\theta}) - f^*(\tau_{l,i}, k_{l,i};\boldsymbol{\theta})\right)\Big]\\ &\left.- \log(1 - F^*(T;\boldsymbol{\theta}))\right], \end{aligned} \tag{2.57}$$

where $\mathbb{1}_{l,i,k}(\cdot)$ is a indicator function taking the value 1 if the event $\boldsymbol{e}_{l,i}$ is associated to mark k, along possibly other concurrent marks in $\mathbb{K}$. However, in this dissertation, we will exclusively focus on the single-mark setting, in which events can exclusively take *one mark* at any time. This implies that the NLL objective for a dataset of L sequences will be systematically given by (2.49) or (2.51).

2.1.4 Sampling from a MTPP

Sampling from a MTPP can be useful for a wide range of practical applications. For instance, by simulating multiple events from a model, we can observe how the process typically evolves, enabling us to assess some of its core characteristics, e.g. does the process exhibit a clustered or a regular point pattern? In other scenarios, we may instead seek to predict the future evolution of a process by simulating possible trajectories from partially-observed sequences.

Once a MTPP model has been fitted to the observed data, two main approaches are commonly used in practice to generate new event sequences within a predefined interval $[0, T]$, namely the *inverse transform method* and *Ogata's modified thinning algorithm* (Ogata, 1981). Both of these methods are presented below.

Inverse transform method via CDF. Before presenting how an entire sequence can be generated using the inverse transform method, it is essential to understand how a single event can be sampled from the inverse CDF conditional on past observations. Suppose that we observe the events $\{(t_1, k_1), ..., (t_{i-1}, k_{i-1})\}$, and that we wished to sample the next event (t_i, k_i). Given the predictive CDF of arrival times $F^*(t; \hat{\boldsymbol{\theta}}) = F(t|(t_1, k_1), ..., (t_{i-1}, k_{i-1}); \hat{\boldsymbol{\theta}})$, and the predictive PMF of marks $p^*(k|t; \hat{\boldsymbol{\theta}}) = p(k|t, (t_1, k_1), ..., (t_{i-1}, k_{i-1}); \hat{\boldsymbol{\theta}})$, sampling the next event can be achieved in two consecutive steps (Ross, 2012). First, we sample

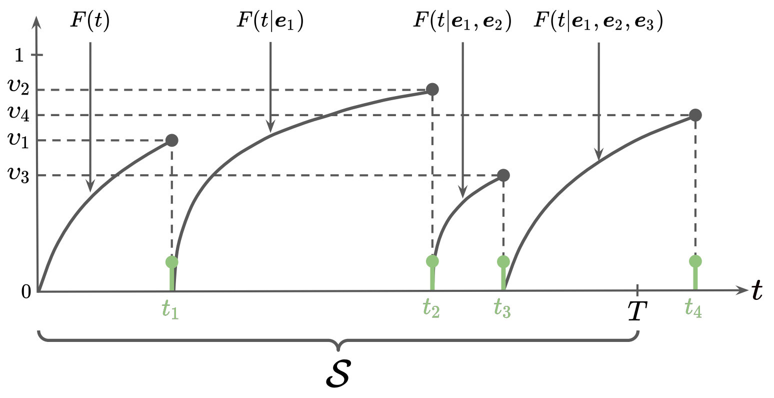


Figure 2.9: Illustration of the inverse transform method via CDF for an unmarked scenario (i.e. $\upsilon_{1,i} = \upsilon_i$). As t_4 falls outside the observation window T, its is not included in the sequence $\mathcal{S}$.

$\upsilon_{1,i} \sim$ Uniform$[0, 1]$ and compute

$$t_i = (F^*)^{-1}(\upsilon_{1,i}; \hat{\boldsymbol{\theta}}), \tag{2.58}$$

where $(F^*)^{-1}$ denote the inverse CDF transform of F^*. Then, to sample k_i conditional on t_i, we first sample $\upsilon_{2,i} \sim$ Uniform$[0, 1]$ and solve

$$k_i = \underset{\pi_k}{\operatorname{argmin}} \sum_{k=1}^{K} p^*(k|t_i; \hat{\boldsymbol{\theta}})_{\pi_k} < \upsilon_{2,i}, \tag{2.59}$$

where π_k is a permutation of the indices $k \in \mathbb{K}$ that sorts the entries of $\boldsymbol{p}^*(\hat{\boldsymbol{\theta}}) = \left[p(k = 1|t_i; \hat{\boldsymbol{\theta}}), ..., p(k = K|t_i; \hat{\boldsymbol{\theta}})\right]^\intercal \in \mathbb{R}_+^K$ from less to most likely. Consequently, to generate a whole new sequence, we can start from $t = 0$ and sample new events one by one from $F^*(\tau; \hat{\boldsymbol{\theta}})$ and $p^*(k|\tau; \hat{\boldsymbol{\theta}})$, conditioning on the sampled events as we go in an auto-regressive fashion, until we reach the end of the observation window T. This procedure is outlined in Algorithm 1, and illustrated on Figure 2.9. However, the inverse transform via CDF method requires us to evaluate $(F^*)^{-1}(\upsilon_{1,i}; \hat{\boldsymbol{\theta}})$, which is not always available in closed-form for some models like the Hawkes process. Fortunately, in such cases, we can rely on efficient numerical root-finding algorithms to compute the inverse $(F^*)^{-1}$.

Algorithm 1 Inverse Transform Sampling via CDF

```
**Input:** Predictive CDF of arrival times $F^*(t;\hat{\boldsymbol{\theta}})$, predictive PMF of marks $p^*(k|t;\hat{\boldsymbol{\theta}})$, end of observation window $T$.
Set $t = 0$, $i = 1$.
**while** $t \leq T$ **do**
    1. Sample $\upsilon_{1,i}, \upsilon_{2,i} \sim \text{Uniform}[0,1]$
    2. Compute the next arrival time $t_i = (F^*)^{-1}(\upsilon_{1,i};\hat{\boldsymbol{\theta}})$.
    3. Compute the next mark $k_i = \underset{\pi_k}{\text{argmin}} \sum_{k=1}^{K} p^*(k|t_i;\hat{\boldsymbol{\theta}})_{\pi_k} < \upsilon_{2,i}$.
    4. Keep the event $(t_i, k_i)$ if $t_i \leq T$, else reject it.
    5. Set $t = t_i$, $i = i + 1$.
**end while**
**Return:** A realization $\mathcal{S} = \{(t_1,k_1),...,(t_n,k_n)\}$ of a MTPP with intensity $\lambda_k^*(t;\hat{\boldsymbol{\theta}})$.
```

Inverse transform method via compensator. Instead of using the predictive CDF of inter-arrival times, an alternative implementation of the inverse transform method relies on the *ground compensator* of the process defined as

$$\Lambda^*(t) = \int_0^t \lambda^*(s)ds, \tag{2.60}$$

to generate new event sequences. This procedure takes its foundation on the *random time change theorem*, which is formally stated below

Theorem 1 (Random Time Change Theorem (Brown et al., 2002))**.** *Let $\mathcal{S} = \{(t_1,k_1),...,(t_n,k_n)\}$ be a realization of a MTPP on $[0,T]$ with ground intensity $\lambda^*(t)$ and ground compensator $\Lambda^*(t)$. Then the sequence $\mathcal{S}_\Lambda = \{\Lambda^*(t_1),...,\Lambda^*(t_n)\}$ follows a univariate HPP with unit rate on the interval $[0,\Lambda^*(T)]$.*

Essentially, the random time change theorem states that a sequence of arrival times generated on $[0,T]$ from any arbitrary process with ground compensator $\Lambda^*(t)$ can be transformed into a realization of a unit rate HPP on $[0,\Lambda^*(T)]$. This in turn leads to Corollary 1.

Algorithm 2 Inverse Transform Sampling via Random Time Change Theorem

```
Input: Predictive ground compensator $\Lambda^*(t;\hat{\boldsymbol{\theta}})$, predictive PMF of marks
$p^*(k|t;\hat{\boldsymbol{\theta}})$, end of observation window $T$.
Set $t = 0$, $z = 0$, $i = 1$.
while $t \leq T$ do
    1. Sample $\upsilon_{1,i}, \upsilon_{2,i} \sim \text{Uniform}[0,1]$
    2. Compute $\nu_i = -\text{log }(1 - \upsilon_{1,i})$
    3. Set the next HPP arrival time as $z_i = z + \nu_i$
    4. Compute the next arrival time as $t_i = (\Lambda^*)^{-1}(z_i;\hat{\boldsymbol{\theta}})$
    5. Compute the next mark $k_i = \underset{\pi_k}{\text{argmin}} \sum_{k=1}^{K} p^*(k|t_i;\hat{\boldsymbol{\theta}})_{\pi_k} < \upsilon_{2,i}$.
    6. If $t_i \leq T$, keep the event $(t_i, k_i)$, else reject it.
    4. Set $t = t_i$, $i = i + 1$.
end while
Return: A realization $\mathcal{S} = \{(t_1, k_1), ..., (t_n, k_n)\}$ of a MTPP with intensity
$\lambda_k^*(t;\hat{\boldsymbol{\theta}})$.
```

Corollary 1 (Inverse Random Time Change Theorem (Rasmussen, 2018))**.** *Let $\mathcal{S}_Z = \{z_1, ..., z_n\}$ be a realization of a univariate unit rate HPP on $[0, T_Z]$, and let $\Lambda^*(t)$ be the ground compensator of an arbitrary MTPP. Assuming the existence of the inverse transform $(\Lambda^*)^{-1}$ of Λ^*, the sequence*

$$\mathcal{S} = \{(\Lambda^*)^{-1}(z_1), ..., (\Lambda^*)^{-1}(z_n)\} \tag{2.61}$$

is a valid realization of the ground MTPP with compensator $\Lambda^*(t)$ on $[0, T]$ where $T = (\Lambda^*)^{-1}(T_Z)$.

This means that simulating a new sequence from any arbitrary MTPP model with ground compensator $\Lambda^*(t;\hat{\boldsymbol{\theta}})$ can be achieved in two consecutive steps. First, we need to generate a sample z_i from a unit rate HPP. Recall from Section 2.1.2 that the inter-arrival times of a unit rate HPP are i.i.d. random variables following an exponential distribution with rate $\lambda = 1$. Consequently, to obtain z_i, we first sample $\upsilon_{1,i} \sim \text{Uniform}[0,1]$ and compute $z_i = z_{i-1} + s_i$, where the inter-arrival time s_i is obtained from the inverse CDF of an exponential distribution with rate $\lambda = 1$, i.e.

$$\nu_i = -\text{log }(1 - \upsilon_{1,i}). \tag{2.62}$$

Finally, to obtain the actual event (t_i, k_i) from the MTPP of interest, we evaluate $t_i = (\Lambda^*)^{-1}(z_i; \hat{\boldsymbol{\theta}})$, and sample the mark k_i from (2.59). Starting from $t = 0$, we repeat this procedure until we reach the end of the observation window T, conditioning $\Lambda^*(t; \hat{\boldsymbol{\theta}})$ and $p^*(k|t; \hat{\boldsymbol{\theta}})$ on the newly generated events as we go. This sampling procedure is summarized in Algorithm (2).

Similarly to $(F^*)^{-1}$, the inverse of the ground compensator $(\Lambda^*)^{-1}$ is not always available in closed-form, requiring numerical root-finding techniques. We will see below how this pitfall can be avoided when using thinning algorithms to sample from MTPP models.

Ogata's modified thinning algorithm. The core idea behind Ogata's modified thinning algorithm (Ogata, 1981) lies in two main steps: (1) Simulate events from a HPP with a greater intensity than the MTPP of interest, and (2) thin out the excess events to account for oversampling. To this end, we must first define an upper bound $m(t)$ on the ground intensity of the process $\lambda^*(t; \hat{\boldsymbol{\theta}})$ that verifies

$$m(t) \geq \sup_{s \in [t,T]} \lambda^*(s; \hat{\boldsymbol{\theta}}). \tag{2.63}$$

For instance, if we want to simulate from a Hawkes process with exponential kernels in (2.42)-(2.43), we can leverage the fact the intensity decreases exponentially between two events and define $m(t)$ as

$$m(t) = \sum_{k=1}^{K} \left[\mu_k + \sum_{k'=1}^{K} \sum_{(t_j,k_j) \in \mathcal{H}_t^{k'}} \gamma_{k,k'} \exp\left(-\beta_{k,k'}(t - t_j)\right) \right], \tag{2.64}$$

To generate a candidate event (t_i, k_i), we first sample an inter-arrival time τ_i from a HPP with rate $m(t)$, and keep the candidate arrival-time $t_i = t_{i-1} + \tau_i$ with probability $\frac{\lambda^*(t_i; \hat{\boldsymbol{\theta}})}{m(t)}$. This ensures that the non-thinned out arrival times have been generated from a MTPP process with ground intensity $\lambda^*(t; \hat{\boldsymbol{\theta}})$ (see Theorem 4.2 of (Chen, 2018) for a detailed proof). If t_i is accepted, we sample the mark k_i from (2.59), and repeat the procedure until we reach the end of the observation window T. Ogata's thinning method is summarized in Algorithm 3.

Although Ogata's modified thinning algorithms doesn't require the computation of the inverse CDF or compensator, its implementation still faces some challenges, as discussed in Shchur (2022). For instance, evaluating the upper bound $m(t)$ may be a daunting task for some parametrizations of MTPP models, and the algorithm can suffer from high rejection rates, making it less efficient than the inverse transform methods.

Algorithm 3 Ogata's Modified Thinning Algorithm

```
**Input:** Predictive ground intensity $\lambda^*(t;\hat{\boldsymbol{\theta}})$, predictive PMF of marks $p^*(k|t;\hat{\boldsymbol{\theta}})$, bound $m^*(t)$, end of observation window $T$.
Set $t = 0$, $i = 1$.
**while** $t \leq T$ **do**
    2. Sample $\upsilon_{1,i}, \upsilon_{2,i}, \upsilon_{3,i} \sim \text{Uniform}[0,1]$
    3. Compute the upper bound $m(t)$.
    4. Compute $\tau_i = -\frac{1}{m(t)}\log\,(1-\upsilon_{1,i})$.
    5. Set the next candidate arrival time as $t_i = t + \tau_i$.
    6. Compute the next mark $k_i = \underset{\pi_k}{\text{argmin}} \sum_{k=1}^{K} p^*(k|t_i;\hat{\boldsymbol{\theta}})_{\pi_k} < \upsilon_{2,i}$.
    7. If $t_i \leq T$ and $\upsilon_{3,i} < \frac{\lambda^*(t_i)}{m(t)}$, keep the event $(t_i, k_i)$ and set $i = i+1$, else reject it.
    8. Set $t = t_i$.
**end while**
**Return:** A realization $\mathcal{S} = \{(t_1,k_1),...,(t_n,k_n)\}$ of a MTPP with intensity $\lambda_k^*(t;\hat{\boldsymbol{\theta}})$.
```

2.1.5 Prediction Tasks and Evaluation Metrics

Recall from Section 2.1.3 that the most common approach to learn a MTPP model $f^*(\tau,k;\boldsymbol{\theta})$ (or equivalently, $\lambda_k^*(t;\boldsymbol{\theta})$, $\Lambda_k^*(t;\boldsymbol{\theta})$ or $F^*(t,k;\boldsymbol{\theta})$) involves minimizing the NLL objective in (2.49). Rearranging the terms of this expression, we obtain

$$\mathcal{L}(\boldsymbol{\theta};\mathcal{S}_{\text{train}}) = \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log\, f^*(\tau_{l,i};\boldsymbol{\theta}) - \log(1-F^*(T;\boldsymbol{\theta}))\right]}_{\mathcal{L}_T(\boldsymbol{\theta};\mathcal{S}_{\text{train}})} \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log\, p^*(k_{l,i}|\tau_{l,i};\boldsymbol{\theta})\right]}_{\mathcal{L}_M(\boldsymbol{\theta};\mathcal{S}_{\text{train}})} \tag{2.65}$$

This reveals that learning a MTPP model from event sequences essentially amounts to solving two main estimation tasks: estimating a predictive PDF of inter-arrival times $f^*(\tau;\boldsymbol{\theta})$ through $\mathcal{L}_T(\boldsymbol{\theta};\mathcal{S}_{\text{train}})$, and a predictive PMF of marks $p^*(k|\tau;\boldsymbol{\theta})$ through $\mathcal{L}_M(\boldsymbol{\theta};\mathcal{S}_{\text{train}})$. Note that a similar decomposition involving $\lambda^*(t;\boldsymbol{\theta})$, $\Lambda^*(t;\boldsymbol{\theta})$, and $p^*(k|\tau;\boldsymbol{\theta})$ can be derived from expression (2.51). Once the model is trained, we can use the estimates $f^*(\tau;\hat{\boldsymbol{\theta}})$ and $p^*(k|\tau;\hat{\boldsymbol{\theta}})$ to compute point predictions for new events, answering queries of the type:

- *When is the next event likely to occur?*
- *What will be the type of the next event, given that it occurs at a certain time t?*
- *How long until an event of type k occurs?*

In the following, we will denote the task of estimating $f^*(\tau;\boldsymbol{\theta})$ and generating point estimates for the (inter-)arrival times of future events as the *time prediction task*. Similarly, the *mark prediction tasks* will refer to the task of estimating $p^*(k|\tau;\boldsymbol{\theta})$ and generating point estimates for the marks of future events. These two tasks are briefly discussed below, along with some commonly used metrics for evaluating model performance with respect to each of them.

Time prediction task. Given a set of new test sequences $\mathcal{S}_{\text{test}}$ generated from the same underlying process as $\mathcal{S}_{\text{train}}$, we can assess the goodness-of-fit of $f^*(\tau;\hat{\boldsymbol{\theta}})$ by reporting the NLL score $\mathcal{L}_T(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}})$[2] in (2.65). In a broad sense, $\mathcal{L}_T(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}})$ provides us with a measure of how likely the inter-arrival times in $\mathcal{S}_{\text{test}}$ have been generated by the model $f^*(\tau;\hat{\boldsymbol{\theta}})$. However, as $\mathcal{L}_T(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}})$ is unbounded and can be arbitrarily shifted by changing the scale of the data, it is challenging to interpret (Shchur et al., 2020a, 2021b). Nonetheless, this metric can be used as basis for comparison between two models, with the one showing the lowest $\mathcal{L}_T(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}})$ score presenting itself as the best fit to the observed arrival times.

Conversely, in applications where point predictions are of interest, an estimate of the next inter-arrival time $\tilde{\tau}_i$ can be obtained by deriving diverse summary statistics from $f^*(\tau;\hat{\boldsymbol{\theta}})$. For instance, assuming that (t_{i-1},k_{i-1}) is the last observed event, one can choose $\tilde{\tau}_i$ as the mean computed from $f^*(\tau;\hat{\boldsymbol{\theta}})$, i.e.

$$\tilde{\tau}_i = \mathbb{E}_{\tau\sim f^*(\tau;\boldsymbol{\theta})}[\tau] = \int_0^\infty \tau f^*(\tau;\hat{\boldsymbol{\theta}})d\tau, \tag{2.66}$$

where the expectation can be estimated via simulation using the inverse transform method of Section 2.1.4, or via Monte-Carlo integration (Kalos & Whit-

[2]Note that this term is computed on the new test sequences $\mathcal{S}_{\text{test}}$ rather than on the training sequences $\mathcal{S}_{\text{train}}$

lock, 2008). Alternatively, $\tilde{\tau}_i$ can be chosen as the median computed from $f^*(\tau;\hat{\boldsymbol{\theta}})$, i.e.

$$\tilde{\tau}_i = (F^*)^{-1}(0.5;\hat{\boldsymbol{\theta}}), \tag{2.67}$$

where $(F^*)^{-1}$ can also be evaluated using the inverse transform method. Finally, we can assess the quality of the estimates $\tilde{\tau}_i$ by quantifying their average deviation from the true observations in $\mathcal{S}_{\text{test}}$, e.g. by using the *Mean Absolute Error (MAE)* or *Mean Squared Error (MSE)* in event time prediction:

$$\text{MAE} = \frac{1}{|\mathcal{S}_{\text{test}}|}\sum_{l=1}^{|\mathcal{S}_{\text{test}}|}\sum_{i=1}^{n_l}\frac{|\tilde{\tau}_{l,i}-\tau_{l,i}|}{n_l}, \tag{2.68}$$

$$\text{MSE} = \frac{1}{|\mathcal{S}_{\text{test}}|}\sum_{l=1}^{|\mathcal{S}_{\text{test}}|}\sum_{i=1}^{n_l}\frac{(\tilde{\tau}_{l,i}-\tau_{l,i})^2}{n_l}. \tag{2.69}$$

Note that the MAE is a consistent scoring function for a forecast of the median in (2.67), while the MSE is a consistent scoring function for a forecast of the mean in (2.66) (Gneiting, 2012).

Mark prediction task. Similarly to the time prediction task, the goodness-of-fit of $p^*(k|\tau;\hat{\boldsymbol{\theta}})$ can be evaluated by computing $\mathcal{L}_M(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}})$ on new test sequences $\mathcal{S}_{\text{test}}$. However, $\mathcal{L}_M(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}})$ inherits the same interpretation challenge as its time counterpart. Fortunately, if we assume again that (t_{i-1},k_{i-1}) is the last observed event in a given sequence, we can generate a point prediction $\tilde{k}_i$ for the next mark as

$$\tilde{k}_i = \underset{k\in\mathbb{K}}{\operatorname{argmax}}\ p^*(k|\tilde{t}_i;\hat{\boldsymbol{\theta}}) = \underset{k\in\mathbb{K}}{\operatorname{argmax}}\ \frac{\lambda_k^*(\tilde{t}_i;\hat{\boldsymbol{\theta}})}{\lambda^*(\tilde{t}_i;\hat{\boldsymbol{\theta}})} = \underset{k\in\mathbb{K}}{\operatorname{argmax}}\ \lambda_k^*(\tilde{t}_i;\hat{\boldsymbol{\theta}}), \tag{2.70}$$

and evaluate the quality of the estimate by means of various classification evaluation metrics. For instance, one could report the *Accuracy (ACC)* in event mark prediction:

$$\text{ACC} = \sum_{l=1}^{|\mathcal{S}_{\text{test}}|}\sum_{i=1}^{n_l}\frac{\mathbb{1}\left(\tilde{k}_{i,l}=k_{i,l}\right)}{n_l\times|\mathcal{S}_{\text{test}}|}, \tag{2.71}$$

where $\mathbb{1}(\cdot)$ is the indicator function, and $k_{i,l}$ is the ground-truth mark. Other choices of classification metrics include the average *F1-score* (Manning et al., 2008):

$$\text{F1-score} = \frac{1}{K}\sum_{k'=1}^{K}2\times\frac{\text{P}_{k'}\times\text{R}_{k'}}{\text{P}_{k'}+\text{R}_{k'}}, \tag{2.72}$$

where, noting $S = |\mathcal{S}_{\text{test}}|$,

$$\mathrm{P}_{k'} = \frac{1}{S}\sum_{l=1}^{S}\frac{1}{n_l}\sum_{i=1}^{n_l}\frac{\mathbb{1}\left(\tilde{k}_{i,l} = k', k_{i,l} = k'\right)}{\left[\mathbb{1}\left(\tilde{k}_{i,l} = k', k_{i,l} = k'\right) + \mathbb{1}\left(\tilde{k}_{i,l} = k', k_{i,l} \neq k'\right)\right]}, \tag{2.73}$$

$$\mathrm{R}_{k'} = \frac{1}{S}\sum_{l=1}^{S}\frac{1}{n_l}\sum_{i=1}^{n_l}\frac{\mathbb{1}\left(\tilde{k}_{i,l} = k', k_{i,l} = k'\right)}{\left[\mathbb{1}\left(\tilde{k}_{i,l} = k', k_{i,l} = k'\right) + \mathbb{1}\left(\tilde{k}_{i,l} \neq k', k_{i,l} = k'\right)\right]}. \tag{2.74}$$

We can also compute the *Mean Reciprocal Rank (MRR)* (Craswell, 2009):

$$\mathrm{MRR} = \frac{1}{S}\sum_{l=1}^{S}\frac{1}{n_l}\sum_{i=1}^{n_l}\frac{1}{\mathrm{Rank}(k_{l,i})}, \tag{2.75}$$

where $\mathrm{Rank}(k_{l,i})$ refers to the rank position of $k_{l,i}$ among the entries of $\boldsymbol{p}^*(\hat{\boldsymbol{\theta}}) = \left[p(k=1|t_i;\hat{\boldsymbol{\theta}}), ..., p(k=K|t_i;\hat{\boldsymbol{\theta}})\right]^\mathsf{T} \in \mathbb{R}_+^K$ sorted from highest to lowest probability, i.e. $\mathrm{Rank}(k_{l,i}) = 1$ if $k_{l,i}$ corresponds to the highest entry of $\boldsymbol{p}^*(\hat{\boldsymbol{\theta}})$, $\mathrm{Rank}(k_{l,i}) = 2$ if $k_{l,i}$ corresponds to its second highest entry, and so on. In practice, a model demonstrating higher accuracy, F1-score, and MRR in mark prediction is desired.

Remark 4. Despite offering clearer interpretation into model performance, point prediction metrics, such as MAE, accuracy or F1-score, are deemed less appropriate to evaluate MTPP models than their NLL counterparts $\mathcal{L}_T(\hat{\boldsymbol{\theta}}; \mathcal{S}_{\text{test}})$ and $\mathcal{L}_M(\hat{\boldsymbol{\theta}}; \mathcal{S}_{\text{test}})$ (Shchur et al., 2021b). The rationale behind this criticism is grounded on the fact that point prediction metrics only take a single prediction into account, whereas NLL metrics enable the evaluation of the entire predictive distributions $f^*(\tau;\hat{\boldsymbol{\theta}})$ and $p^*(k|\tau;\hat{\boldsymbol{\theta}})$. In Section 2.2.1, we discuss alternative metrics inspired from the forecasting literature that allow us to assess the quality of the predictive distributions while providing clearer interpretation than the NLL.

2.1.6 Link to Survival Analysis

Temporal Point Processes bear close relations to the field of *Survival Analysis (SAN)* (Kalbfleisch & Prentice, 2002), where the goal is to estimate the time to occurrence of an event of interest, such as death or recovery of a patient, component failure, or membership cancellation. The time to event $t^\star$, called *survival time*, is typically modeled using a *survival function* $S(t)$, that specifies the probability that the event did not occur until time t, i.e.

$$S(t) = 1 - F(t) = \mathbb{P}\left(t^\star > t\right), \tag{2.76}$$

where $F(t)$ is the CDF of survival times. These survival times can also be characterized using the *hazard function*, which gives the instantaneous risk of the event occurring at time t, i.e.

$$\lambda(t) = \frac{f(t)}{S(t)} = \lim_{\Delta t \to 0} \frac{\mathbb{P}\left(t \leq t^\star \leq t + \Delta t | t^\star > t\right)}{\Delta t}, \tag{2.77}$$

with $f(t)$ the PDF of survival times. Looking at the similarity between the hazard function in (2.77) and the conditional intensity function in (2.8), we can easily draw a parallel between SAN and TPP. However, while SAN and TPP are both used to model events that occur over time, there exists some key differences between the two fields. First, SAN usually focuses on a single event occurrence for the system under study (e.g. the death of a patient), whereas TPP involve modeling multiple events that materialize over time. Consequently, as only the first time to event is usually of interest, SAN models do not leverage past observations for future inference. In contrast, TPP usually account for the entire sequence of past observations, and appropriately capturing the complex inter-dependencies between event occurrences is often crucial to enable accurate predictions. Instead, SAN typically relies on external explanatory variables to characterize the survival or hazards functions, such as in the Cox proportional hazards regression model (Cox, 1972). While explanatory variables can also be included in a TPP model, the history of the process is usually considered as the main covariate. Finally, a central concept in SAN is *censoring*, which describes cases where the main event has not been observed within the specified observation window. Censoring is well-studied in SAN, and many approaches have been developed to accommodate missing observations during training and inference. Despite the availability of several methods designed to handle censored observations within event sequences (Mei et al., 2019; Shchur et al., 2020a; Gupta et al., 2021; Boyd et al., 2023), the problem generally received less attention in the context of MTPP modeling. For further resources on survival analysis, see for instance Clark et al. (2003); Klein et al. (2020); Wiegrebe et al. (2024).

2.2. Uncertainty Quantification

In this section, we slightly deviate from our context of MTPP modeling to introduce some key concepts of uncertainty quantification (UQ) that lay at the center of core contributions in this thesis. In the context of predictive modeling, UQ can be broadly defined as the process of identifying, modeling, and measuring the uncertainty in the predictions made by a model (Abdar et al., 2021). Essentially, uncertainty estimates must reflect the degree of confidence of a model with respect to its own predictions, allowing for appropriate caution when the predicted outcome is highly uncertain. This ability is crucial for safe decision-making in critical applications such as medical diagnosis or autonomous driving, where incorrect predictions can lead to disastrous consequences.

Conventionally, the taxonomy of uncertainties divides them into two main types based on the originating sources: *aleatoric* and *epistemic* uncertainty (Hüllermeier & Waegeman, 2021). Aleatoric uncertainty, also called data uncertainty, usually refers to the natural randomness and noise present in the observations. This property is inherent to the underlying data generating process itself, making it an irreducible source of uncertainty. On the other hand, epistemic uncertainty, or model uncertainty, refers to the incomplete knowledge of the fitted model, either stemming from incorrect modeling assumptions or inadequate training. Contrary to aleatoric uncertainty, epistemic uncertainty can be reduced by defining a more appropriate hypothesis set for the model, or by gathering more data. Nonetheless, sources of uncertainty are often misunderstood, and clearly distinguishing between aleatoric and epistemic uncertainty is not always feasible in practice (Gruber et al., 2023).

There has been growing interest in providing direct estimates of these sources of uncertainty in machine learning algorithms, leading to the development of numerous approaches and applications (Abdar et al., 2021; Gawlikowski et al., 2023). Among others, common examples include Bayesian inference (Blundell et al., 2015; Gal & Ghahramani, 2016) or deep ensemble methods (Lakshminarayanan et al., 2017; Wen et al., 2020).

In practice however, we also want these uncertainty estimates to be *reliable*, in the sense that they must faithfully reflect on the true uncertainty in the predictions. While we already introduced proper scoring rules for the evaluation of probabilistic forecasts in Sections 2.1.3 and 2.1.5, we continue our discussion by introducing two related topics. The first, *probabilistic calibration*, is a property referring to the statistical consistency between the observations and the predictive distributions returned by the model. The second, *conformal prediction*, is a statistically sound framework that enables the construction of distribution-free prediction regions from any arbitrary base model with finite-sample coverage guarantees.

2.2.1 Calibration

Calibration, in the context of forecasting theory, refers to the statistical consistency between the predicted distribution and the observed outcomes (Gneiting et al., 2007). It is a property expressed by any ideal model, and thus, any competent forecaster should strive to attain it. In the following, we will present how calibration is defined in the general context of regression and classification problems, while its application to our MTPP setting will be introduced later in Chapter 3.

Calibration in regression. Consider the case of a univariate regression problem involving a target variable $Y \in \mathbb{R}$ depending on an input variable $X \in \mathbb{R}$. Suppose that we have access to a dataset of n i.i.d. observations $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$ drawn from some joint distribution P_{XY}. To $\mathcal{D}$, we fit a probabilistic predictor $F_{\hat{\boldsymbol{\theta}}} : \mathbb{R} \to \mathcal{F}$ parametrized by $\hat{\boldsymbol{\theta}} \in \Theta$ that maps a realization x of X to a predictive CDF $F(\cdot|x; \hat{\boldsymbol{\theta}}) \in [0, 1]$ in the space $\mathcal{F}$ of distributions over $\mathbb{R}$.

The model $F_{\hat{\boldsymbol{\theta}}}$ is said to be (unconditionally) *probabilistically calibrated* (or Probability Integral Transform (PIT) calibrated) if (Dawid, 1984; Kuleshov et al., 2018)

$$\mathbb{P}\big(F(Y|X; \hat{\boldsymbol{\theta}}) \leq p\big) = p, \quad \forall p \in [0, 1], \tag{2.78}$$

where the probability is taken over X and Y. Intuitively, this definition implies that, if the predictive CDF $F(y|x; \hat{\boldsymbol{\theta}})$ is well-calibrated, then for $p = 0.9$, 90% of all prediction intervals would contain, on average, the true observation 90% of the time.

An interesting observation is that the left hand side of expression (2.78) corresponds to the CDF of $F(Y|X; \hat{\boldsymbol{\theta}})$, while the right hand side relates to the CDF of a uniform random variable $U \in [0, 1]$ independent of $F(Y|X; \hat{\boldsymbol{\theta}})$. Consequently, evaluating the probabilistic calibration of a forecaster $F_{\hat{\boldsymbol{\theta}}}$ can be achieved by measuring the discrepancy between the two distributions using, e.g. the 1-Wasserstein distance (Zhao et al., 2020; Zhou et al., 2021a) or the Cramér-von Mises distance (Kuleshov et al., 2018; Utpala & Rai, 2021). Let F_Z and F_U denote the CDFs of the random variables $Z = F(Y|X; \hat{\boldsymbol{\theta}})$ and U, respectively. The 1-Wasserstein distance between $F_Z(p;\hat{\boldsymbol{\theta}})$ and $F_U(p)$ is given by (Dheur & Ben Taieb, 2023)

$$d(F_Z, F_U) = \int_0^1 \left| F_Z(p;\hat{\boldsymbol{\theta}}) - F_U(p) \right| dp. \tag{2.79}$$

In practice, given a set of n observations $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$, this distance is approximated using Monte Carlo by discretizing the interval [0,1] in M equidistant values $p_1, ..., p_M$. By evaluating the empirical CDF of PITs at each p_m, i.e.

$$\bar{F}_Z(p_m;\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}[F(Y_i|X_i) \leq p_m], \tag{2.80}$$

and given that $F_U(p_m) = p_m$, an approximation of the distance in (2.79) is given by the *Probabilistic Calibration Error (PCE)* (Zhao et al., 2020; Zhou et al., 2021b; Dheur & Ben Taieb, 2023), defined as

$$\text{PCE} = \frac{1}{M}\sum_{m=1}^{M} \left| \bar{F}_Z(p_m;\hat{\boldsymbol{\theta}}) - p_m \right|. \tag{2.81}$$

Essentially, the higher the PCE, the further the distribution of PITs under the model deviates from a uniform distribution, indicating that the model $F(y|x;\hat{\boldsymbol{\theta}})$ exhibits poor probabilistic calibration. This deviation from a uniform distribution can be further assessed from a variety of statistical tests, such as the Kolmogorov-Smirnov test (Massey, 1951), the Cramér-von Mises test (Cramér, 1928; Mises, 1928), or the Anderson-Darling test (Anderson & Darling, 1952).

Calibration in classification. Consider a univariate classification problem where the target variable $Y \in \mathbb{K} = \{0, ..., K\}$ belongs to a discrete space $\mathbb{K}$ of K labels. Assuming again that we have access to a dataset of n observations $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$ drawn i.i.d. from some distribution P_{XY}, we fit a forecaster $p_{\hat{\boldsymbol{\theta}}} : \mathcal{X} \to \mathcal{Y} : \mathbb{R} \to \mathcal{F}$ with parameters $\hat{\boldsymbol{\theta}} \in \Theta$ that maps a realization x of X to a predictive PMF $p(\cdot|x;\hat{\boldsymbol{\theta}}) \in [0,1]$ in the space $\mathcal{F}$ of discrete distributions over $\mathbb{K}$. In this case, the forecaster $p_{\hat{\boldsymbol{\theta}}}$ is said to be *top-label calibrated* (Guo et al., 2017; Gawlikowski et al., 2023) if

$$\mathbb{P}\left(Y = \underset{y\in\mathbb{K}}{\text{argmax}}\; p(y|X;\hat{\boldsymbol{\theta}}) \mid \max_{y\in\mathbb{K}} p(y|X;\hat{\boldsymbol{\theta}}) = p\right) = p \quad \forall p \in [0,1], \tag{2.82}$$

where the probability is taken over X and Y. Intuitively, if a predictive PMF $p(y|x;\hat{\boldsymbol{\theta}})$ is top-label calibrated, it implies that all predictions made with a confidence level of 0.9 should correctly match the observed label Y 90% of the time.

Denoting $\tilde{y} = \underset{y \in \mathbb{K}}{\operatorname{argmax}}\, p(y|X; \hat{\boldsymbol{\theta}})$ and $\tilde{p} = \max_{y \in \mathbb{K}} p(y|X; \hat{\boldsymbol{\theta}})$ as the top-label prediction and corresponding confidence at X, respectively, quantifying top-label calibration is achieved by taking the expected absolute difference between the left and right hand sides of (2.82), i.e.:

$$\mathbb{E}_{\tilde{p}}\left[\left|\mathbb{P}\left(Y = \tilde{y} \mid \tilde{p} = p\right) - p\right|\right], \tag{2.83}$$

which is approximated by binning the predictions of the model, as detailed next. Let $\{\tilde{y}_i\}_{i=1}^n$ be the predictions made by the model with confidence $\{\tilde{p}_i\}_{i=1}^n$ on a dataset $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^n$. To approximate (2.83), the most common approach consists in partitioning the interval [0,1] in M equal-size bins $b_1, ..., b_M$. Each prediction $\tilde{y}_i$ is then assigned to a bin b_m if its associated confidence $\tilde{p}_i$ lies in the interval $[\frac{m-1}{M}, \frac{m}{M}]$. Then, we evaluate the average confidence and accuracy of predictions within a given bin b_m as

$$\operatorname{acc}(b_m) = \frac{1}{|b_m|} \sum_{\tilde{y}_i \in b_m} \mathbb{1}\left(Y_i = \tilde{y}_i\right) \quad \text{and} \quad \operatorname{conf}(b_m) = \frac{1}{|b_m|} \sum_{\tilde{y}_i \in b_m} \tilde{p}_i, \tag{2.84}$$

where $|b_m|$ denotes the number of predictions in b_m. Finally, an approximation of (2.83), called the *Expected Calibration Error (ECE)* (Naeini et al., 2015) is obtained as

$$\text{ECE} = \frac{1}{M} \sum_{m=1}^{M} \left|\operatorname{acc}(b_m) - \operatorname{conf}(b_m)\right|. \tag{2.85}$$

The ECE is frequently used in the literature as the default metric to assess the top-label calibration of classifiers (Laves et al., 2019; Mehrtash et al., 2020; Thulasidasan et al., 2019; Wenger et al., 2020), and this measure can be used to construct various statistical tests (Vaicenavicius et al., 2019; Widmann et al., 2019; Mortier et al., 2023). Similarly to the PCE metric, a high ECE is indicative that the model $p(y|x; \hat{\boldsymbol{\theta}})$ exhibits poor top-label calibration.

Reliability diagrams. A disadvantage with PCE and ECE metrics is that information regarding the calibration error at individual probability levels $p_1, ..., p_M$, or within individual bins $b_1, ..., b_M$, is lost. Reliability diagrams (Niculescu-Mizil & Caruana, 2005; Pinson & Hagedorn, 2011; Guo et al., 2017; Kuleshov et al., 2018) are visual tools that can be used to assess (probabilistic or top-level) calibration at a fine-grained level for both continuous and discrete predictive distributions.

For $F(y|x;\hat{\boldsymbol{\theta}})$, a reliability diagram is obtained by plotting the empirical CDF of PITs $\bar{F}_Z(p_m;\hat{\boldsymbol{\theta}})$ in (2.80) against all probability levels p_m. Alternatively, a reliability diagram is obtained for $p(y|x,\hat{\boldsymbol{\theta}})$ by plotting $\mathrm{acc}(b_m)$ against $\mathrm{conf}(b_m)$ in (2.84) for all b_m. In both cases, a well calibrated forecaster should align with the diagonal line, and any deviation corresponds to miscalibration. Figure 2.10 shows an illustration of reliability diagrams for a regression and a classification setting. In both cases, the miscalibrated model deviates significantly from the diagonal line.

Intuitively, in a regression setting, a model curve falling below the diagonal means that a proportion smaller than p of observations fell below their corresponding quantile forecast at level p, i.e. $\mathbb{P}\left(Y \leq F^{-1}(p|X;\hat{\boldsymbol{\theta}})\right) < p$, where $F^{-1}(\cdot|X;\boldsymbol{\theta})$ is the inverse CDF transform at X. For instance, on the left panel of Figure 2.10, 20% of observations fell below their corresponding 40% quantile forecast. Conversely, a model curve falling above the diagonal indicates that a too large proportion of observations were observed below the said quantile forecast, i.e. $\mathbb{P}\left(Y \leq F^{-1}(p|X;\hat{\boldsymbol{\theta}})\right) > p$. In both cases, the model demonstrates miscalibration, in the sense that its predictive distributions are inconsistent with the realized outcomes.

In the classification setting, an intuitive interpretation of the reliability diagrams for uncalibrated models can be similarly extracted. Specifically, the right panel of Figure 2.10 shows that, for all predictions made by the uncalibrated model with confidence in the bin [0.7, 0.8], the accuracy reaches only 0.6 instead of 0.75. For that specific confidence bin, the uncalibrated model is therefore *overconfident* in its predictions. Alternatively, if the accuracy for a given bin was to exceed the diagonal line, the uncalibrated model would instead exhibit *underconfidence* for that confidence bin.

Remark 5. A calibrated model doesn't necessarily guarantee that the generated predictions are useful. In a regression setting, consider a forecaster that outputs the marginal CDF $F(y) = \mathbb{P}(Y \leq y)$ for every x. As discussed in Kuleshov et al. (2018), despite the model being probabilistically calibrated in the sense of (2.78), its predictions do not depend on the input X, and hence, they are not very informative. Similarly, in a classification setting, a model that outputs the marginal PMF $p(y) = \mathbb{P}(Y = y)$ for every x is top-label calibrated according to (2.82), but not particularly useful. In practice, we also want the forecaster to be sharp, meaning that the prediction intervals (or estimated PMFs) should be tight around the realized outcome for every input x.

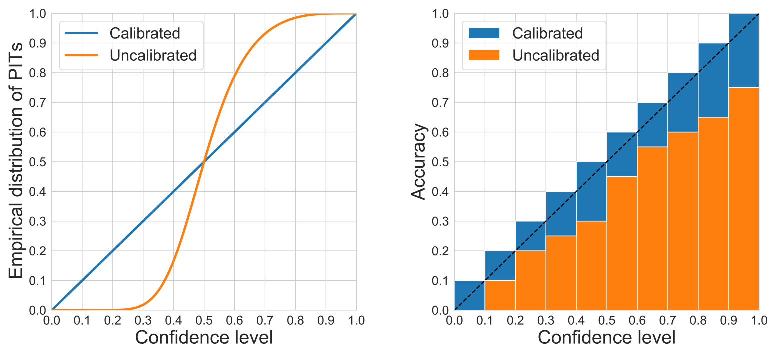


Figure 2.10: Left: In a regression setting, a reliability diagram for $F(y|x;\hat{\boldsymbol{\theta}})$ is obtained by plotting the empirical CDF of PITs in (2.80) against all confidence levels $p \in [0,1]$. Right: In a classification setting, a reliability diagram for $p(y|x;\hat{\boldsymbol{\theta}})$ is obtained by plotting $\mathrm{acc}(b_m)$ against $\mathrm{conf}(b_m)$ in (2.84) for all b_m.

2.2.2 Conformal Prediction

Conformal prediction (CP) (Vovk et al., 2005) is a framework designed to create statistically rigorous uncertainty regions from the predictions of any arbitrary predictive models. Specifically, CP works by transforming the output of the model into *prediction regions* that offer strong finite-sample coverage guarantees. More importantly, CP ensures that these prediction regions are valid in a distribution-free sense, meaning that no assumptions are made regarding the underlying data distribution. This property allows conformal prediction to provide robust uncertainty estimates across a wide range of applications and models. This section aims to introduce the reader to the basic concepts of conformal prediction, and is mostly based on the works of Angelopoulos & Bates (2023) and Tibshirani (2023).

Marginal coverage guarantee. Suppose that we have access to a dataset of n training samples $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{n}$ drawn i.i.d. from some joint distribution P_{XY} with $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$. Depending on whether we are working in a regression or classification setting, the target space $\mathcal{Y}$ can either be continuous, i.e. $\mathcal{Y} = \mathbb{R}$, or discrete, i.e. $\mathcal{Y} = \mathbb{K} = \{0, 1, ..., K\}$, with K the number of classes. For clarity of exposition, we focus our discussion to a regression setting. The concepts introduced in this section can be directly extended to a classification problem, as we will see in Chapter 5.

Similar to the previous section, suppose that we have access to a predictive model $F(y|x;\hat{\boldsymbol{\theta}})$ fitted to the observations in $\mathcal{D}$. Given a new test input X_{n+1}, we wish to construct a distribution-free prediction interval $R(X_{n+1};\hat{\boldsymbol{\theta}}) \subseteq \mathbb{R}$ for the target Y_{n+1} where (X_{n+1}, Y_{n+1}) is also assumed be drawn i.i.d. from P_{XY}. In particular, given a user-specified nominal miscoverage level α, we want this prediction region to verify

$$\mathbb{P}[Y_{n+1} \in R(X_{n+1};\hat{\boldsymbol{\theta}})] \geq 1-\alpha, \tag{2.86}$$

where the probability is taken over all $\{(X_i, Y_i)\}_{i=1}^n \cup (X_{n+1}, Y_{n+1})$, hence the name *marginal coverage* given to this requirement. In other words, we want that among all prediction regions generated for a new input X_{n+1}, at least a fraction $1-\alpha$ of them contains the true observation Y_{n+1}.

Let $Q(\alpha|x;\hat{\boldsymbol{\theta}}) = F^{-1}(\alpha\ |x;\hat{\boldsymbol{\theta}})$ denote the α-quantile of $F(y|x;\hat{\boldsymbol{\theta}})$. A natural first approach to construct $R(X_{n+1};\hat{\boldsymbol{\theta}})$ is to take the interval delimited by the quantiles $Q(^{\alpha}/_{2}|X_{n+1};\hat{\boldsymbol{\theta}})$ and $Q(^{1-\alpha}/_{2}|X_{n+1};\hat{\boldsymbol{\theta}})$, i.e.

$$R(X_{n+1};\hat{\boldsymbol{\theta}}) = \left[Q(^{\alpha}/_{2}|X_{n+1};\hat{\boldsymbol{\theta}}), Q(^{1-\alpha}/_{2}|X_{n+1};\hat{\boldsymbol{\theta}})\right]. \tag{2.87}$$

However, this interval is only guaranteed to satisfy the requirement in (2.86) if $F(y|x;\hat{\boldsymbol{\theta}})$ corresponds to the true, unobserved data distribution $F(y|x)$. Unfortunately, due to model misspecification or lack of training data, $F(y|x;\hat{\boldsymbol{\theta}})$ can be a poor estimate of $F(y|x)$, meaning that the interval in (2.87) fails to achieve marginal coverage. Fortunately, CP enables us to build prediction intervals that achieve (2.86) in finite-sample, even when the model is unreliable. In the following, we introduce *split conformal prediction* (Papadopoulos et al., 2002), a framework of CP that is widely used in practice given its computational efficiency.

Split conformal prediction. This approach, which involves partitioning the data, is relatively simple but effective in transforming a heuristic notion of uncertainty—any trained predictive model—into a rigorous one (Angelopoulos & Bates, 2023). Prior to elaborate further on this methodology, we need to define a *non-conformity score* function $s\left(x, y; F(y|x;\hat{\boldsymbol{\theta}})\right) : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ as a measure of disagreement between an observation y and the predictive model $F(y|x;\hat{\boldsymbol{\theta}})$. To ease notations in the remainder of the discussion, we will drop the dependency of s on $F(y|x;\hat{\boldsymbol{\theta}})$, i.e. $s(x,y) = s\left(x, y; F(y|x;\hat{\boldsymbol{\theta}})\right)$. Returning to our regression problem, an example of non-conformity score involves taking the maximum difference between the target y and the nearest of the quantiles

$Q(\alpha/2|x;\hat{\boldsymbol{\theta}})$ and $Q(1-\alpha/2|x;\hat{\boldsymbol{\theta}})$ (Romano et al., 2019), i.e.

$$s(x,y) = \max\{Q(\alpha/2|x;\hat{\boldsymbol{\theta}}) - y, y - Q(1-\alpha/2|x;\hat{\boldsymbol{\theta}})\}. \tag{2.88}$$

The score in (2.88) has an intuitive interpretation: the further the observation falls outside of $\left[Q(\alpha/2|x;\hat{\boldsymbol{\theta}}), Q(1-\alpha/2|x;\hat{\boldsymbol{\theta}})\right]$, the greater is $s(x,y)$, and so is the disagreement between y and the model. Conversely, the closer y is to the central point of the interval, the smaller is $s(x,y)$, and so is the agreement between y and the model. In practice, alternative choices to (2.88) are available (Kato et al., 2023), the only requirement being that the score must be negatively-oriented (Angelopoulos & Bates, 2023), i.e. lower values must encode lower agreement.

Consider the dataset $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{n}$ of n i.i.d. observations and the predictive model $F(y|x;\boldsymbol{\theta})$[3]. The split conformal procedure to construct a prediction interval for a new target Y_{n+1} at coverage level $1-\alpha$ can be summarized in the following steps:

1. Split $\mathcal{D}$ into two non-overlapping sets, $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{cal}}$ with $\mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{cal}} = \mathcal{D}$.

2. Fit $F(y|x;\boldsymbol{\theta})$ to the observations in $\mathcal{D}_{\text{train}}$, yielding a heuristic measure of uncertainty $F(y|x;\hat{\boldsymbol{\theta}})$.

3. Use $F(y|x;\hat{\boldsymbol{\theta}})$ to define a non-conformity score function
$$s(x,y) = s\left(x, y; F(y|x;\hat{\boldsymbol{\theta}})\right). \tag{2.89}$$

4. Compute the calibration scores using the observations in $\mathcal{D}_{\text{cal}}$, i.e.
$$\{\, s_i \,\}_{i=1}^{|\mathcal{D}_{\text{cal}}|} := \{\, s\left(X_i, Y_i\right) : (X_i, Y_i) \in \mathcal{D}_{\text{cal}} \,\} \tag{2.90}$$

5. Compute the adjusted $1-\alpha$ empirical quantile of these calibration scores:
$$\hat{q} = \text{Quantile}\left(s_1, ..., s_{|\mathcal{D}_{\text{cal}}|} \cup \{\, \infty \,\}; \frac{\lceil (|\mathcal{D}_{\text{cal}}|+1)(1-\alpha) \rceil}{|\mathcal{D}_{\text{cal}}|}\right). \tag{2.91}$$

6. For a new test input X_{n+1}, use $\hat{q}$ to construct a prediction region for Y_{n+1} with a $1-\alpha$ coverage level as follows:
$$R(X_{n+1};\hat{\boldsymbol{\theta}}) = \{y \in \mathcal{Y} : s(X_{n+1}, y) \leq \hat{q}\}. \tag{2.92}$$

[3]Note that the notation $\boldsymbol{\theta}$ for the parameters means that the model is not yet trained.

We can show that prediction interval $R(X_{n+1};\hat{\boldsymbol{\theta}})$ obtained from the split conformal procedure satisfies the marginal coverage property in (2.86).

Proof. Let $\{\, s_i \,\}_{i=1}^{|\mathcal{D}_{\text{cal}}|} := \{\, s\,(X_i, Y_i) : (X_i, Y_i) \in \mathcal{D}_{\text{cal}} \,\}$ and $s_{n+1} = s(X_{n+1}, Y_{n+1})$. By definition of $R(X_{n+1};\hat{\boldsymbol{\theta}})$, we have

$$Y_{n+1} \in R(X_{n+1};\hat{\boldsymbol{\theta}}) \iff s_{n+1} \leq \hat{q}, \tag{2.93}$$

which implies that

$$\mathbb{P}(Y_{n+1} \in R(X_{n+1};\hat{\boldsymbol{\theta}})) = \mathbb{P}(s_{n+1} \leq \hat{q}). \tag{2.94}$$

Let us denote $s_{\lceil(|\mathcal{D}_{\text{cal}}|+1)(1-\alpha)\rceil}$ as the $\lceil(|\mathcal{D}_{\text{cal}}|+1)(1-\alpha)\rceil$-smallest values among $\{\, s_i \,\}_{i=1}^{|\mathcal{D}_{\text{cal}}|}$. As the scores $\{\, s_i \,\}_{i=1}^{|\mathcal{D}_{\text{cal}}|}$ and s_{n+1} are i.i.d. random variables, we have

$$\begin{aligned}\mathbb{P}(s_{n+1} \leq \hat{q}) &= \mathbb{P}(s_{n+1} \leq s_{\lceil(|\mathcal{D}_{\text{cal}}|+1)(1-\alpha)\rceil}) &(2.95)\\ &= \frac{\lceil(|\mathcal{D}_{\text{cal}}|+1)(1-\alpha)\rceil}{|\mathcal{D}_{\text{cal}}|+1} &(2.96)\end{aligned}$$

This last equality essentially tells us that s_{n+1} is equally likely to fall in between any score in $\{\, s_i \,\}_{i=1}^{|\mathcal{D}_{\text{cal}}|}$. Combining (2.94) and (2.96), and as $\lceil x \rceil \geq x$, we finally obtain the desired marginal coverage property

$$\mathbb{P}(Y_{n+1} \in R(X_{n+1};\hat{\boldsymbol{\theta}})) = \frac{\lceil(|\mathcal{D}_{\text{cal}}|+1)(1-\alpha)\rceil}{|\mathcal{D}_{\text{cal}}|+1} \geq 1-\alpha. \tag{2.97}$$

$\square$

Moreover, if no ties between the scores occur with probability almost one, then marginal coverage is also upper bounded (Theorem 2.2. of Lei et al. (2018)), i.e.

$$1-\alpha \leq \mathbb{P}(Y_{n+1} \in R(X_{n+1};\hat{\boldsymbol{\theta}})) \leq 1-\alpha+\frac{1}{|\mathcal{D}_{\text{cal}}|+1}. \tag{2.98}$$

Returning to our regression problem, we can show that the split conformal procedure combined with the non-conformity score in (2.88) finally leads to the following prediction interval:

$$R(X_{n+1};\hat{\boldsymbol{\theta}}) = [Q(\alpha/2|X_{n+1};\hat{\boldsymbol{\theta}}) - \hat{q}, Q(1-\alpha/2|X_{n+1}) + \hat{q};\hat{\boldsymbol{\theta}})], \tag{2.99}$$

which also has an intuitive interpretation. Depending on the value of $\hat{q}$, which accounts for the models' mistakes on the calibration data, $R(X_{n+1};\hat{\boldsymbol{\theta}})$ simply grows or shrink the prediction interval in (2.87) to achieve the desired coverage. This is illustrated on Figure 2.11.

Finally, despite split conformal prediction guaranteeing that marginal coverage is always satisfied, we also seek *informative* prediction intervals, meaning they should be as tight as possible. This will depend mainly on two key factors: the accuracy of the base model $F(y|x;\hat{\boldsymbol{\theta}})$, and the quality of the chosen score function. Nonetheless, interval size is not the only factor that we should consider when evaluating a conformal approach, as we will discuss later.

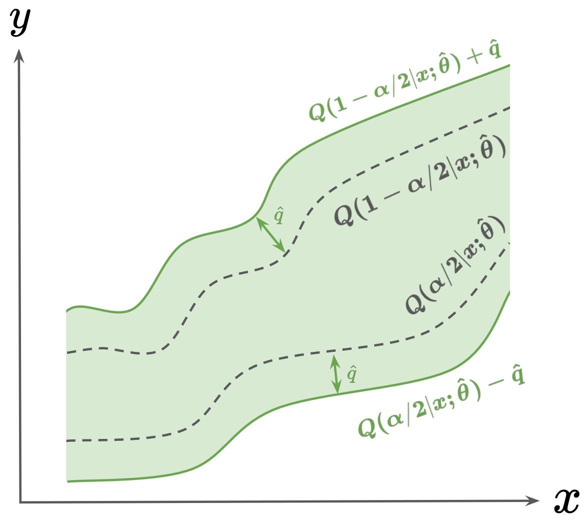

y
x
$Q(1-\alpha/2|x;\hat{\theta})+\hat{q}$
$Q(1-\alpha/2|x;\hat{\theta})$
$Q(\alpha/2|x;\hat{\theta})$
$Q(\alpha/2|x;\hat{\theta})-\hat{q}$
$\hat{q}$
$\hat{q}$

Figure 2.11: Illustration of the prediction interval returned by 2.99. As in this case $\hat{q} > 0$, the split conformal procedure grows the original prediction interval in 2.87.

Remark 6. Although we focused our discussion on a regression problem involving a model that outputs a predictive CDF $F(y|x;\hat{\boldsymbol{\theta}})$, the split conformal prediction remains valid for **any** heuristic notions of uncertainty, such as PDFs, quantile functions, or point estimates. In particular, the procedure does not change either in a classification setting, and only the non-conformity score function must be adapted to the problem considered. We will discuss other conformal approaches in both regression and classification settings later in Chapter 5.

Remark 7. So far, we assumed that the pairs $\{(X_i, Y_i)\}_{i=1}^n$ and (X_{n+1}, Y_{n+1}) were drawn i.i.d. from the unknown distribution P_{XY}. In practice, the split conformal procedure only requires that these observations are *exchangeable*, which is weaker than the i.i.d. assumption. Formally, we say that $\{(X_i, Y_i)\}_{i=1}^n \cup (X_{n+1}, Y_{n+1})$ are exchangeable if, for any permutation $\pi_1, ..., \pi_{n+1}$ of the indices $1, ..., n+1$, their joint distribution P remains unchanged (Tibshirani, 2023), i.e.

$$P_{X_1,...,X_{n+1}}(X_1, ..., X_{n+1}) = P_{X_{\pi_1},...,X_{\pi_{n+1}}}(X_1, ..., X_{n+1}). \tag{2.100}$$

Conditional coverage and adaptivity. Recall that the coverage guarantee in (2.86) is a marginal property taken over all $\{(X_i, Y_i)\}_{i=1}^n \cup (X_{n+1}, Y_{n+1})\}$. Although marginal coverage is a desirable property, we might also seek to derive a prediction region that achieves a stronger notion of conditional coverage, i.e.

$$\mathbb{P}[Y_{n+1} \in R(X_{n+1}; \hat{\boldsymbol{\theta}}) | X_{n+1} = x] \geq 1 - \alpha \quad \forall x \in \mathcal{X}. \tag{2.101}$$

Unfortunately, such conditional coverage guarantee is not achievable without relying on strong distributional assumptions (Vovk, 2012; Foygel Barber et al., 2021). Nonetheless, we may still enforce the generated prediction region $R(X_{n+1}; \hat{\boldsymbol{\theta}})$ to satisfy an approximate notion of conditional coverage. To achieve this in practice, we want the conformal approach to be *adaptive* to the input x. In particular, we want the prediction region to be tight when the confidence of the model at x is high, and large when its confidence is low. For instance, the prediction interval defined in (2.92) adapts to different values of x, shrinking or growing based on the uncertainty of the base model. In Chapter 5, we will discuss several metrics to evaluate the adaptivity of different conformal approaches in terms of their ability to approximate (2.101).

CHAPTER 3

On the Predictive Accuracy of Neural MTPP Models

The classical parametrizations of MTPP models introduced in Section 2.1.2 have found successful applications in modeling event sequences across diverse sets of real-world applications. These include seismology (Ogata, 1988; Rotondi & Varini, 2019), crime analysis (Egesdal et al., 2010; Mohler et al., 2011), epidemiology (Choi et al., 2015; Rizoiu et al., 2018), finance (Bacry & Muzy, 2014; Bacry et al., 2015; Hawkes, 2018), or even social interactions (Guo et al., 2015; Zhao et al., 2015; Lukasik et al., 2016). Nonetheless, these classical models usually rely on strong modeling assumptions, which inherently limit their flexibility in capturing the complex dynamics of real-world point patterns (Mei & Eisner, 2017). To illustrate this statement, recall the Hawkes process with exponential kernels, whose marked intensity functions are given by

$$\lambda_k^*(t) = \lambda_k + \sum_{k'=1}^{K} \sum_{(t_j,k_j)\in\mathcal{H}_t^{k'}} \gamma_{k,k'} \exp\left(-\beta_{k,k'}(t-t_j)\right), \tag{3.1}$$

where $\lambda_k \in \mathbb{R}_+$, $\gamma_{k,k'} \in \mathbb{R}_+$, $\beta_{k,k'} \in \mathbb{R}_+$ and $\mathcal{H}_t^{k'} = \{(t_j, k_j) \in \mathcal{S} \mid t_j < t, k_j = k'\}$. In (3.1), the influence of past events on future ones is by design

1. *Positive*: $\lambda_k^*(t)$ is raised by a factor $\gamma_{k,k'}$ every time a new event of mark k' appears.
2. *Additive*: Each event contributes independently and additively to $\lambda_k^*(t)$ through the sum in (3.1).
3. *Exponentially decaying with time*: After the initial excitation jump, the influence of a past observation on $\lambda_k^*(t)$ fades away exponentially with decay parameter $\beta_{k,k'}$.

However, it is not a difficult task to imagine scenarios where (1), (2), and (3) are no longer valid. Indeed, in situations where events have an inhibiting rather than exciting influence on each other, assumption (1) is no longer satisfied. For instance, buying a new car usually inhibits the likelihood of purchasing a next one in the foreseeable future. Secondly, assumption (2) can be violated in situations where an event of mark k' at time t_1 triggers an excitation $\gamma_{k,k'}^1$, different than the excitation $\gamma_{k,k'}^2$ of another event of mark k' at time $t_2 > t_1$. For instance, we may play a newly discovered song on repeat at first, but the initial excitation is likely to fade after the 100th hearing, meaning that $\gamma_{k,k'}^2 < \gamma_{k,k'}^1$. Finally, assumption (3) does not hold for processes where the intensity is expected to increase over time. This can be exemplified by the modeling of large magnitude earthquakes, where stress build-up over time increases the likelihood of future occurrences (Ogata & Vere-Jones, 1984).

To address these limitations, several studies focused on proposing modifications of the Hawkes process to capture additional effects that the original formulation in (3.1) would miss by design. In this line of work, Zhou et al. (2013) replaces the exponential kernel with a flexible non-parametric triggering kernel estimated directly from data, while Lee et al. (2016) defines the excitation parameters $\gamma_{k',k}$ as stochastic, rather than constant over time. To capture non-linear effects of past events, Wang et al. (2016) proposed to process the intensity of the Hawkes process function via a non-parametric isotonic function. Subsequent developments then enabled to encompass complementary effects, such as joint excitation and inhibition (Chen et al., 2019; Costa et al., 2020; Duval et al., 2022).

While a common thread of these works focused on enhancing the flexibility of MTPP models, others pursued a different direction, leveraging recent advances in deep learning techniques to capture complex event dynamics. The resulting framework, named *neural MTPP models* (Shchur et al., 2021b), has experienced rapid development since its introduction by the seminal work of Du et al. (2016), with the emergence of numerous novel architectures and applications.

Given a sequence of events, neural MTPP models typically involve a combination of three main architectural components: 1) an event encoder, creating a fixed-sized embedding for each event in the sequence, 2) a history encoder, generating a representation of the history from the embeddings of past events, and 3) a decoder parametrizing a function that fully characterizes a predictive distribution over future arrival times and marks. Among other possibilities, improvements with respect to existing baselines are obtained by proposing alternatives to either of these components. For instance, one can replace a RNN-based history encoder with a self-attentive one, or choose to parametrize a certain MTPP function that leads to useful properties, such as reduced computational costs or closed-form sampling. However, as pointed out by Shchur et al. (2021b), "*new architectures often change all these components at once, which makes it hard to pinpoint the source of empirical gains*". Moreover, the baselines against which a newly proposed architecture is compared, as well as the datasets employed and the experimental setups, often differ from paper to paper, which renders a fair comparison even harder.

In practice, the average sequence NLL is a commonly used metric to evaluate the predictive performance of neural MTPP models. However, reporting a single NLL value encompasses the contributions of both predictive distributions of arrival times and marks, effectively obscuring model performance in fitting each predictive distribution separately. Moreover, while the NLL is useful for comparing different baselines, it is difficult to interpret. In the context of forecasting theory, probabilistic calibration is a desired property that refers to the statistical consistency between the predictive distributions extracted from the model and the actual outcomes (Gneiting et al., 2007; Dheur & Ben Taieb, 2023). However, while probabilistic calibration is a property that any competent or ideal predictive distribution should possess (Gneiting et al., 2007; Dheur & Ben Taieb, 2023), assessing the calibration of neural MTPP models has been generally overlooked by the community, both for the time and mark predictive distributions.

In this chapter, our objective is to address the aforementioned concerns by presenting the following contributions:

- We perform a large-scale experimental study to assess the predictive accuracy of state-of-the-art neural MTPP models on 15 real-world event sequence datasets in a carefully designed unified setup. Our study also includes classical parametric MTPP models as well as synthetic datasets. In particular, we study the influence of each major architectural component (event encoding, history encoder, and decoder) for both the time and mark prediction tasks.
- We assess the probabilistic calibration of neural MTPP models, both for the time and mark predictive distributions. To this end, we employ standard metrics and tools borrowed from the forecasting literature, namely the probabilistic calibration error and reliability diagrams. While the distribution of arrival times is generally well-calibrated, our research reveals that classical parametric baselines exhibit better calibration of mark predictive distributions compared to neural MTPP models.
- Among other findings, we found that neural MTPP models often do not fully leverage the complete information contained in all historical events. In fact, relying solely on a subset of the most recent observed occurrences can yield comparable performance to encoding the entire historical context. Furthermore, we demonstrate the high sensitivity of various decoder parametrizations to the event encoder, highlighting the significant gains in predictive accuracy that can be achieved through appropriate selection. In addition, Lastly, our study shows that several commonly used event sequence datasets within the TPP literature may not be suitable for accurately benchmarking neural MTPP baselines.

In this chapter, we begin our discussion with a comprehensive overview of the general encoder-decoder framework upon which neural MTPP models are commonly built, along with a detailed description of its major architectural components. Then, in Section 3.3, we continue with a large scale experimental study of these neural MTPP models in a unified setup, highlighting their strengths and weaknesses across a broad of datasets and evaluation metrics.

3.1. Neural MTPP Models

Recall that a realization of a MTPP is an ordered sequence of n events $\mathcal{S} = \{\boldsymbol{e}_i = (t_i, k_i)\}_{i=1}^{n}$ observed in $[0, T]$, where $t_i \in \mathbb{R}_+$ is the i^{th} event's arrival time, and $k_i \in \mathbb{K}$ is its categorical mark. Equivalently, $\mathcal{S} = \{\boldsymbol{e}_i = (\tau_i, k_i)\}_{i=1}^{n}$, where $\tau_i = t_i - t_{i-1}$ is the i^{th} event's inter-arrival time.

Moreover, recall that defining a MTPP model involves specifying either of the functions $f^*(t, k; \boldsymbol{\theta})$, $\lambda_k^*(t; \boldsymbol{\theta})$, $\Lambda_k^*(t; \boldsymbol{\theta})$ or $F^*(t, k; \boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$, provided that the chosen parametrization defines a valid joint distribution of arrival times and marks. Essentially, a *neural* MTPP model carries this task by leveraging the flexibility of neural network architectures. A neural MTPP model often involves three principal components (Shchur et al., 2021b):

1. An **event encoder** $u(\cdot) : \mathbb{R}_+ \times \mathbb{K} \to \mathbb{R}^{d_l}$ which, for each $\boldsymbol{e}_i \in \mathcal{S}$, generates a fixed size embedding $\boldsymbol{l}_i \in \mathbb{R}^{d_l}$.
2. A **history encoder** $\text{ENC}(\cdot) : \mathcal{P}\big(\mathbb{R}^{d_l}\big) \to \mathbb{R}^{d_h}$, which for each $\boldsymbol{e}_i \in \mathcal{S}$, generates a fixed size history embedding $\boldsymbol{h}_i \in \mathbb{R}^{d_h}$ from past event representations $\{\boldsymbol{l}_1, ..., \boldsymbol{l}_{i-1}\}$. Here, $\mathcal{P}(\mathbb{R}^{d_l})$ represents the power set of $\mathbb{R}^{d_l}$.
3. A **decoder**, which given $\boldsymbol{h}_i$ and a query time $t_{i-1} < t \leq t_i$, parametrizes a function that uniquely characterizes the MTPP, e.g. $\lambda_k^*(t; \boldsymbol{\theta})$.

Figure 3.1 illustrates the general modeling pipeline of neural MTPP models. In the next section, we discuss each of the principal components in more detail.

3.1.1 Event Encoding

The task of event encoding consists in generating a representation $\boldsymbol{l}_i \in \mathbb{R}^{d_l}$ for each event in $\mathcal{S}$, which will be passed to the history encoder, and eventually to the decoder, at a later stage. This task essentially boils down to finding an embedding $\boldsymbol{l}_i^t \in \mathbb{R}^{d_t}$ for the (inter-)arrival time t_i (τ_i), and an embedding $\boldsymbol{l}_i^k \in \mathbb{R}^{d_k}$ for the associated mark k_i. The event embedding $\boldsymbol{l}_i \in \mathbb{R}^{d_l}$ with

$d_l = d_t + d_k$ is finally obtained through some combination of $\boldsymbol{l}_i^t$ and $\boldsymbol{l}_i^k$, e.g. via concatenation:

$$\boldsymbol{l}_i = u(\boldsymbol{e}_i) = \begin{bmatrix} \boldsymbol{l}_i^t \\ \boldsymbol{l}_i^k \end{bmatrix}. \tag{3.2}$$

Encoding the (inter-)arrival times. A straightforward approach to obtain $\boldsymbol{l}_i^t$ is to select the raw inter-arrival times as time embeddings $\boldsymbol{l}_i^t = \tau_i$ or their logarithms $\boldsymbol{l}_i^t = \log \tau_i$ (Omi et al., 2019; Shchur et al., 2020a; Mei & Eisner, 2017; Du et al., 2016). Inspired by the positional encoding of Transformer architectures (Vaswani et al., 2017) and their extension to temporal data (Kazemi et al., 2019), other works exploit more expressive representations by encoding the arrival times as vectors of sinusoidal functions (Enguehard et al., 2020):

$$\boldsymbol{l}_i^t = \bigoplus_{j=0}^{d_t/2-1} \sin\left(\beta_j t_i\right) \oplus \cos\left(\beta_j t_i\right), \tag{3.3}$$

where $\beta_j \propto 1000^{\frac{-2j}{d_t}}$ and $\oplus$ is the concatenation operator. Variants of sinusoidal encoding can also be found in Zhang et al. (2020); Zuo et al. (2020); Mei et al. (2020); Yuan et al. (2023). Alternatively, learnable embeddings can be obtained by feeding the inter-arrival times to a Multi-Layer Perceptron (MLP) architecture (Goodfellow et al., 2016):

$$\boldsymbol{l}_i^t = \sigma_R\left(\boldsymbol{w}_2 \sigma_R\left(\boldsymbol{w}_1 \tau_i + \boldsymbol{b}_1\right)^\intercal + \boldsymbol{b}_2\right), \tag{3.4}$$

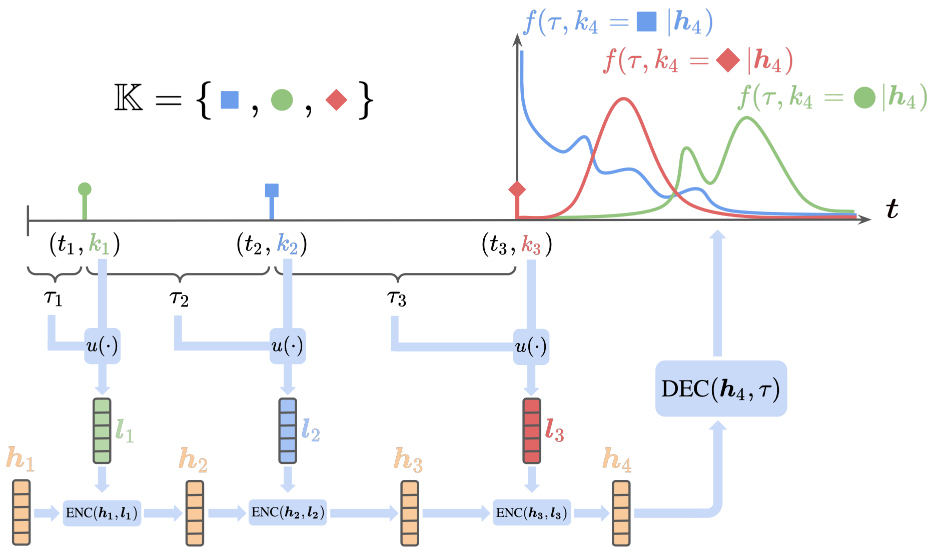
where $\sigma_R(x) = \max(0, x)$ is the ReLU activation function (Fukushima, 1969) and $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{b}_1, \boldsymbol{b}_2 \in \mathbb{R}^{d_l}$.

Encoding the mark: When marks are available, the most common approach to obtain the mark embeddings is achieved by specifying a learnable embedding matrix $\mathbf{W}^k \in \mathbb{R}^{d_k \times K}$ (Du et al., 2016; Zhang et al., 2020; Zuo et al., 2020; Li et al., 2023). Given the one-hot encoding $\boldsymbol{k}_i \in \{0,1\}^K$ of mark k_i, its embedding $\boldsymbol{l}_i^k \in \mathbb{R}^{d_k}$ is obtained as

$$\boldsymbol{l}_i^k = \mathbf{W}^k \boldsymbol{k}_i. \tag{3.5}$$

3.1.2 History Encoding

The general principle behind history encoding in the context of neural MTPP models is to construct a fixed-size embedding $\boldsymbol{h}_i \in \mathbb{R}^{d_h}$ for an event $\boldsymbol{e}_i$ from the sequence of past events representations $\{\boldsymbol{l}_1, ..., \boldsymbol{l}_{i-1}\}$, using an auto-regressive



$\mathbb{K} = \{ ■ , ● , ◆ \}$
$f(\tau, k_4 = ■ | \boldsymbol{h}_4)$
$f(\tau, k_4 = ◆ | \boldsymbol{h}_4)$
$f(\tau, k_4 = ● | \boldsymbol{h}_4)$
t
(t_1, k_1)
(t_2, k_2)
(t_3, k_3)
τ_1
τ_2
τ_3
$u(\cdot)$
$u(\cdot)$
$u(\cdot)$
$\mathrm{DEC}(\boldsymbol{h}_4, \tau)$
$\boldsymbol{l}_1$
$\boldsymbol{l}_2$
$\boldsymbol{l}_3$
$\boldsymbol{h}_1$
$\boldsymbol{h}_2$
$\boldsymbol{h}_3$
$\boldsymbol{h}_4$
$\mathrm{ENC}(\boldsymbol{h}_1, \boldsymbol{l}_1)$
$\mathrm{ENC}(\boldsymbol{h}_2, \boldsymbol{l}_2)$
$\mathrm{ENC}(\boldsymbol{h}_3, \boldsymbol{l}_3)$

Figure 3.1: General workflow of neural parametrizations for MTPP models. First, an event encoder $u(\cdot)$ takes the events' inter-arrival times τ_i and marks k_i and outputs an event embedding $\boldsymbol{e}_i$. Event embeddings are then passed through an auto-regressive mechanism $\mathrm{ENC}(\cdot)$ (e.g. a RNN or a self-attention encoder) to create the history embedding $\boldsymbol{h}_i$. Finally, the history embedding of the last observed event is fed to a decoder to estimate, e.g. the joint density $f^*(\tau, k; \boldsymbol{\theta})$ of the next event.

mechanism or a set aggregator. Naturally, the main goal when constructing $\boldsymbol{h}_i$ is to capture as closely as possible relevant dynamics in the history of the process, in order to accurately estimate the distribution of arrival times and marks of future events.

Recurrent architectures. A natural choice to handling ordered sequence of events is to rely on a form of recurrent architecture (Du et al., 2016; Mei & Eisner, 2017; Guo et al., 2018b; Shchur et al., 2020a; Soen et al., 2021). In this context, starting from an initial state $\boldsymbol{h}_1$ initialized at random, the history embedding $\boldsymbol{h}_i$ of an event $\boldsymbol{e}_i$ is constructed by sequentially updating the history embeddings at the previous time steps by using the next event's representation, i.e.

$$\begin{aligned} \boldsymbol{h}_2 &= \mathrm{ENC}\big(\boldsymbol{h}_1, \boldsymbol{l}_1\big), \\ &\dots \\ \boldsymbol{h}_i &= \mathrm{ENC}\big(\boldsymbol{h}_{i-1}, \boldsymbol{l}_{i-1}\big), \end{aligned} \tag{3.6}$$

where $\boldsymbol{h}_1 \in \mathbb{R}^{d_h}$ is the initial state. For instance, a common choice for the update function is the Gated Recurrent Unit (GRU) (Cho et al., 2014), which can be summarized as:

$$\boldsymbol{r}_i = \sigma_{\text{si}}\left(\mathbf{W}_r^h \boldsymbol{h}_{i-1} + \mathbf{W}_r^l \boldsymbol{l}_{i-1} + \boldsymbol{b}_r\right), \tag{3.7}$$

$$\boldsymbol{z}_i = \sigma_{\text{si}}\left(\mathbf{W}_z^h \boldsymbol{h}_{i-1} + \mathbf{W}_z^l \boldsymbol{l}_{i-1} + \boldsymbol{b}_z\right), \tag{3.8}$$

$$\boldsymbol{n}_i = \tanh\left(\mathbf{W}_n^l \boldsymbol{l}_{i-1} + \boldsymbol{b}_n^l + \boldsymbol{r}_i \circ \left(\mathbf{W}_n^h \boldsymbol{h}_{i-1} + \boldsymbol{b}_n^h\right)\right), \tag{3.9}$$

$$\boldsymbol{h}_i = (1 - \boldsymbol{z}_i) \circ \boldsymbol{n}_i + \boldsymbol{z}_i \circ \boldsymbol{h}_{i-1}, \tag{3.10}$$

where $\sigma_{\text{si}}(x) = 1/(1 + e^{-x})$ is the sigmoid activation function, $\mathbf{W}_r^h$, $\mathbf{W}_z^h$, $\mathbf{W}_n^h \in \mathbb{R}^{d_h \times d_h}$, $\mathbf{W}_r^l$, $\mathbf{W}_z^l$, $\mathbf{W}_n^l \in \mathbb{R}^{d_h \times d_l}$, $\boldsymbol{b}_r, \boldsymbol{b}_z$, $\boldsymbol{b}_n^l$, $\boldsymbol{b}_n^h \in \mathbb{R}^{d_h}$, and $\circ$ is the Hadamard product. Intuitively, the reset gate $\boldsymbol{r}_i \in [0,1]^{d_h}$ tells the network how much information from the previous steps $\boldsymbol{h}_{i-1}$ should be ignored, whereas the update gate $\boldsymbol{z}_i \in [0,1]^{d_h}$ balances the new information contained in the candidate activation $\boldsymbol{n}_i$ with the old one in $\boldsymbol{h}_{i-1}$ to output $\boldsymbol{h}_i$. This architecture, together with the Long-Short Term Memory (LSTM) network (Hochreiter & Schmidhuber, 1997), was initially developed to more effectively capture long-term dependencies in data sequences by mitigating the problem of exploding/vanishing gradients found in classical RNN architectures. Nonetheless, the inherent sequential nature of RNN prevents them from computing each $\boldsymbol{h}_i$ in parallel. This is in contrast to Self-Attention (SA) mechanisms, introduced next.

Self-attentive encoders. As an alternative to recurrent architectures, the SA mechanism of Transformers (Vaswani et al., 2017) computes the history embeddings $\boldsymbol{h}_i$ for each $\boldsymbol{e}_i$ independently as follows (Zhang et al., 2020; Zuo et al., 2020; Enguehard et al., 2020; Li et al., 2023):

$$\boldsymbol{h}_i^{(h)} = \text{SA}^{(h)}\left(\boldsymbol{q}_i^{(h)}, \mathbf{K}_i^{(h)}, \mathbf{V}_i^{(h)}\right) \tag{3.11}$$

$$= \mathbf{W}_2^{(h)} \sigma_R\left(\mathbf{W}_1^{(h)} \left(\bar{\boldsymbol{h}}_i^{(h)}\right)^\top + \boldsymbol{b}_1^{(h)}\right) + \boldsymbol{b}_2^{(h)}, \tag{3.12}$$

with

$$\bar{\boldsymbol{h}}_i^{(h)} = \text{Softmax}\left(\frac{\left(\boldsymbol{q}_i^{(h)}\right)^\top \mathbf{K}_i^{(h)}}{\sqrt{d_q}}\right)\left(\mathbf{V}_i^{(h)}\right)^\top, \tag{3.13}$$

where $\mathrm{SA}^{(h)}(\cdot)$ refers to the h^{th} out of H parallel SA mechanisms (heads) between a *query* vector $\boldsymbol{q}_i^{(h)}$, a *key* matrix $\mathbf{K}_i^{(h)}$ and a *value* matrix $\mathbf{V}_i^{(h)}$ defined as

$$\boldsymbol{q}_i^{(h)} = \mathbf{W}_Q^{(h)} \boldsymbol{l}_{i-1}, \quad \mathbf{K}_i^{(h)} = \mathbf{W}_K^{(h)} [\boldsymbol{l}_1, ..., \boldsymbol{l}_{i-1}], \quad \mathbf{V}_i^{(h)} = \mathbf{W}_V^{(h)} [\boldsymbol{l}_1, ..., \boldsymbol{l}_{i-1}], \tag{3.14}$$

where $\mathbf{W}_Q^{(h)}, \mathbf{W}_K^{(h)} \in \mathbb{R}^{d_q \times d_l}$, $\mathbf{W}_V^{(h)} \in \mathbb{R}^{d_h \times d_l}$, $\mathbf{W}_2^{(h)}$, $\mathbf{W}_1^{(h)} \in \mathbb{R}^{d_h \times d_h}$, $\boldsymbol{b}_1^{(h)}$, $\boldsymbol{b}_2^{(h)} \in \mathbb{R}^{d_h}$, and σ_R is the ReLU activation function. In essence, the history embedding $\boldsymbol{h}_i^{(h)}$ is constructed as a weighted sum of past events' representations, where the attention weights are obtained by measuring a similarity score (e.g. dot product or deep Fourier kernels (Zhu et al., 2021)) between the projection of $\boldsymbol{l}_{i-1}$, and the projections of $\{\boldsymbol{l}_1, ..., \boldsymbol{l}_{i-1}\}$. Finally, the different representations $\boldsymbol{h}_i^{(h)}$ learned by each of the H independent attention heads are concatenated and fed to a feed-forward layer to produce the final history embedding:

$$\boldsymbol{h}_i = \mathbf{W}_3 \left[\bigoplus_{h=1}^{H} \boldsymbol{h}_i^{(h)} \right], \tag{3.15}$$

where $\mathbf{W}_3 \in \mathbb{R}^{d_h \times H d_h}$. Architectures based on SA mechanisms can compute the history embeddings $\boldsymbol{h}_i$ for each event $\boldsymbol{e}_i$ independently in parallel. However, since each $\bar{\boldsymbol{h}}_i$ depends on all preceding events in (3.13), computing $\boldsymbol{h}_i$ for all n events scales in $O(n^2)$ time for architectures based on a SA mechanism. Conversely, computing $\boldsymbol{h}_i$ scales in $O(n)$ time for their recurrent counterparts (Shchur et al., 2021b).

3.1.3 Decoders

The role of the decoder in a neural MTPP model is to define a valid parametrization of $\lambda_k^*(t; \boldsymbol{\theta})$, $\Lambda_k^*(\tau, k; \boldsymbol{\theta})$, $f^*(t, k; \boldsymbol{\theta})$, or $F^*(t, k; \boldsymbol{\theta})$. While being mathematically equivalent, recall that a specific choice among these functions leads to specific requirements, as discussed in Section 2.1.1. Nonetheless, given the encoding $\boldsymbol{l} \in \mathbb{R}^{d_l}$ of a query event $\boldsymbol{e} = (t, k)$ with $t > t_{i-1}$, and its history embedding $\boldsymbol{h}_i$, the parametrization of these functions is systematically achieved by using neural network architectures. We will consider the following state-of-the-art neural MTPP decoders in this study:

- The Exponential Constant (EC) decoder (Upadhyay et al., 2018) parametrizes a constant $\lambda_k^*(t; \boldsymbol{\theta})$ between two events from a MLP that takes $\boldsymbol{h}_i$ as input.

- The Monte-Carlo Multi-Layer Perceptron (MLP/MC) decoder (Enguehard et al., 2020) parametrizes $\lambda_k^*(t;\boldsymbol{\theta})$ from a MLP that takes $\boldsymbol{e}$ and $\boldsymbol{h}_i$ as input.
- Monte-Carlo Self-Attention (SA/MC) decoder (Enguehard et al., 2020) parametrizes $\lambda_k^*(t;\boldsymbol{\theta})$ by letting $\boldsymbol{e}$ attend to all $\{\boldsymbol{h}_1, ..., \boldsymbol{h}_i\}$.
- The Neural Hawkes (NH) decoder (Mei & Eisner, 2017) parametrizes $\lambda_k^*(t;\boldsymbol{\theta})$ from a set of continuous-time LSTM equations.
- The Recurrent Marked Temporal Point Process (RMTPP) decoder (Du et al., 2016) separately parametrizes $\lambda^*(t;\boldsymbol{\theta})$ as a decaying exponential function, and a predictive PMF of marks independent of time given the history.
- The LogNormMix (LNM) decoder (Shchur et al., 2020a) separately parametrizes $f^*(\tau;\boldsymbol{\theta})$ as a mixture of log-normal distributions, and a predictive PMF of marks similarly to RMTPP. Also, we consider the LogNorm (LN) decoder, defined equivalently to LNM but with a single mixture component.
- The Fully Neural Network (FNN) decoder (Omi et al., 2019; Enguehard et al., 2020) parametrizes $\Lambda_k^*(t;\boldsymbol{\theta})$ from a MLP that takes $\boldsymbol{e}$ and $\boldsymbol{h}_i$ as inputs.
- The Cumulative Self-Attention (SA/CM) decoder (Enguehard et al., 2020) parametrizes $\Lambda_k^*(t;\boldsymbol{\theta})$ by letting $\boldsymbol{e}$ attend to all $\{\boldsymbol{h}_1, ..., \boldsymbol{h}_i\}$.

In the following, we will describe and discuss each of the above decoders in more details.

EC decoder. This is the simplest neural MTPP models amongst our baselines. The marked intensities are assumed to be constant between two events, and are parametrized from a MLP that takes as input a history embedding $\boldsymbol{h}_i$, i.e.

$$\lambda_k^*(t;\boldsymbol{\theta}) = \lambda_k^* = \mathrm{MLP}(\boldsymbol{h}_i) = \sigma_{S,k}\left(\boldsymbol{w}_k^\mathsf{T}(\sigma_R\big(\mathbf{W}\boldsymbol{h}_i + \boldsymbol{b}\big) + b_k\right), \tag{3.16}$$

where $\mathbf{W} \in \mathbb{R}^{d_{in} \times d_h}$, $\boldsymbol{b} \in \mathbb{R}^{d_{in}}$, $\boldsymbol{w}_k \in \mathbb{R}^{d_{in}}$ and $b_k \in R$ are mark-specific weights and biases, respectively. In (3.16), $\sigma_{S,k}$ is a mark-specific softplus activation function, defined as

$$\sigma_{S,k}(x) = \beta_k \log\left(1 + \exp\left(\frac{x}{\beta_k}\right)\right), \tag{3.17}$$

with $\beta_k \in \mathbb{R}_+$ ensuring R_1^λ in (2.19) (i.e. $\lambda_k^*(t) \geq 0$). The marked intensities being independent of time between two events, the compensators are given by

$$\Lambda_k^*(t;\boldsymbol{\theta}) = (t - t_{i-1})\lambda_k^*, \tag{3.18}$$

and the distribution of inter-arrival times is exponential with rate $\lambda^* = \sum_{k=1}^K \lambda_k^*$, i.e.

$$f^*(\tau;\boldsymbol{\theta}) = \lambda^* \exp(-\lambda^* \tau). \tag{3.19}$$

MLP/MC decoder. Although the EC decoder can capture arbitrary influence of past observations, it does not allow for the marked intensity functions to evolve between consecutive events. To circumvent this limitation, the MLP/MC decoder takes as input the concatenation of $\boldsymbol{h}_i$ and an encoding $\boldsymbol{l}^t$ of a query time $t \geq t_{i-1}$. Specifically, the marked intensities are given by

$$\lambda_k^*(t;\boldsymbol{\theta}) = \text{MLP}(\boldsymbol{h}_i, \boldsymbol{l}^t) = \lambda_k + \sigma_{S,k}\big(\boldsymbol{w}_k^\top(\sigma_R\big(\mathbf{W}[\boldsymbol{h}_i, \boldsymbol{l}^t] + \boldsymbol{b}\big) + b_k\big), \tag{3.20}$$

where $\mathbf{W} \in \mathbb{R}^{d_{in} \times (d_h + d_t)}$. In (3.20), $\lambda_k \in \mathbb{R}_+$ ensures that R_2^λ in (2.20) (i.e. $\lim\limits_{t\to\infty} \int_{t_{i-1}}^t \lambda_k^*(s)ds = \infty$) is met as $\lim\limits_{t\to\infty} \lambda_k t = \infty$.

However, the softplus activation function in (3.20) prevents the evaluation of $\Lambda_k^*(t;\boldsymbol{\theta})$ in closed form, requiring numerical integration techniques such as Monte Carlo to approximate its value (Press et al., 2007):

$$\tilde{\Lambda}_k^*(t;\boldsymbol{\theta}) \simeq \frac{t - t_{i-1}}{n_\upsilon} \sum_{j=1}^{n_\upsilon} \lambda_k^*(\upsilon_j;\boldsymbol{\theta}), \tag{3.21}$$

where $\upsilon_j \sim \text{Uniform}[t_{i-1}, t]$ and n_υ being the number of Monte Carlo samples.

SA/MC decoder. The marked intensities are parametrized from a similar set of equations as (3.13) and (3.12), at the major difference that the query vector is now constructed from $\boldsymbol{l}^t$, while the key and value matrices are build from $\boldsymbol{h}_i$. By doing so, we allow a query time t to attend to previous event representations in $\mathcal{H}_t$, i.e.

$$\boldsymbol{q}^{(h)}(t) = \mathbf{W}_Q^{(h)}\boldsymbol{l}^t, \quad \mathbf{K}_i^{(h)} = \mathbf{W}_K^{(h)}[\boldsymbol{h}_1, ..., \boldsymbol{h}_i], \quad \mathbf{V}_i^{(h)} = \mathbf{W}_V^{(h)}[\boldsymbol{h}_1, ..., \boldsymbol{h}_i], \tag{3.22}$$

$$\boldsymbol{a}(t) = \mathbf{W}_H \left[\bigoplus_{h=1}^H \text{SA}\left(\boldsymbol{q}^{(h)}(t), \mathbf{K}_i^{(h)}, \mathbf{V}_i^{(h)}\right)\right], \tag{3.23}$$

$$\lambda_k^*(t;\boldsymbol{\theta}) = \mu_k + \sigma_{S,k}\big(\boldsymbol{w}_k^\top \boldsymbol{a}(t) + b_k\big), \tag{3.24}$$

where $\mathbf{W}_Q^{(h)} \in \mathbb{R}^{d_q \times d_t}$, $\mathbf{W}_K^{(h)} \in \mathbb{R}^{d_q \times d_h}$ and $\mathbf{W}_V^{(h)} \in \mathbb{R}^{d_h \times d_z}$, $\mathbf{W}_H \in \mathbb{R}^{d_{in} \times H d_z}$, and $\boldsymbol{w}_k \in \mathbb{R}^{d_{in}}$, d_z being the output dimension of the attention mechanism. Similarly to the MLP/MC decoder, $\Lambda_k^*(t;\boldsymbol{\theta})$ must be approximated by numerical integration techniques, e.g. as in (3.21).

RMTPP decoder. Instead of specifying the marked intensities directly, the RMTPP decoder separately parametrizes the ground intensity $\lambda^*(t;\boldsymbol{\theta})$, and a PMF of marks $p^*(k;\boldsymbol{\theta})$ independent of time:

$$\lambda^*(t;\boldsymbol{\theta}) = \exp\Big(w^t(t - t_{i-1}) + (\boldsymbol{w}^h)^\top \boldsymbol{h}_i + b\Big), \tag{3.25}$$

$$p^*(k;\boldsymbol{\theta}) = \sigma_{\text{so}}\left(\mathbf{W}_2 \sigma_R\left(\mathbf{W}_1 \boldsymbol{h} + \boldsymbol{b}_1\right)\right) + \boldsymbol{b}_2\right), \tag{3.26}$$

where $w^t \in \mathbb{R}_+$, $\boldsymbol{w}^h \in \mathbb{R}^{d_h}$, $b \in \mathbb{R}$, $\mathbf{W}_1 \in \mathbb{R}^{d_1 \times d_h}$, $\mathbf{W}_2 \in \mathbb{R}^{K \times d_1}$, $\boldsymbol{b}_1 \in \mathbb{R}^{d_1}$, $\boldsymbol{b}_2 \in \mathbb{R}^K$, and σ_{so} is the softmax activation function. The exponential transformation in (3.25), along with the positivity of w^t, ensure that R_1^λ in (2.19) and R_2^λ in (2.20) are met. By assuming the distribution of marks to be independent of the time given the history of the process, the RMTPP decoder makes a strong simplifying assumption, which has been criticized in subsequent works (Enguehard et al., 2020). However, the expression of the ground intensity enables us to directly compute $\Lambda^*(t;\boldsymbol{\theta})$ in closed form:

$$\Lambda^*(t;\boldsymbol{\theta}) = \frac{1}{w^t}\Big(\exp\big(w^t(t - t_{i-1}) + (\boldsymbol{w}^h)^\top \boldsymbol{h}_i + b\big) - \exp\big((\boldsymbol{w}^h)^\top \boldsymbol{h}_i + b\big)\Big). \tag{3.27}$$

Moreover, as pointed out by Shchur et al. (2020a), the RMTPP decoder defines a Gompertz distribution (Wienke, 2010) on the inter-arrival times with shape $\gamma = \frac{\exp\big((\boldsymbol{w}^h)^\top \boldsymbol{h}_i + b\big)}{w^t}$ and scale $\beta = w^t$, i.e.

$$f^*(\tau;\boldsymbol{\theta}) = \beta\gamma \exp\left(\beta\tau - \gamma\exp(\beta\tau) + \gamma\right). \tag{3.28}$$

NH decoder. The marked intensities $\lambda_k^*(t)$ are parametrized using a single layer MLP that takes as input a history embedding $\boldsymbol{h}(t)$ that is allowed to vary between consecutive events, i.e:

$$\lambda_k^*(t;\boldsymbol{\theta}) = \sigma_{S,k}\Big((\boldsymbol{w}_k)^\top \boldsymbol{h}(t)\Big). \tag{3.29}$$

In contrast to a classical discrete-time LSTM, which would only update the history embedding $\boldsymbol{h}_i$ when the event at time t_i is observed, a continuous-time LSTM allows the history embedding to evolve with time for t $> t_{i-1}$, i.e.

$$\boldsymbol{h}(t) = \boldsymbol{o}_i \circ \Big(2\sigma_{\text{si}}\big(2\boldsymbol{c}(t)\big) - 1\Big), \tag{3.30}$$

$$\boldsymbol{c}(t) = \bar{\boldsymbol{c}}_i + (\boldsymbol{c}_i - \bar{\boldsymbol{c}}_i)\exp\big(-\boldsymbol{\delta}_i(t - t_{i-1})\big), \tag{3.31}$$

where σ_{si} is the sigmoïd activation function. In (3.30) and (3.31), the vectors $\boldsymbol{o}_i$, $\bar{\boldsymbol{c}}_i$, $\boldsymbol{c}_i$ and $\boldsymbol{\delta}_i$ are hidden states obtained from a modified set of discrete-time LSTM equations that updates their values each time a new event appears, i.e.

$$\boldsymbol{\delta}_i = \sigma_S(\mathbf{W}_d^k \boldsymbol{k}_{i-1} + \mathbf{W}_d^h \boldsymbol{h}_{i-1}(t_i) + \boldsymbol{b}_d), \tag{3.32}$$

$$\boldsymbol{q}_i = \sigma_{\text{si}}(\mathbf{W}_q^k \boldsymbol{k}_{i-1} + \mathbf{W}_q^h \boldsymbol{h}_{i-1}(t_i) + \boldsymbol{b}_q), \tag{3.33}$$

$$\boldsymbol{z}_i = 2\sigma_{\text{si}}(\mathbf{W}_z^k \boldsymbol{k}_{i-1} + \mathbf{W}_z^h \boldsymbol{h}_{i-1}(t_i) + \boldsymbol{b}_z) - 1, \tag{3.34}$$

$$\boldsymbol{o}_i = \sigma_{\text{si}}(\mathbf{W}_o^k \boldsymbol{k}_{i-1} + \mathbf{W}_o^h \boldsymbol{h}_{i-1}(t_i) + \boldsymbol{b}_o), \tag{3.35}$$

$$\boldsymbol{f}_i = \sigma_{\text{si}}(\mathbf{W}_f^k \boldsymbol{k}_{i-1} + \mathbf{W}_f^h \boldsymbol{h}_{i-1}(t_i) + \boldsymbol{b}_f), \tag{3.36}$$

$$\boldsymbol{c}_i = \boldsymbol{f}_i \circ \boldsymbol{c}(t_{i-1}) + \boldsymbol{q}_i \circ \boldsymbol{z}_i, \tag{3.37}$$

$$\bar{\boldsymbol{c}}_i = \boldsymbol{f}_i \circ \bar{\boldsymbol{c}}_{i-1} + \boldsymbol{q}_i \circ \boldsymbol{z}_i, \tag{3.38}$$

where $\boldsymbol{k}_{i-1} \in \{0,1\}^K$ is the one-hot encoding of mark k_{i-1}, $\boldsymbol{w}_k \in \mathbb{R}^{d_h}$, $\mathbf{W}_q^k$, $\mathbf{W}_f^k$, $\mathbf{W}_z^k$, $\mathbf{W}_o^k$, $\mathbf{W}_d^k \in \mathbb{R}^{d_h \times K}$, $\mathbf{W}_q^h$, $\mathbf{W}_f^h$, $\mathbf{W}_z^h$, $\mathbf{W}_o^h$, $\mathbf{W}_d^h \in \mathbb{R}^{d_h \times d_h}$, and $\boldsymbol{b_q}$, $\boldsymbol{b_f}$, $\boldsymbol{b}_z$, $\boldsymbol{b}_o$, $\boldsymbol{b}_d \in \mathbb{R}^{d_h}$. Finally, σ_S is the unmarked formulation of the softplus activation function, i.e. $\sigma_S(x) = \beta \log(1 + \exp(x/\beta))$ with $\beta \in \mathbb{R}_+$. Here also, one must rely on numerical integration techniques to estimate $\Lambda_k^*(t; \boldsymbol{\theta})$.

LNM decoder. The predictive PDF of inter-arrival times $f^*(\tau; \boldsymbol{\theta})$ is parametrized as a mixture of M log-normal distributions, where the parameters of the m^{th} mixture component are obtained via a hypernetwork, i.e. a network parametrizing the weights of another network (Ha et al., 2016). Specifically,

$$f^*(\tau; \boldsymbol{\theta}) = \sum_{m=1}^{M} p_m \frac{1}{\tau \sigma_m \sqrt{2\pi}} \exp\Big(-\frac{(\log \tau - \mu_m)^2}{2\sigma_m^2}\Big), \tag{3.39}$$

where $p_m = \sigma_{\text{so}}\big(\mathbf{W}_p \boldsymbol{h}_i + \boldsymbol{b}_p\big)_m$ corresponds to the probability that τ_i is generated by the m^{th} mixture component, while $\mu_m = (\mathbf{W}_\mu \boldsymbol{h}_i + \boldsymbol{b}_\mu)_m$, $\sigma_m = \exp(\mathbf{W}_\sigma \boldsymbol{h}_i + \boldsymbol{b}_\sigma)_m$ are the mean and standard deviation of the m^{th} mixture component, respectively. Finally, $\mathbf{W}_p, \mathbf{W}_\mu, \mathbf{W}_\sigma \in \mathbb{R}^{M \times d_h}$ and $\boldsymbol{b}_p, \boldsymbol{b}_\mu, \boldsymbol{b}_\sigma \in \mathbb{R}^M$, M being the number of mixture components. Defined similarly to equation (3.26), the predictive PMF of marks $p^*(k; \boldsymbol{\theta})$ is assumed to be conditionally independent of time given the history of the process. Although not available in closed-form, the cumulative distribution of a mixture of log-normals can be approximated with high precision (Abramowitz & Stegun, 1965) as

$$F^*(\tau; \boldsymbol{\theta}) = \sum_{m=1}^{M} \frac{1}{2}\Big[1 + \text{erf}\Big(\frac{\log \tau - \mu_m}{\sigma_m \sqrt{2}}\Big)\Big], \tag{3.40}$$

where $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-s^2} ds$ is the Gaussian error function. From $F^*(\tau; \boldsymbol{\theta})$, we can compute $\Lambda^*(t; \boldsymbol{\theta})$ as:

$$\Lambda^*(t; \boldsymbol{\theta}) = -\log \Big(1 - F^*(\tau; \boldsymbol{\theta})\Big). \tag{3.41}$$

FNN decoder. A major bottleneck of models parametrizing $\lambda_k^*(t; \boldsymbol{\theta})$ resides in the marked compensators not always being available in closed-form, thus requiring expensive numerical integration techniques. An elegant way of addressing this challenge is to directly parametrize $\Lambda_k^*(t; \boldsymbol{\theta})$, from which $\lambda_k^*(t; \boldsymbol{\theta})$ can be easily retrieved through differentiation. The original definition of FNN (Omi et al., 2019) involved parametrizing $\Lambda^*(t; \boldsymbol{\theta})$ using a MLP that operated on both the inter-arrival times and the history embeddings. In the presence of marks, this can be expressed as follows:

$$\Lambda_k^*(t; \boldsymbol{\theta}) = \sigma_{S,k}\big(\boldsymbol{w}_k^\top (\tanh\big(\boldsymbol{w}^t (t - t_{i-1}) + \mathbf{W}^h \boldsymbol{h}_i + \boldsymbol{b}\big) + b_k\big), \tag{3.42}$$

where each weight is constrained to be positive, ensuring condition R_1^Λ in (2.21) (i.e. $\Lambda_k^*(t) > 0$) and R_4^Λ in (2.24) (i.e. $d\Lambda_k^*(t)/dt \geq 0$). Specifically, $\boldsymbol{w}_k \in \mathbb{R}_+^{d_{in}}$, $\boldsymbol{w}^t \in \mathbb{R}_+^{d_{in}}$, $\mathbf{W}^h \in \mathbb{R}_+^{d_{in} \times d_h}$, $\boldsymbol{b} \in \mathbb{R}_+^{d_{in}}$ and $b_k \in \mathbb{R}_+$. However, as pointed out by Shchur et al. (2020a), equation (3.42) fails to satisfy R_2^Λ in (2.22) (i.e. $\Lambda_k^*(t_{i-1}) = 0$) as

$$\Lambda_k^*(t_{i-1}; \boldsymbol{\theta}) = \sigma_S\big(\boldsymbol{w}_k^\top (\tanh\big(\mathbf{W}^h \boldsymbol{h}_i + \boldsymbol{b}_1\big) + b_k\big) > 0, \tag{3.43}$$

In other terms, the original FNN model attributes non-zero probability mass to null inter-arrival times, i.e. $F^*(0; \boldsymbol{\theta}) = 1 - \exp\Big(-\sum_{k=1}^K \Lambda_k^*(t_{i-1}; \boldsymbol{\theta})\Big) > 0$. The original formulation also fails to satisfy R_3^Λ in (2.23) (i.e. $\lim\limits_{t \to \infty} \Lambda_k^*(t) = \infty$) due to the saturation of the tanh activation function:

$$\lim_{t \to \infty} \Lambda_k^*(t; \boldsymbol{\theta}) = \sigma_S\left(\sum_{d=1}^{d_{in}} w_{k,d} + b_k\right) < \infty. \tag{3.44}$$

To prevent both these shortcomings, Enguehard et al. (2020) proposed to replace the tanh activation function with a Gumbel-softplus activation:

$$\sigma_{GS,k}(x) = \big[1 - \big(1 + \alpha_k \exp(x)\big)^{-\frac{1}{\alpha_k}}\big]\big[1 + \sigma_{S,k}(x)\big], \tag{3.45}$$

with $\alpha_k \in \mathbb{R}_+$. The Gumbel-Softplus activation function being non-saturating (i.e. $\lim\limits_{x \to \infty} \sigma_{GS,k}(x) = \infty$), using it as a replacement for the tanh activation function in (3.42) effectively satisfies R_3^Λ. Finally, by defining $\Lambda_k^*(t; \boldsymbol{\theta})$ as

$$\Lambda_k^*(t; \boldsymbol{\theta}) = G_k^*(t; \boldsymbol{\theta}) - G_k^*(t_{i-1}; \boldsymbol{\theta}), \tag{3.46}$$

$$G_k^*(t;\boldsymbol{\theta}) = \sigma_{S,k}\big(\boldsymbol{w}_k^\mathsf{T}(\sigma_{GS,k}\big(\mathbf{W}^t\boldsymbol{l}^t + \mathbf{W}^h\boldsymbol{h}_i + \boldsymbol{b}\big) + b_k\big), \tag{3.47}$$

we now have $\Lambda_k^*(t_{i-1};\boldsymbol{\theta}) = 0$, satisfying R_2^Λ. The expression in (3.46) defines the generalized corrected version of FNN. Also, note that $G_k^*(t;\boldsymbol{\theta})$ takes now as input a general encoding $\boldsymbol{l}^t$ of a query time t with $\mathbf{W}^t \in \mathbb{R}^{d_{in}\times d_t}$. However, to ensure that R_4^Λ remains satisfied, $\boldsymbol{l}^t$ must be monotonic in its input. Therefore, when parametrizing a cumulative decoder, the temporal event encoding in (3.3) cannot be used, and the weights of the encoding in (3.4) must be constrained to be positive. From (3.46), $\lambda_k^*(t;\boldsymbol{\theta})$ can be finally retrieved through differentiation:

$$\lambda_k^*(t;\boldsymbol{\theta}) = \frac{d}{dt}\Lambda_k^*(t;\boldsymbol{\theta}). \tag{3.48}$$

SA/CM decoder. The SA/CM decoder shares a similar set of equations to (3.24) to parametrize $\Lambda_k^*(t;\boldsymbol{\theta})$. However, similarly to FNN, several modifications of the latter are required to ensure that the decoder meets the various constraints imposed by $\Lambda_k^*(t)$. First, the softmax activation is replaced by a sigmoid to satisfy R_4^Λ. Moreover, as

$$\frac{d^2\sigma_R(x)}{dx^2} = 0, \tag{3.49}$$

the ReLU activation is replaced by a Gumbel-softplus, which prevents

$$\frac{d^2\Lambda_k^*(t;\boldsymbol{\theta})}{dt^2} = \frac{d\lambda_k^*(t;\boldsymbol{\theta})}{dt} = 0. \tag{3.50}$$

In other terms, a cumulative model defined with a ReLU activation is equivalent to the EC decoder in (3.16) (Enguehard et al., 2020). Moreover, given the saturation of the sigmoid function, a term $\mu_k(t - t_{i-1})$ with $\lambda_k \in \mathbb{R}_+$ is introduced to satisfy R_4^Λ. Note that including this term is equivalent to adding a constant λ_k directly to $\lambda_k^*(t;\boldsymbol{\theta})$. All these pieces stitched together, the SA/CM decoder with a single attention head is given by

$$\Lambda_k^*(t;\boldsymbol{\theta}) = G_k^*(t;\boldsymbol{\theta}) - G_k^*(t_{i-1};\boldsymbol{\theta}), \tag{3.51}$$

$$G_k^*(t;\boldsymbol{\theta}) = \lambda_k(t - t_{i-1}) + \sigma_{S,k}\big(\boldsymbol{w}_k^\mathsf{T}(\mathrm{SA}\big(\boldsymbol{q}(t),\mathbf{K}_i,\mathbf{V}_i\big) + b_k\big), \tag{3.52}$$

$$\mathrm{SA}(\boldsymbol{q}(t),\mathbf{K}_i,\mathbf{V}_i) = \mathbf{W}_2\sigma_{GS,k}\big(\mathbf{W}_1\bar{\boldsymbol{z}}_i^\mathsf{T} + \boldsymbol{b}_1\big) + \boldsymbol{b}_2, \tag{3.53}$$

$$\bar{\boldsymbol{z}}_i = \sigma_{\mathrm{si}}\Big(\frac{\boldsymbol{q}(t)^\mathsf{T}\mathbf{K}_i}{\sqrt{d_q}}\Big)\mathbf{V}_i^\mathsf{T},$$

where $\mathbf{W}_1$, $\mathbf{W}_2 \in \mathbb{R}_+^{d_z\times d_z}$, $\boldsymbol{b}_1, \boldsymbol{b}_2 \in \mathbb{R}_+^{d_z}$, and where each entry of $\mathbf{W}_K$, $\mathbf{W}_V$, $\mathbf{W}_Q$, $\boldsymbol{w}_k$ and b_k is now constrained to be positive.

The query $\boldsymbol{q}(t)$, keys $\mathbf{K}_i$, and values $\mathbf{V}_i$ are given in (3.22), and the model can be extended to multi-head attention similarly to (3.23). Similar to FNN, $\lambda_k^*(t; \boldsymbol{\theta})$ can be retrieved through differentiation.

3.2. Related Work

Neural Marked Temporal Point Processes. Simple parametric forms of MTPP models, such as the self-exciting Hawkes process (Hawkes, 1971), or the self-correcting process (Isham & Westcott, 1979), rely on strong modeling assumptions that inherently limit their flexibility. To capture complex dynamics of real-world patterns, the Machine Learning (ML) community eventually turned to the latest advances in neural modeling and proposed a novel class of MTPP models based on various neural-network architectures. Precursor to the field, Du et al. (2016) proposed to encode the history using a vanilla RNN, an idea that has been largely adopted and improved in subsequent works (Mei & Eisner, 2017; Xiao et al., 2017b,a; Li et al., 2018; Guo et al., 2018b; Türkmen et al., 2019; Zhu et al., 2020; Soen et al., 2021). Inspired by their huge success as sequence encoders in Natural Language Processing (NLP), another line of work relies instead on self-attention mechanisms to encode the history (Zuo et al., 2020; Zhang et al., 2020; Zhu et al., 2021; Yang et al., 2022; Li et al., 2023; Yuan et al., 2023). Most of these works focus on modeling the trajectories of future events through the (marked) intensity functions. However, parametrizations of $\lambda_k^*(t)$ often come at the cost of being unable to evaluate the log-likelihood in closed-form, requiring expensive Monte Carlo integration. This consideration motivated the design of compensator-based approaches that parametrize $\Lambda_k^*(t)$ (Omi et al., 2019; Enguehard et al., 2020), from which $\lambda_k^*(t)$ can be retrieved through differentiation. Instead of parametrizing the (cumulative) marked intensities, Xiao et al. (2017b) directly modeled the PDF of inter-arrival times with a Gaussian distribution, while Shchur et al. (2020a) relies on the flexibility of a mixture of log-normal distributions. However, this work supposes conditional independence of arrival times and marks, an assumption later alleviated in Waghmare et al. (2022). Finally, Ben Taieb (2022) proposed to learn from the CRPS objective a recurrent neural spline parametrization of the conditional quantile function, enabling analytical sampling of inter-arrival times. Other alternatives to NLL optimization techniques for training neural MTPP models include variational objectives (Boyd et al., 2020), reinforcement learning (Upadhyay et al., 2018; Li et al., 2018), noise contrastive estimation (Guo et al., 2018b; Mei et al., 2020), adversarial training (Xiao et al., 2017a; Yan et al., 2018; Zhu et al., 2020) and score-matching (Sahani et al., 2016;

Li et al., 2023). Although any MTPP model could in theory be trained using these optimization techniques, we focus our analysis on the NLL objective. For surveys on (neural) MTPP modeling, refer to Shchur et al. (2021b) and Yan (2019a).

Experimental studies. Most papers in the neural MTPP literature mainly focused on proposing methodological improvements inspired by contemporary advances in deep learning techniques. While these contributions are paramount to driving future progress in the field, few studies have been carried out to identify sources of empirical gains across neural architectures and highlight future interesting research directions. The work of Lin et al. (2021) is the closest to our experimental study. While they also compared the impact of various history encoders and decoders, they did not discuss the influence of different event encoding mechanisms. However, we found that specific choices for this architectural component can lead to drastic performance gains. Additionally, they did not include simple parametric models to their baselines, such as the Hawkes decoder. Considering these models in an experimental study allows however to fairly evaluate the true gains brought by neural architectures. We also assess the calibration of neural and non-neural MTPP models on the distribution of arrival times and marks, and our experiments are conducted across a wider range of real-world datasets. Finally, our results are supported by rigorous statistical tests. Lin et al. (2022) also conducted empirical comparisons in the context of neural MTPP models but their attention was mainly focused on deep generative models.

3.3. Experimental Study

We carry out a large-scale experimental study to assess the predictive accuracy of state-of-the-art neural MTPP models on 15 real-world event sequence datasets and a synthetic Hawkes dataset in a carefully designed unified setup. In particular, we study the influence of each major architectural component, including event encoding, history encoder, and decoder in estimating the time and mark predictive distributions. For completeness, we also include classical parametric MTPP models into our set of baselines, such as the Hawkes and Poisson processes. The next section summarizes the various architectural components that we have considered in our experimental study. Section 3.3.2 presents the datasets including summary statistics and pre-processing steps. Section 3.3.3 describes our experimental setup. Finally, Section 3.3.4 presents the evaluation metrics and statistical tools used to assess the accuracy of the considered models, and Section 3.3.5 describes the procedure employed to aggregate the results across multiple datasets.

3.3.1 Models

To ease the understanding of the following sections, a brief summary of the different architectures considered in our experiments is presented below. Table 3.1 summarizes the correspondence between all variations of event encoding, history encoder, and decoder, and their mathematical expressions.

Event encoding mechanisms. For the event encoding mechanism, we consider the raw inter-arrival times τ (Times Only (TO)), the logarithms of inter-arrival times (Log-Times Only (LTO)), a temporal encoding of arrival times (Temporal (TEM)), and a learnable encoding of inter-arrival times (Learnable (LE)). Additionally, we include all their variants resulting from the concatenation of the mark embeddings $\boldsymbol{l}^k$, i.e. Concatenate (CONCAT) with TO and $\boldsymbol{l}^k$, Log-Concatenate (LCONCAT) with LTO and $\boldsymbol{l}^k$, Temporal With Labels (TEMWL) with TEM and $\boldsymbol{l}^k$, and Learnable With Labels (LEWL) with LE and $\boldsymbol{l}^k$.

History encoders. To encode the history of the process, we employ the GRU and SA mechanism presented in Section 3.1.2. Additionally, we consider a Constant (CONS) history encoder that systematically outputs $\boldsymbol{h}_i = \mathbf{1}_{d_h}$. Note that a decoder equipped with this encoder parametrizes a function independent of the history and reduces essentially to a IPP—a process whose marked intensity functions are independent of the history, as in (2.31).

Decoders. The decoders considered in this experimental study can be classified on the basis of the function that they parametrize, as well as on the assumptions they make regarding the distribution of inter-arrival times and marks. As described in Section 3.1.3, decoders that parametrize $\lambda_k^*(t;\boldsymbol{\theta})$ are the EC , MLP/MC , SA/MC and NH decoders. In this category, we also consider as classical baselines a Hawkes decoder with exponential kernels, as well as a simple Poisson decoder, whose marked intensities are parametrized by (3.1) and (2.29), respectively. On the other hand, RMTPP separately parametrizes the ground intensity $\lambda^*(t;\boldsymbol{\theta})$ and a time-independent PMF of marks $p^*(k;\boldsymbol{\theta})$, while LNM and its single mixture version, LN, separately parametrize the PDF of inter-arrival times $f^*(\tau;\boldsymbol{\theta})$ and a time-independent PMF of marks. Recall that for LN, LNM, and RMTPP, (inter-)arrival times and marks are assumed to be conditionally independent given the history. Finally, FNN and SA/CM directly parametrize $\Lambda_k^*(t;\boldsymbol{\theta})$.

Table 3.1: Summary of the various architectural components considered in the comparative study. For clarity, the dependence on $\boldsymbol{\theta}$ is omitted.

Component	**Name**	**Acronym**	**Parametrization**
Event encoder	Times	TO	$\boldsymbol{l}_i = \tau_i$
	Log-times	LTO	$\boldsymbol{l}_i = \log\, \tau_i$
	Concatenate	CONCAT	$\boldsymbol{l}_i^t = \tau_i$, $\boldsymbol{l}_i^k$ as in (3.5), $\boldsymbol{l}_i$ as in (3.2)
	Log-concatenate	LCONCAT	$\boldsymbol{l}_i^t = \log\, \tau_i$, $\boldsymbol{l}_i^k$ as in (3.5, $\boldsymbol{l}_i$ as in (3.2
	Temporal	TEM	$\boldsymbol{l}_i$ as in (3.3)
	Temporal with labels	TEMWL	$\boldsymbol{l}_i^t$ as in (3.3), $\boldsymbol{l}_i^k$ as in (3.5), $\boldsymbol{l}_i$ as in (3.2)
	Learnable	LE	$\boldsymbol{l}_i$ as in (3.4)
	Learnable with labels	LEWL	$\boldsymbol{l}_i^t$ as in (3.4), $\boldsymbol{l}_i^k$ as in (3.5), $\boldsymbol{l}_i$ as in (3.2)
History encoder	GRU	GRU	$\boldsymbol{h}_i = \mathrm{GRU}(\boldsymbol{l}_1, ..., \boldsymbol{l}_{i-1})$ as in (3.6)
	Self-attention	SA	$\boldsymbol{h}_i$ as in (3.12)
	Constant	CONS	$\boldsymbol{h}_i = \mathbf{1}_{d_h}$

Component	**Name**	**Acronym**	**Parametrization**	**Closed-form MLE**
Decoder	Exponential constant	EC	λ_k^* as in (3.16)	✓
	MLP	MLP/MC	$\lambda_k^*(t)$ as in (3.20)	✗
	FullyNN	FNN	$\Lambda_k^*(t)$ as in (3.46)	✓
	LogNormMix	LNM	$f^*(\tau)$ as in (3.39), $p^*(k)$ in (3.26)	✓
	LogNorm	LN	$f^*(\tau)$ as in (3.39) with $M = 1$, $p^*(k)$ in (3.26)	✓
	RMTPP	RMTPP	$\lambda^*(t)$ as in (3.25), $p^*(k)$ in (3.26)	✓
	Neural Hawkes	NH	$\lambda_k^*(t)$ as in (3.29)	✗
	Self-attention	SA/MC	$\lambda_k^*(t)$ as in (3.24)	✗
	Cumulative Self-attention	SA/CM	$\Lambda_k^*(t)$ as in (3.52)	✓
	Hawkes	Hawkes	$\lambda_k^*(t)$ as in (3.1)	✓
	Poisson	Poisson	$\lambda_k^*(t)$ as in (2.29)	✓

We define a model as a specific choice of event encoding mechanism, history encoder, and decoder, e.g. GRU-MLP/MC-TO corresponds to a model using the GRU encoder to build $\boldsymbol{h}_i$, the MLP decoder to parametrizes $\lambda_k^*(t)$, and where the events are encoded using the TO event encoding. While in most cases, any variation of a component can be seamlessly associated with any other variation of other components, some combinations are either impossible or meaningless. Indeed, all cumulative decoders (SA/CM and FNN) cannot be trained with the TEM or TEMWL event encodings, as both would violate the monotonicity constraint of $\Lambda_k^*(t; \boldsymbol{\theta})$. Moreover, as all event encodings are irrelevant with respect to the CONS history encoder, all CONS-EC models are equivalent. For the NH decoder, we stick to the original model definition, as it can be hardly disentangled into different components. Although not required to define a valid parametrization of a MTPP, we also consider for completeness the setting where a constant baseline intensity term $\lambda_k \geq 0$ (B) is added to $\lambda_k^*(t; \boldsymbol{\theta})$ of the EC and RMTPP decoders. A complete list of the considered combinations is given in Table C.7.

We further classify the different models into three categories: *parametric*, *semi-parametric*, and *non-parametric*. Parametric MTPP models include classical (i.e. non-neural) architectures, characterized by strong modeling assumptions and hence, low flexibility. We consider the Hawkes and Poisson decoders as parametric baselines. On the other hand, semi-parametric models include architectures that still make assumptions regarding the distribution of inter-arrival times, but their parameters are obtained from the output of a neural network. All models equipped with LNM, LN, RMTPP, or EC decoders are deemed semi-parametric. All remaining baselines, i.e. FNN, MLP, SA/MC, SA/CM and NH are considered non-parametric models.

3.3.2 Datasets

Real-world datasets. A total of 15 real-world datasets containing sequences of various lengths are used in our experiments, among which 7 possess marked events. A brief description is presented below, while their general statistics are summarized in Tables 3.2 and 3.3.

- **Marked Datasets**
  - **LastFM** [1] (Hidasi & Tikk, 2012): This dataset comprises records of people listening to songs over time. Each sequence relates to a user, and each mark corresponds to the artist of the song.
  - **MOOC** [1] (Kumar et al., 2019): This dataset captures the activities of students on a Massive Open Online Course (MOOC) platform. A sequence corresponds to a student, and the mark refers to the type of activity carried out by the student, e.g. watching a video or answering a quiz.
  - **Wikipedia** [1] (Kumar et al., 2019) : Contains records of edits made to Wikipedia pages in the course of a month. Each sequence corresponds to a given page, and the marks relate to specific editors.
  - **MIMIC2** [2] (Du et al., 2016) : EHR of patients in an intensive care units for seven years. A sequence corresponds to a patient, and the marks are the types of diseases.
  - **Github** [3] (Trivedi et al., 2019) : This dataset captures the activity records of public account owners on the open-source plat-

[1]https://github.com/srijankr/jodie/
[2]https://github.com/babylonhealth/neuralTPPs
[3]https://github.com/uoguelph-mlrg/LDG

form GitHub, covering the period from January 2013 to December 2013. Each sequence corresponds to an account, and the marks describe the types of action performed, i.e. "Watch", "Star", "Fork", "Push", "Issue", "Comment Issue", "Pull Request", "Commit".

- **Stack Overflow** [5] (Du et al., 2016) : Contains records of the times users received a badge on the question-answering platform Stack Overflow between 2012 and 2014. Each sequence corresponds to a user, and the marks are the types of badges received, e.g. "Stellar Question", "Guru", "Great Answer".
- **Retweets** [2] (Zhao et al., 2015) : This dataset comprises streams of retweet events following the creation of an original tweet. Each sequence corresponds to a tweet, and marks refer to the category to which the retweeter belongs based off his/her popularity, i.e. small, medium, and large number of followers.

• **Unmarked Datasets** (Shchur et al., 2020a,b)

- **Twitter** [4]: This dataset contains records of tweets made over several years.
- **PUBG** [4]: Records of players' deaths in the online game PUBG. Each sequence corresponds to a game, and a timestamp refers to the death of a given player.
- **Yelp Airport** [4]: This dataset contains customers check-in times to 319 businesses on the platform Yelp at the McCarran International Airport. Each sequence corresponds to a business.
- **Yelp Mississauga** [4]: Similar to the Yelp Airport dataset, but for 319 businesses in the city of Mississauga.
- **Yelp Toronto** [5]: This dataset contains sequences of reviews made by customers on the platform Yelp for 300 restaurants in the city of Toronto. Each sequence corresponds to a restaurant.
- **Reddit Comments** [4]: Records of comments in reply to Reddit discussion threads within 24 hours of the original post submission. The data is recorded between 2018 and 2020, and each sequence corresponds to a discussion thread.

[4]https://github.com/shchur/triangular-tpp
[5]https://github.com/shchur/ifl-tpp

Table 3.2: Marked datasets' statistics after pre-processing. Mean Sequence Length (MSL) corresponds to the average number of events per sequence.

	Sequences	**Events**	**MSL**	**Max Len.**	**Min Len.**	**Marks**
Wikipedia	590	30472	51.6	1163	2	50
MOOC	7047	351160	49.8	416	2	50
LastFM	856	193441	226.0	6396	2	50
MIMIC2	599	1812	3.0	32	2	43
Github	173	20657	119.4	4698	3	8
Stack Overflow	7959	569688	71.6	735	40	22
Retweets	24000	2610102	108.8	264	50	3

- **Reddit Submissions** [4]: This dataset comprises records of submissions made to a political sub-Reddit between 2017 and 2020. Each sequence corresponds to a 24 hours window.

- **Taxi** [4]: Contains the records of taxi pick-ups in the South of Manhattan. Each sequence corresponds to a taxi, and timestamps refer to the times at which passengers were taken on board.

Pre-processing. Some marked datasets, such as Wikipedia and LastFM, originally presented a very large amount of marks, whose distributions turn out to be highly spread across their respective domains. This observation raises two issues: (1) Some marks are therefore highly under-represented, rendering the task of learning their underlying distribution difficult, and (2), the GPU memory requirements can increase substantially with the number of marks, which is even more exacerbated when Monte Carlo samples need to be drawn. We provide more details regarding the time and space complexity of the different models later in Section 3.3. Hence, with the incentive to avoid either of these two bottlenecks, each marked dataset is filtered to only contain events belonging to the 50 most represented marks. The resulting sequences containing less than two events are further removed from the dataset, which makes the number of distinct marks in MIMIC2 drop from 75 to 43. Finally, to avoid numerical instabilities, the arrival times of events are scaled in the interval [0,10]. Specifically, we compute $t_{i,\text{scaled}} = 10t_i/t_{\max}$, where $t_{\max}$ is the largest observed timestamp in the dataset. Unmarked datasets do not go through any processing steps, at the exception of the scaling and removal of sequences containing less than two events.

Table 3.3: Unmarked datasets' statistics after pre-processing. MSL corresponds to the average number of events per sequence.

	Sequences	**Events**	**MSL**	**Max Len.**	**Min Len.**
Reddit Submissions	1094	1235128	1129.0	2658	362
Reddit Comments	1355	400933	295.9	2137	4
Taxi	182	17904	98.4	140	12
Twitter	1804	29862	16.6	169	2
Yelp Toronto	300	215146	717.2	2868	424
Yelp Airport	319	9716	30.5	55	9
Yelp Mississauga	319	17621	55.2	107	3
PUBG	3001	229703	76.5	97	26

As observed in Tables 3.2 and 3.3, the considered datasets are relatively diverse in their characteristics. Indeed, some datasets, such as Yelp Toronto or LastFM, possess a relatively short number of sequences with high MSL, while others, such as Twitter or MOOC, display the exact opposite characteristics. Figure 3.2 shows the distributions of the inter-arrival times logarithms across all sequences for all real-world datasets, as well as the mark distribution across all sequences for marked datasets. As observed, the distribution of the pooled (log) inter-arrival times differ significantly from one dataset to the other, in some cases presenting characteristics such as multi-modality (LastFM, MOOC, Wikipedia, Yelp Toronto, PUBG) or large variance (MOOC, Wikipedia, Github, Yelp Toronto). The distribution of pooled marks also shows different characteristics. While the marks look uniformly spread across their domains on LastFM, MOOC, and Wikipedia, their distribution appears sharper on Github, MIMIC2, Stack Overflow, and Retweets. Such diversity should be empirically beneficial, as it would allow us to assess the predictive accuracy across a wider range of real-world applications. In Figures C.3, C.4 and C.5 of Appendix C.5, we show the distribution of inter-arrival times and marks for some randomly sampled sequences in each dataset.

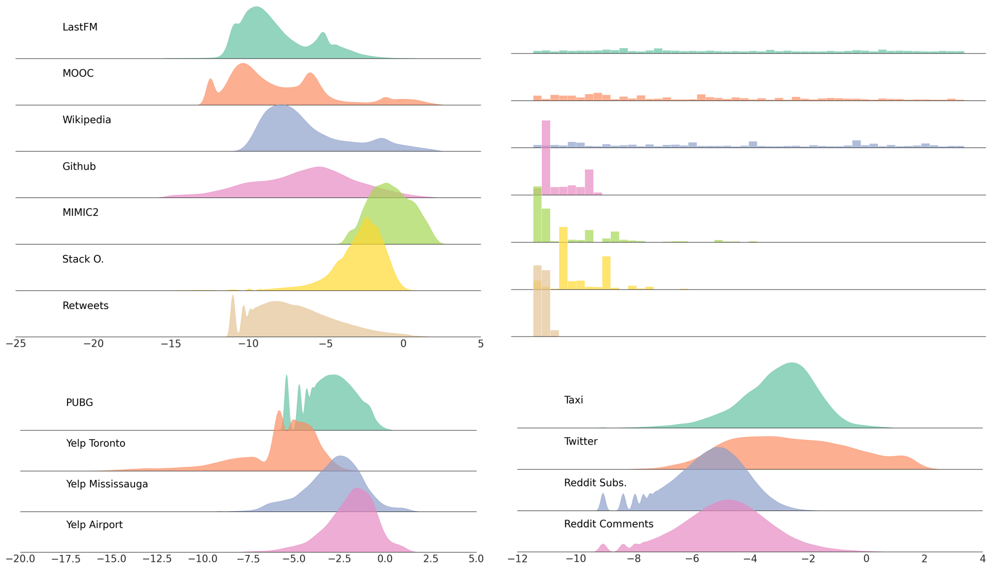


Figure 3.2: Distribution of log τ (top left) and mark distribution (top right) for marked datasets, after pre-processing. As all rows share a common x-axis, we show the distribution of log τ instead of τ to improve readability. For unmarked datasets, only the distribution of log τ (bottom) is reported.

Synthetic datasets. We generate a synthetic dataset from the multidimensional Hawkes process with exponential kernels as in (2.42)-(2.43) with the following parameter values:

$$\boldsymbol{\lambda} = \begin{pmatrix} 0.2 \\ 0.6 \\ 0.1 \\ 0.7 \\ 0.9 \end{pmatrix} \boldsymbol{\gamma} = \begin{pmatrix} 0.25 & 0.13 & 0.13 & 0.13 & 0.13 \\ 0.13 & 0.35 & 0.13 & 0.13 & 0.13 \\ 0.13 & 0.13 & 0.2 & 0.13 & 0.13 \\ 0.13 & 0.13 & 0.13 & 0.3 & 0.13 \\ 0.13 & 0.13 & 0.13 & 0.13 & 0.25 \end{pmatrix} \boldsymbol{\beta} = \begin{pmatrix} 4.1 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 2.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 6.2 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 4.9 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 4.1 \end{pmatrix},$$

where the matrix $\boldsymbol{\gamma}$ was scaled to have a spectral radius of approximately 0.8, guaranteeing stationarity of the process (Bacry et al., 2020). The process essentially corresponds to a marked process with $K = 5$ marks, from which we simulate 5 distinct datasets of 1000 sequences using the library *tick* (Bacry et al., 2018).

3.3.3 Experimental Setup

For each real-world and synthetic dataset, we randomly split the sequences $\mathcal{S}$ into non-overlapping train/validation/test splits of sizes 60%/20%/20%, respectively. This yields $\mathcal{S}_{\text{train}}$, $\mathcal{S}_{\text{val}}$, and $\mathcal{S}_{\text{test}}$ with $\mathcal{S} = \mathcal{S}_{\text{train}} \cup \mathcal{S}_{\text{val}} \cup \mathcal{S}_{\text{test}}$. The models are trained to minimize the NLL objective in (2.49) on the training sequences using mini-batch gradient descent (Ruder, 2017). At each epoch, the NLL is also evaluated on the validation sequences $\mathcal{S}_{\text{val}}$, and the training procedure is interrupted if the number of epochs reaches 500, or if no improvement is observed for 20 consecutive epochs. In the latter case, the model's parameters are reverted to their state of lowest validation loss. For all models, optimization is carried out using the Adam optimizer (Kingma & Ba, 2014) with an initial learning rate of 10^{-3}. If no improvement in validation loss is observed for 5 consecutive epochs, the learning rate is divided by a factor of 2, and training continues. We repeat this protocol 5 times using different random train/validation/test splits.

We conduct experiments with different values of event encoding dimension, specifically $\{4, 8, 16, 32\}$, as well as varying the number of hidden units for MLP layers in $\{8, 16, 32\}$. For models utilizing GRU or SA mechanisms (at the encoder or decoder stage), we explore different numbers of hidden units in $\{8, 16, 32, 64\}$, and consider one or two layers. In the case of SA encoders and decoders, we consider one or two heads, and we employ layer normalization which has been shown to aid convergence of Transformer architectures (Xiong et al., 2020). Additionally, the number of mixtures in LNM is explored in $\{8, 16, 32, 64\}$. To determine the model's hyperparameters, we follow the following procedure. For each model and dataset split, we randomly select five hyperparameter configurations. The model is trained using each of these configurations, and we select the configuration with the lowest validation loss. These five best configurations (which may differ depending on the split) are then evaluated on the respective test set. Finally, we report the average test metrics as described in the following section.

3.3.4 Evaluation Metrics

Once the MTPP model is trained, yielding the parameters $\hat{\boldsymbol{\theta}}$, we consider a range of metrics to assess the performance of the model in terms of the time and mark prediction tasks, which involve estimating $f^*(\tau)$ and $p^*(k|\tau)$, respectively. Among other metrics, a common practice in the MTPP literature consists in reporting the NLL in (2.49) averaged over all sequences in $\mathcal{S}_{\text{test}}$.

However, reporting a single NLL metric encompasses the contributions of time and mark predictive distributions, effectively obscuring how a model actually performs on fitting the two distributions separately. Consequently, as discussed in Section 2.1.5, we split the NLL in (2.49) into $\mathcal{L}_T$ and $\mathcal{L}_M$ terms[6], highlighting the separate contributions of each of these two terms to the total NLL metric:

$$\begin{aligned}\mathcal{L}(\hat{\boldsymbol{\theta}};\mathcal{S}_{\text{test}}) = & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log f^*(\tau_{l,i};\hat{\boldsymbol{\theta}}) - \log(1-F^*(T;\hat{\boldsymbol{\theta}}))\right]}_{\mathcal{L}_T(\hat{\boldsymbol{\theta}},\mathcal{S}_{\text{test}})} \\ & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log p^*(k_{l,i}|\tau_{l,i};\hat{\boldsymbol{\theta}})\right]}_{\mathcal{L}_M(\hat{\boldsymbol{\theta}},\mathcal{S}_{\text{test}})}.\end{aligned} \tag{3.54}$$

On the other hand, a model's ability to predict the mark of the next event is measured by means of the mark averaged **F1-score** in (2.72). Recall that the mark of the next event can be predicted as

$$\tilde{k} = \underset{k\in\mathbb{K}}{\operatorname{argmax}}\ p^*(k|t;\hat{\boldsymbol{\theta}}) = \underset{k\in\mathbb{K}}{\operatorname{argmax}}\ \lambda_k^*(t;\hat{\boldsymbol{\theta}}), \tag{3.55}$$

and the mark averaged F1-score is obtained as

$$\text{F1-score} = \frac{1}{K}\sum_{k'=1}^{K} 2\times\frac{\mathrm{P}_{k'}\times\mathrm{R}_{k'}}{\mathrm{P}_{k'}+\mathrm{R}_{k'}}, \tag{3.56}$$

where, noting $S = |\mathcal{S}_{\text{test}}|$,

$$\mathrm{P}_{k'} = \frac{1}{S}\sum_{l=1}^{S}\frac{1}{n_l}\sum_{i=1}^{n_l}\frac{\mathbb{1}\left(\tilde{k}_{i,l}=k', k_{i,l}=k'\right)}{\left[\mathbb{1}\left(\tilde{k}_{i,l}=k', k_{i,l}=k'\right)+\mathbb{1}\left(\tilde{k}_{i,l}=k', k_{i,l}\neq k'\right)\right]}, \tag{3.57}$$

$$\mathrm{R}_{k'} = \frac{1}{S}\sum_{l=1}^{S}\frac{1}{n_l}\sum_{i=1}^{n_l}\frac{\mathbb{1}\left(\tilde{k}_{i,l}=k', k_{i,l}=k'\right)}{\left[\mathbb{1}\left(\tilde{k}_{i,l}=k', k_{i,l}=k'\right)+\mathbb{1}\left(\tilde{k}_{i,l}\neq k', k_{i,l}= k'\right)\right]}. \tag{3.58}$$

For a mark $k'\in\mathbb{K}$, the term $\mathbb{1}\left(\tilde{k}_{i,l}=k', k_{i,l}=k'\right)$ is equal to 1 when the predicted and observed mark at time t_i are both equal to k', which corresponds

[6]For the remainder of the discussion, we omit the explicit dependency of $\mathcal{L}_T$ and $\mathcal{L}_M$ on $\hat{\boldsymbol{\theta}}$ and $\mathcal{S}_{\text{test}}$.

to a *true positive* for the k'-th mark. In this context, the $\mathbb{1}\left(\tilde{k}_{i,l} = k', k_{i,l} \neq k'\right)$ is equal to 1 if the predicted mark at time t_i is equal to k', while the observed mark is different to k', which accounts for *false positive.* Finally, the term $\mathbb{1}\left(\tilde{k}_{i,l} \neq k', k_{i,l} = k'\right)$ is equal to 1 if the observed mark at time t_i is equal to k', while the model predicted predicted a different mark than k', i.e. a *false negative* for the k'-th mark.

Calibration. While achieving a low out-of-sample NLL score is crucial, it is also important to ensure that the predictive distributions are well-calibrated. As discussed in Section 2.2.1, calibration refers to the statistical consistency between the predicted distribution and the observed outcomes (Gneiting et al., 2007). Formally, recall that a model that outputs a predictive CDF $F^*(\tau;\hat{\boldsymbol{\theta}})$ (which may be retrieved from $f^*(\tau;\hat{\boldsymbol{\theta}})$, $\lambda^*(t;\hat{\boldsymbol{\theta}})$ or $\Lambda^*(t;\hat{\boldsymbol{\theta}})$) is probabilistically calibrated if (Dawid, 1984; Kuleshov et al., 2018):

$$\mathbb{P}\big(F^*(\tau;\hat{\boldsymbol{\theta}}) \leq p\big) = p, \quad \forall p \in [0,1], \tag{3.59}$$

where the probability is taken over τ and $\boldsymbol{h}$. If the predictive distributions $F^*(\tau;\boldsymbol{\theta})$ are well-calibrated, it means that 90% prediction intervals for inter-arrival times would, on average, contain the observed inter-arrival times 90% of the time. Similarly, in the context of mark prediction, top label calibration is defined as (Guo et al., 2017)

$$\mathbb{P}\left(k = \operatorname*{argmax}_{k\in\mathbb{K}} p^*(k|\tau;\hat{\boldsymbol{\theta}}) \mid \max_{k\in\mathbb{K}} p^*(k|\tau;\hat{\boldsymbol{\theta}}) = p\right) = p \quad \forall p \in [0,1], \tag{3.60}$$

where the probability is also taken over all $\boldsymbol{h}$. Intuitively, if $p^*(k|\tau;\hat{\boldsymbol{\theta}})$ is well-calibrated, it means that 90% of predictions made with a confidence level of 0.9 should match, on average, the observed mark 90% of the time.

We measure the probabilistic calibration of the time predictive distribution using the PCE, defined as (Dheur & Ben Taieb, 2023):

$$\mathrm{PCE} = \frac{1}{M}\sum_{m=1}^{M}\left|\sum_{i=1}^{n}\frac{\mathbb{1}[F^*(\tau_i;\hat{\boldsymbol{\theta}}) \leq p_m]}{n} - p_m\right|, \tag{3.61}$$

where $p_m = \frac{m}{M} \in [0,1]$ are specified probability levels, $n = \sum_{l=1}^{L} n_l$, and where we set $M = 50$. For the predictive PMF of marks, we report the ECE (Naeini et al., 2015) defined as:

$$\mathrm{ECE} = \frac{1}{M}\sum_{j=1}^{M}\Big|\mathrm{acc}(b_m) - \mathrm{conf}(b_m)\Big|, \tag{3.62}$$

where $\text{acc}(b_m)$ and $\text{conf}(b_m)$ represent the accuracy and average confidence, respectively, within the m^{th} bin b_m, out of a total of M bins. Refer to Section 2.2.1 for further details. We set $M = 10$, and lower PCE and ECE are better.

Reliability diagrams. As discussed in Section 2.2.1, a disadvantage with PCE and ECE metrics is that information regarding the calibration error at individual probability levels $p_1, ..., p_M$, or within individual bins $b_1, ..., b_M$, is lost. Reliability diagrams are visual tools that can be used to assess the probabilistic calibration of a model at a fine-grained level for both continuous and discrete distributions. For the time predictive distributions, a reliability diagram is obtained by plotting the empirical CDF $\sum_{i=1}^{n} \frac{\mathbb{1}[F^*(\tau_i;\hat{\boldsymbol{\theta}}) \leq p_m]}{n}$ in (3.61) against all probability levels p_m. For the mark predictive distributions, it is obtained by plotting $\text{acc}(b_m)$ against $\text{conf}(b_m)$ for all b_m. In both cases, a probabilistically calibrated model should align with the diagonal line, and any significant deviation from it corresponds to miscalibration.

Statistical comparisons. We further conduct statistical pairwise comparisons between all decoders, for each metric separately. Following Demšar (2006) and García & Herrera (2008), we consider the setting in which a statistical sample corresponds to the performance of a decoder on a metric of interest for a given dataset. Hence, for a given metric, this implies that each decoder is associated to exactly one sample per dataset. First, for each metric separately, Friedman test (Friedman, 1937, 1940) is employed to find at least one statistically significant difference among all decoders. If the null hypothesis is rejected at the $\alpha = 0.05$ significance level, we proceed with comparing each decoder against each other, using Holm's post-hoc test (Holm, 1979) to account for multiple hypothesis testing. The outcome of the pairwise comparisons is displayed on Critical Distance (CD) diagrams (Demšar, 2006), which show the average rank of a model on a metric of interest across all datasets, as well as groups of models that are not statistically different from one another at a given significance level. For additional details on Friedman and Holm's post-hoc tests, we refer the reader to Appendix C.1.

3.3.5 Results Aggregation

Given the high number of variations per model component, the number of possible combinations renders the comparison of all individual models across every dataset unmanageable. To overcome this challenge, we aggregate the model results across all datasets for each metric separately as follows. When comparing different event encodings, we first group all models that are equipped with a specific decoder variation (e.g. MLP/MC). Then, among this decoder group, we further group all models by event encoding variations (e.g. TO-Any history encoder-MLP/MC). We then compute the average performance of that event encoding-decoder group on a given dataset with respect to each metric. Finally, for each metric, we rank this encoding-decoder group against other encoding-decoder groups based on their average score on the same dataset. We apply this operation for each dataset separately and report the average and median scores, as well as the average rank of that component variation group across all datasets.

Moreover, as the scale of the NLL metrics ($\mathcal{L}_T$ and $\mathcal{L}_M$) vary significantly from one dataset to another, we standardize their values on each dataset separately prior to applying the aggregation procedure above. For each model, we compute a standardized NLL metric $\mathcal{L}$ ($\mathcal{L}_T$ or $\mathcal{L}_M$) as

$$\frac{\mathcal{L}_d^m - m_d}{\text{IQR}}, \tag{3.63}$$

where $\mathcal{L}_d^m$ is the NLL value of model m on dataset d, while m_d and IQR are the median and inter-quartile ranges of NLL values for all models on dataset d, respectively.

3.4. Results and Discussion

Tables 3.4 and 3.5 present the average results across all marked datasets for different variations of decoders and event encoding/history encoders, respectively. Table 3.6 displays the results for the combination of architectural components that achieved the lowest $\mathcal{L}_T$ and $\mathcal{L}_M$ on average across all marked datasets and decoders. Appendix C.2 provides the same results for unmarked datasets, while Appendix C.3 includes additional raw metrics and standard errors for each dataset. In the tables, the "Mean" and "Median" columns represent the average and median aggregated scores, respectively. The "Rank" column indicates the average rank across all marked datasets, as explained in the previous section. For the subsequent discussion, we will focus on the "Mean" column.

We would like to note that while these tables include all marked datasets, we found that certain datasets (MIMIC2, Stack Overflow, Taxi, Reddit Subs, Reddit Comments, Yelp Toronto, and Yelp Mississauga) may not be suitable for benchmarking neural MTPP models, as most decoders would achieve competitive time and mark predictive performance on them. We urge researchers to exercise caution when using these datasets in future studies, and we will discuss this concern later in this section. For completeness, we provide the results of the aggregation procedure with these datasets excluded from the analysis in Appendix C.4, and we found no significant differences in the results.

Analysis of the event encoding. Comparing the results in Table 3.4, we aim to provide answers to the two following questions: (1) *Are vectorial representations of time more appropriate to estimate the distributions of inter-arrival times?* (2) *Do we need to encode past mark occurrences to better model the distribution of future arrival times and marks?*

(1) We observe that vectorial representations of time (i.e. TEM and LE in opposition to TO and LTO) improve $\mathcal{L}_T$ and PCE scores for the EC and both SA decoders. Given that a model equipped with the EC decoder only uses the event encoding mechanism at the history encoding stage, this finding suggests that GRU and SA encoders rely on expressive transformations of time to capture patterns of event occurrences. However, we observed that the GRU encoder is rather stable with respect to the time encoding employed, while the performance of the SA encoder drastically decreases with TO or LTO encodings. Added to the fact that both SA decoders only perform well when using TEM and LE, we conclude that self-attention mechanisms must be combined with vectorial representations of time to yield good performance in the context of neural MTPP models.

Furthermore, a log transformation of inter-arrival times (i.e. LTO) improves performance on the same metrics for the RMTPP, FNN and MLP/MC decoders. Using the LTO encoding in combination with the RMTPP decoder effectively defines an inverse Weibull distribution for the inter-arrival times (Kleiber & Kotz, 2003). It appears to be a better fit for the data than the Gompertz distribution from the original formulation of RMTPP.

Moreover, for FNN and MLP/MC, we found that the last softplus activation prevents the gradient of $\lambda^*(t;\boldsymbol{\theta})$ to taking large values for very short inter-arrival times. However, in many of the considered datasets, most events occur in clusters during extremely short time spans, requiring $\lambda^*(t;\boldsymbol{\theta})$ to change quickly between two events. The LTO encoding in combination with the softplus activation allows steeper gradients for short inter-arrival times and thus enables rapid changes in $\lambda^*(t;\boldsymbol{\theta})$. As a result, the FNN and MLP/MC decoders can reach lower $\mathcal{L}_T$ with the LTO encoding.

(2) Including a mark representation in the event encoding generally improves performance in terms of $\mathcal{L}_T$ and PCE when moving from TO to CONCAT and from LTO to LCONCAT. On the one hand, this observation suggests that information contained in previous marks does help the model to better estimate the arrival times of future events. However, we observe that in most cases, TEMWL and LEWL encodings show higher $\mathcal{L}_T$ compared to their TE and LE counterparts. Therefore, relevant information contained in previous marks appears less readily exploitable by the models when the mark embedding is concatenated to a vectorial representation of time.

All decoders improve substantially with respect to $\mathcal{L}_M$, ECE, and F1-score metrics when the mark is included in the event encoding. While expected, this finding confirms that expressive representations of past marks are paramount to the mark prediction task.

Analysis of the history encoder. From the results of Table 3.5, we observe that models equipped with a GRU history encoder yield overall better performance with respect to all time and mark-specific metrics compared to ones equipped with a SA encoder. While self-attention mechanisms have gained increasing popularity since their introduction by Vaswani et al. (2017) for sequence modeling tasks, we found that they are on average less suited than their RNN counterparts in the context of MTPP modeling. Specifically, the GRU encoder is more stable with respect to the choice of event encoding mechanism, while the SA encoder requires vectorial event representation to achieve good performance. Furthermore, the CONS history-independent encoder systematically achieves the worst results with respect to all metrics. This observation confirms the common assumption that future event occurrences are directly influenced by past observations in real-world processes.

Analysis of the decoder. In Table 3.6, we report for each decoder separately the combination that yielded the best performance with respect to $\mathcal{L}_T$ (top rows) and $\mathcal{L}_M$ (bottom rows), on average across all marked datasets. While we previously explained that some variations of event encoding and history encoders worked on average better for a given decoder, it does not necessarily mean that the best combination includes that variation. Moreover, we find that a single combination does not perform equally well on both metrics separately. In the following, we will thus focus our discussion on the top-row models of Table 3.6 for time-related metrics ($\mathcal{L}_T$, PCE), and on bottom-row models for the mark-related ones ($\mathcal{L}_M$, ECE, F1-score).

Table 3.4: Average and median scores, as well as average ranks per decoder and variation of event encoding, for marked datasets. Best results are highlighted in bold. Among others, our key insights include that (1) the SA/MC and SA/CM decoders achieve significantly lower $\mathcal{L}_T$ and PCE when combined with vectorial representations of time (i.e. TEM and LE) compared to the TO and LTO encodings, (2) in comparison to the TO encoding, LTO significantly improves performance on the $\mathcal{L}_T$ and PCE for the RMTPP, FNN and MLP/MC decoders, and (3) including a representation of past observed marks when encoding the history (CONCAT, TEMWL, LEWL) is paramount to achieve good performance with respect to the $\mathcal{L}_M$, ECE and F1-score for all decoders. Refer to text for more details, and to Section 3.3.5 for details on the aggregation procedure.

	Marked Datasets														
	$\mathcal{L}_T$			PCE			$\mathcal{L}_M$			ECE			F1-score		
	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank
EC-TO	0.59	0.56	6.5	0.2	0.23	6.25	0.19	0.21	6.5	0.44	0.44	7.12	0.19	0.17	6.88
EC-LTO	0.7	0.57	7.25	0.2	0.23	6.38	0.15	0.17	5.5	0.44	0.45	7.0	0.19	0.17	7.25
EC-CONCAT	0.13	0.14	**2.5**	**0.17**	**0.15**	**2.25**	-0.51	-0.75	2.75	0.34	0.31	**1.88**	**0.3**	0.28	2.12
EC-LCONCAT	0.36	0.29	4.25	0.19	0.2	4.75	-0.45	-0.66	3.5	0.34	0.3	2.75	0.28	0.27	2.88
EC-TEM	0.2	0.17	4.38	0.18	0.15	4.5	0.18	0.2	6.38	0.42	0.42	6.12	0.2	0.18	5.88
EC-TEMWL	1.48	0.3	4.25	0.19	0.17	4.88	**-0.67**	**-0.78**	**2.0**	**0.33**	**0.27**	2.38	0.3	**0.29**	**1.88**
EC-LE	**0.05**	**0.11**	3.0	0.18	0.15	2.88	0.15	0.15	5.38	0.41	0.42	5.38	0.2	0.18	5.75
EC-LEWL	0.17	0.19	3.88	0.18	0.15	4.12	-0.33	-0.59	4.0	0.35	0.35	3.38	0.29	0.28	3.38
LNM-TO	-0.95	-0.57	6.0	**0.02**	0.02	5.38	0.06	0.11	6.88	0.44	0.45	6.88	0.19	0.17	7.62
LNM-LTO	-0.58	-0.59	6.25	0.02	**0.01**	5.5	0.03	0.06	6.38	0.44	0.46	7.25	0.19	0.17	6.5
LNM-CONCAT	**-1.14**	**-0.98**	**2.0**	0.02	0.01	4.0	**-2.25**	-1.47	**1.38**	**0.25**	**0.21**	**1.75**	**0.39**	**0.35**	**1.5**
LNM-LCONCAT	-0.83	-0.83	2.62	0.02	0.01	**2.12**	-1.87	-1.18	2.88	0.29	0.25	2.75	0.34	0.31	2.88
LNM-TEM	-0.87	-0.79	4.75	0.02	0.01	4.5	0.03	0.05	6.62	0.42	0.44	5.75	0.2	0.18	6.0
LNM-TEMWL	-0.83	-0.87	4.75	0.03	0.02	4.12	-1.99	**-1.49**	2.25	0.28	0.25	2.12	0.36	0.32	2.25
LNM-LE	-1.11	-0.78	5.12	0.02	0.02	6.12	0.0	0.02	6.12	0.42	0.44	6.12	0.2	0.18	5.88
LNM-LEWL	-0.95	-0.92	4.5	0.02	0.01	4.25	-1.89	-1.18	3.5	0.3	0.28	3.38	0.32	0.29	3.38
FNN-TO	0.96	0.68	4.5	0.21	0.28	4.5	0.3	0.34	4.5	0.46	0.47	5.25	0.17	0.17	5.25
FNN-LTO	-0.32	-0.31	2.0	**0.03**	**0.02**	2.12	0.05	0.01	3.75	0.4	0.42	2.25	0.21	0.19	2.62
FNN-CONCAT	0.86	0.45	4.38	0.21	0.27	4.5	0.06	0.02	2.88	0.43	0.46	3.5	0.19	0.19	3.0
FNN-LCONCAT	**-0.57**	**-0.66**	**1.0**	0.03	0.02	**1.62**	**-0.99**	**-0.69**	**2.12**	**0.31**	**0.35**	**1.75**	**0.26**	**0.23**	**1.38**
FNN-LE	0.87	0.53	4.25	0.21	0.29	3.75	0.27	0.31	4.12	0.46	0.47	4.25	0.17	0.16	4.62
FNN-LEWL	0.95	0.41	4.88	0.21	0.29	4.5	0.14	0.12	3.62	0.45	0.46	4.0	0.18	0.18	4.12
MLP/MC-TO	0.11	0.22	6.0	0.16	0.16	6.38	0.36	0.33	6.75	0.41	0.4	6.75	0.2	0.18	7.25
MLP/MC-LTO	-0.16	-0.14	4.0	**0.09**	**0.06**	**1.88**	0.44	0.25	6.75	0.39	0.42	6.0	0.21	0.18	6.0
MLP/MC-CONCAT	0.0	0.06	4.88	0.16	0.13	5.62	-0.16	-0.26	**2.62**	0.33	0.34	3.38	0.28	**0.29**	3.12
MLP/MC-LCONCAT	**-0.34**	**-0.28**	**2.25**	0.09	0.07	2.0	-0.19	-0.42	3.25	**0.29**	0.28	**2.38**	0.28	0.28	2.88
MLP/MC-TEM	0.01	-0.01	5.5	0.15	0.13	5.75	0.26	0.23	5.5	0.4	0.41	6.38	0.2	0.18	6.5
MLP/MC-TEMWL	0.29	0.36	7.25	0.18	0.17	7.62	**-0.45**	**-0.45**	**2.62**	0.34	0.33	2.75	**0.3**	0.29	2.5
MLP/MC-LE	-0.2	-0.19	3.12	0.13	0.11	3.12	0.37	0.22	5.62	0.38	0.37	6.0	0.21	0.19	5.5
MLP/MC-LEWL	-0.18	-0.16	3.0	0.13	0.11	3.62	-0.42	-0.35	2.88	0.29	**0.26**	**2.38**	0.3	0.28	**2.25**
RMTPP-TO	0.15	0.23	6.75	0.16	0.16	6.75	0.11	0.12	6.25	0.42	0.43	7.12	0.19	0.18	7.12
RMTPP-LTO	-0.19	-0.19	4.62	0.06	0.05	3.25	0.09	0.07	6.25	0.42	0.42	7.0	0.19	0.18	6.38
RMTPP-CONCAT	-0.05	-0.05	4.0	0.15	0.12	4.12	-1.52	-1.28	2.88	0.26	0.24	2.5	0.35	0.33	2.5
RMTPP-LCONCAT	**-0.42**	**-0.34**	**2.25**	**0.04**	**0.04**	**2.0**	-1.65	**-1.51**	3.0	0.25	**0.21**	2.12	0.37	0.34	2.0
RMTPP-TEM	-0.01	0.01	5.5	0.15	0.12	5.25	0.1	0.15	6.0	0.41	0.41	6.25	0.21	0.19	6.5
RMTPP-TEMWL	0.0	0.05	4.5	0.15	0.14	5.0	**-2.1**	-1.42	**1.5**	**0.24**	0.21	**1.88**	**0.4**	**0.37**	**1.62**
RMTPP-LE	-0.14	0.01	3.88	0.15	0.13	4.75	0.12	0.14	6.25	0.4	0.41	5.62	0.21	0.19	6.0
RMTPP-LEWL	-0.03	-0.0	4.5	0.15	0.13	4.88	-1.19	-1.03	3.88	0.3	0.29	3.5	0.32	0.32	3.88
SA/CM-TO	10.78	0.12	4.75	0.11	0.06	4.12	1.14	0.31	4.75	0.45	0.46	5.12	0.14	0.1	4.75
SA/CM-LTO	20.39	-0.01	3.0	0.1	0.05	2.75	0.66	0.11	4.0	0.43	**0.44**	2.88	0.18	0.1	**2.88**
SA/CM-CONCAT	-0.08	-0.09	3.38	0.07	0.05	3.25	0.6	0.16	4.0	0.44	0.44	3.38	0.17	**0.17**	3.0
SA/CM-LCONCAT	20.53	0.05	4.12	0.12	0.06	3.62	0.61	0.08	**2.5**	0.43	0.45	3.25	0.18	0.11	3.0
SA/CM-LE	**-0.37**	**-0.4**	**2.0**	**0.05**	**0.04**	**2.38**	0.09	0.07	3.0	0.42	0.44	3.62	0.19	0.17	4.12
SA/CM-LEWL	2.14	-0.07	3.75	0.09	0.08	4.88	**-0.0**	**-0.01**	2.75	**0.41**	0.44	**2.75**	**0.2**	0.16	3.25
SA/MC-TO	1.22	1.0	7.38	0.23	0.31	6.75	0.24	0.31	5.12	0.47	0.47	7.12	0.17	0.16	7.5
SA/MC-LTO	1.07	0.82	5.75	0.22	0.29	5.88	0.28	0.4	6.25	0.46	0.47	6.0	0.17	0.16	6.38
SA/MC-CONCAT	0.64	0.85	6.5	0.19	0.17	6.25	0.1	0.2	5.0	0.44	0.47	6.25	0.2	0.17	5.88
SA/MC-LCONCAT	0.53	0.74	5.12	0.19	0.16	6.25	0.16	0.23	5.38	0.43	0.47	5.25	0.2	0.17	5.5
SA/MC-TEM	-0.42	-0.4	3.25	0.1	0.07	2.75	0.15	0.05	5.12	0.39	0.4	4.25	0.2	0.18	4.12
SA/MC-TEMWL	-0.28	-0.26	4.25	0.11	0.08	4.25	**-0.51**	**-0.34**	**1.88**	**0.32**	**0.32**	**1.88**	**0.28**	**0.25**	**1.5**
SA/MC-LE	**-0.76**	**-0.52**	**1.62**	**0.07**	**0.04**	**1.5**	0.11	-0.03	4.88	0.37	0.38	3.38	0.21	0.19	3.38
SA/MC-LEWL	-0.48	-0.45	2.12	0.08	0.07	2.38	-0.29	-0.23	2.38	0.34	0.32	**1.88**	0.25	0.24	1.75

Table 3.5: Average and median scores, as well as average ranks per decoder and variation of history encoder, for marked datasets. Best results are highlighted in bold. The key insight parsed from this Table is that models equipped with a GRU encoder (i.e. GRU-$*$) show overall improved performance with respect to all metrics compared to ones equipped with a self-attention encoder (i.e. SA-$*$). Refer to text for more details, and to Section 3.3.5 for details on the aggregation procedure.

	Marked Datasets														
	$\mathcal{L}_T$			PCE			$\mathcal{L}_M$			ECE			F1-score		
	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank
CONS-EC	1.74	1.34	3.0	0.3	0.35	2.83	0.63	0.56	2.67	0.48	0.49	3.0	0.14	0.06	3.0
SA-EC	0.73	0.63	2.0	0.25	0.27	2.17	0.04	-0.04	2.0	0.43	0.42	2.0	0.2	0.15	1.83
GRU-EC	**0.19**	**0.14**	**1.0**	**0.22**	**0.23**	**1.0**	**-0.17**	**-0.29**	**1.33**	**0.39**	**0.38**	**1.0**	**0.22**	**0.17**	**1.17**
CONS-LNM	-0.35	-0.44	2.83	**0.02**	0.02	2.33	0.58	0.43	3.0	0.48	0.49	3.0	0.14	0.06	3.0
SA-LNM	-0.61	-0.68	1.83	0.02	0.02	2.17	-0.63	-0.39	1.83	0.39	0.39	1.83	0.24	0.17	1.83
GRU-LNM	**-0.8**	**-0.83**	**1.33**	0.03	**0.01**	**1.5**	**-1.11**	**-0.83**	**1.17**	**0.35**	**0.37**	**1.17**	**0.26**	**0.2**	**1.17**
CONS-FNN	1.02	0.72	3.0	0.2	0.24	2.67	0.5	0.25	2.83	0.46	0.46	2.83	0.17	**0.11**	2.5
SA-FNN	0.78	0.53	1.83	0.19	0.23	2.0	0.05	0.06	2.17	0.45	0.45	1.83	**0.18**	0.11	2.0
GRU-FNN	**0.52**	**0.13**	**1.17**	**0.18**	**0.21**	**1.33**	**-0.17**	**-0.04**	**1.0**	**0.42**	**0.44**	**1.33**	0.18	0.11	**1.5**
CONS-MLP/MC	0.5	0.52	3.0	0.19	0.23	2.83	0.4	0.3	2.33	0.39	0.4	2.0	0.22	0.17	2.33
SA-MLP/MC	0.13	0.13	2.0	0.17	0.21	2.0	0.04	-0.02	2.33	0.38	0.39	2.67	0.22	0.17	2.33
GRU-MLP/MC	**-0.15**	**-0.14**	**1.0**	**0.16**	**0.18**	**1.17**	**-0.03**	**-0.26**	**1.33**	**0.35**	**0.37**	**1.33**	**0.23**	**0.18**	**1.33**
CONS-RMTPP	0.87	0.66	3.0	0.18	0.21	2.33	0.97	0.65	2.67	0.47	0.48	3.0	0.13	0.05	3.0
SA-RMTPP	0.13	0.17	1.83	0.17	0.19	2.17	-0.38	-0.36	2.17	0.38	0.39	1.83	0.24	0.18	1.83
GRU-RMTPP	**-0.15**	**-0.17**	**1.17**	**0.15**	**0.17**	**1.5**	**-0.95**	**-0.66**	**1.17**	**0.34**	**0.37**	**1.17**	**0.26**	**0.22**	**1.17**
CONS-SA/CM	0.01	-0.06	3.0	0.1	0.12	2.83	0.51	0.36	2.67	**0.44**	**0.44**	2.17	**0.18**	**0.11**	2.17
SA-SA/CM	-0.17	-0.2	1.83	0.08	**0.08**	1.83	**0.12**	0.09	**1.5**	0.44	0.44	**1.83**	0.18	0.11	**1.17**
GRU-SA/CM	**-0.22**	**-0.22**	**1.17**	**0.07**	0.08	**1.33**	0.12	**0.07**	1.83	0.44	0.44	2.0	0.18	0.11	2.67
CONS-SA/MC	0.44	0.2	3.0	0.18	0.19	2.83	0.42	0.28	2.67	0.41	**0.41**	2.33	**0.2**	**0.13**	2.0
SA-SA/MC	0.07	0.03	1.67	**0.16**	**0.16**	1.83	-0.05	**-0.16**	**1.5**	**0.39**	0.41	**1.67**	0.19	0.12	2.33
GRU-SA/MC	**0.05**	**-0.04**	**1.33**	0.16	0.16	**1.33**	**-0.1**	-0.16	1.83	0.39	0.42	2.0	0.2	0.12	**1.67**

As a first observation, we note that LNM achieves the lowest $\mathcal{L}_T$, outperforming all other baselines. The difference in performance with the LN decoder further suggests that the assumption of log-normality for the distribution of the inter-arrival times is not a sufficient inductive bias by itself and that the additional flexibility granted by the mixture is necessary to achieve a lower $\mathcal{L}_T$. Additionally, most neural baselines achieve on average a lower $\mathcal{L}_T$ than their parametric counterparts, indicating that high-capacity models are more amenable to capturing complex patterns in real-world event data.

Figure 3.3 shows the evolution of $\lambda^*(t;\hat{\boldsymbol{\theta}})$ and $f^*(t;\hat{\boldsymbol{\theta}})$ between two events in a test sequence of LastFM, for the combinations of Table 3.6 that performed the best on $\mathcal{L}_T$. The greater the gap between $\lambda^*(t;\hat{\boldsymbol{\theta}})$ and $f^*(t;\hat{\boldsymbol{\theta}})$, the greater $\Lambda^*(t;\hat{\boldsymbol{\theta}})$ between two events. We observe that most neural MTPP decoders learn to assign high probability mass to very low inter-arrival times. This

Table 3.6: Average and median scores, and average ranks of the best combinations per decoder on $\mathcal{L}_T$ (top rows) and $\mathcal{L}_M$ (bottom row) across all marked datasets. Best results are highlighted in bold.

	Marked Datasets														
	$\mathcal{L}_T$			PCE			$\mathcal{L}_M$			ECE			F1-score		
	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank
GRU-EC-LE	-0.22	-0.02	7.75	0.17	0.14	9.12	-0.1	-0.03	6.5	0.39	0.41	7.0	0.21	0.18	7.38
GRU-LNM-TO	**-1.44**	**-0.93**	**1.5**	**0.02**	**0.01**	**1.88**	-0.06	-0.07	6.5	0.41	0.42	6.75	0.21	0.18	7.12
GRU-LN-LEWL	-0.54	-0.56	5.25	0.07	0.04	5.75	-1.56	-1.38	2.75	0.28	0.28	4.5	0.33	0.35	2.88
GRU-FNN-LCONCAT	-0.6	-0.71	3.38	0.02	0.02	2.25	-1.57	-1.0	3.5	0.29	0.35	4.5	0.27	0.23	4.12
GRU-MLP/MC-LCONCAT	-0.42	-0.38	5.5	0.09	0.07	6.38	-0.47	-0.47	5.62	0.26	0.26	2.88	0.28	0.28	4.38
GRU-RMTPP-LCONCAT	-0.52	-0.41	4.75	0.04	0.03	4.88	-1.88	**-1.99**	**2.38**	0.22	0.15	2.88	0.4	0.35	**1.62**
GRU-SA/CM-LE	-0.46	-0.42	6.12	0.05	0.03	3.75	0.1	0.07	7.75	0.43	0.45	8.12	0.19	0.18	8.75
GRU-SA/MC-LE	-0.85	-0.55	3.88	0.08	0.05	4.88	0.08	-0.04	7.62	0.35	0.37	6.5	0.21	0.2	6.62
Hawkes	1.21	-0.15	7.62	0.11	0.1	6.5	**-3.42**	-1.01	4.12	**0.15**	**0.13**	**2.12**	**0.45**	**0.43**	2.38
Poisson	1.86	1.35	10.75	0.25	0.34	10.25	2.23	1.26	10.75	0.48	0.48	10.5	0.16	0.15	10.75
NH	1.32	1.07	9.5	0.24	0.32	10.38	0.25	0.41	8.5	0.48	0.47	10.25	0.17	0.16	10.75
GRU-EC-TEMWL	0.16	0.16	7.88	0.18	0.16	8.62	-1.42	-1.1	5.0	0.29	0.24	5.75	0.3	0.3	6.12
GRU-LNM-CONCAT	**-1.41**	**-1.0**	**1.12**	**0.02**	**0.01**	**1.62**	-2.62	-1.76	2.5	0.23	0.18	3.88	0.38	0.34	3.0
GRU-LN-CONCAT	-0.51	-0.57	4.25	0.07	0.04	4.88	-2.59	-1.81	3.12	0.25	0.25	4.12	0.33	0.34	4.38
GRU-FNN-LCONCAT	-0.6	-0.71	2.75	0.02	0.02	2.25	-1.57	-1.0	5.25	0.29	0.35	4.88	0.27	0.23	5.12
GRU-MLP/MC-TEMWL	0.11	0.23	7.5	0.17	0.16	7.12	-0.73	-0.64	7.0	0.32	0.33	6.38	0.32	0.3	5.5
GRU-RMTPP-TEMWL + B	-0.19	-0.11	5.62	0.14	0.11	5.62	-2.67	**-1.85**	**2.38**	0.2	0.16	2.88	0.42	0.42	**2.62**
GRU-SA/CM-LEWL	-0.23	-0.23	5.5	0.08	0.07	5.12	-0.18	-0.11	8.12	0.41	0.43	8.5	0.21	0.2	8.88
GRU-SA/MC-TEMWL	-0.29	-0.3	4.88	0.11	0.09	5.25	-0.55	-0.55	7.38	0.31	0.31	6.25	0.28	0.26	6.5

allows them to reach lower $\mathcal{L}_T$ than their parametric counterparts on datasets where events are highly clustered, such as LastFM.

With respect to the mark prediction task, we find that the Hawkes decoder achieves on average the best results, outperforming all baselines in terms of $\mathcal{L}_M$, ECE, and F1-score. While the Hawkes decoder did not perform favorably on time-related metrics, its superiority compared to more complex models on the mark prediction task is rather intriguing. Specifically, the results of Table C.5 show that the Hawkes decoder significantly outperforms other neural baselines on mark related metrics for LastFM, Wikipedia, and Github. Likewise, the RMTPP and LN(M) decoders also show competitive performance on these metrics, despite their simplifying assumption of marks being independent of the time given the history of the process. The overall superiority of parametric and semi-parametric architectures on the mark prediction task suggests that non-parametric decoders may actually suffer from their high flexibility, making them hard to optimize in practice. Nonetheless, although LN, LNM, RMTPP, and Hawkes do better than their non-parametric counterparts, we find that all decoders perform rather poorly on the mark prediction task.

Figure 3.4 shows the CD diagrams between the average ranks of all decoders at the $\alpha = 0.1$ significance level, on each metric separately. The lower the rank (further to the left on the top horizontal axis), the better the performance of

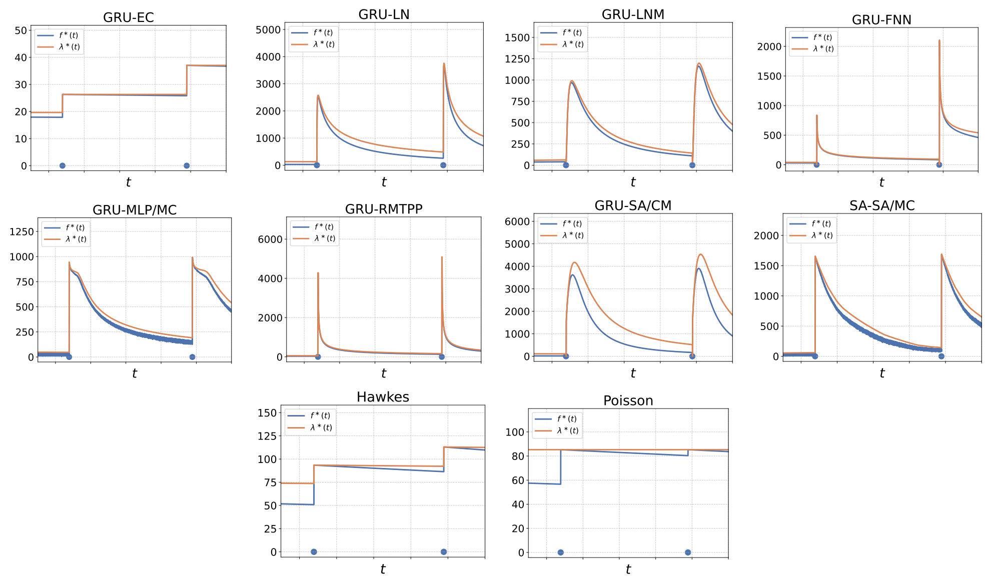


Figure 3.3: Evolution of $\lambda^*(t; \hat{\boldsymbol{\theta}})$ and $f^*(t; \hat{\boldsymbol{\theta}})$ between two events on the LastFM dataset. Arrival times correspond to 2.0975 and 2.0986, respectively.

a decoder with respect to that metric, while a bold black line groups decoders that are not significantly different from one another. As can be seen, there are clear differences between the models' average ranks (e.g. LNM on $\mathcal{L}_T$ for marked datasets). The fact that Holm's posthoc tests do not allow us to conclude statistically significant differences between them can be explained by a large number of pairwise comparisons compared to the few available samples (datasets), which results in high adjusted p-values.

On the calibration of MTPP models. Figure 3.5 shows the reliability diagrams for the time predictive predictive distributions associated with the combinations presented in Table 3.6, averaged over all marked datasets. We observe a good probabilistic calibration for the LNM, FNN, MLP/MC, RMTPP, and SA/CM decoders, as demonstrated by their curves closely matching the diagonal. On the other hand, the other decoders present higher degrees of miscalibration. More precisely, their empirical curve lays systematically above the diagonal line, which suggests that the quantiles of the time predictive distributions are biased upwards, i.e. a larger proportion than p of observations fell below the corresponding predictive quantile at level p.

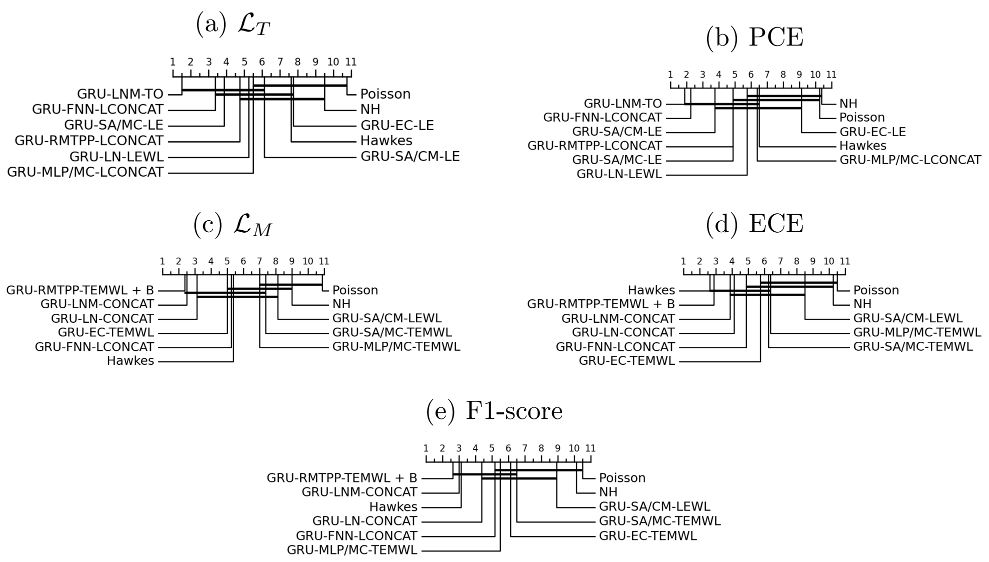


Figure 3.4: CD diagrams per metric at the $\alpha = 0.05$ significance level for all decoders on marked datasets. The decoders' average ranks are displayed on top, and a bold line joins the decoders' variations that are not statistically different.

Figure 3.6 shows the reliability diagrams for the mark predictive distributions associated with the best-performing combinations (in terms of $\mathcal{L}_M$) in Table 3.6, aggregated over all marked datasets. Overall, we observe that the calibration of these decoders mirrors their performance on $\mathcal{L}_M$ and F1-scores, i.e. better values with respect to these metrics correspond to better calibration. Indeed, the Hawkes decoder is the best-calibrated model among our baselines, followed by RMTPP and LNM. Nonetheless, we note that all decoders are usually over-confident in their predictions, as shown by the bin accuracies falling systematically below the diagonal line.

Analysis of the history size. When training a neural MTPP model, it is typically assumed that the complete history $\mathcal{H}_t$, spanning from the first to the last observed event, contains valuable information about the future dynamics of the process. However, in certain real-world scenarios, it may be more appropriate to explicitly assume a q-order Markovian property, where an event $\boldsymbol{e}_i$ is only influenced by q of the last $i-1$ events preceding $\boldsymbol{e}_i$, i.e. $f^*(\boldsymbol{e}_i|\mathcal{H}_t) = f^*(\boldsymbol{e}_i|\boldsymbol{e}_{i-q-1}, ..., \boldsymbol{e}_{i-1})$.

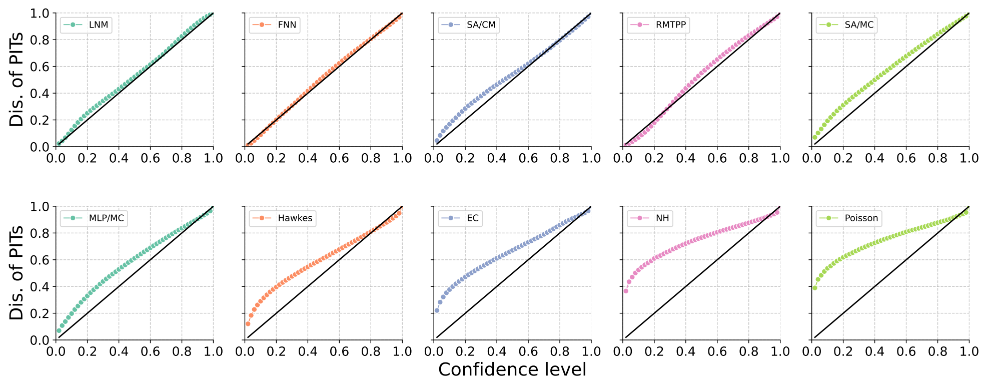


Figure 3.5: Reliability diagrams of the time predictive distributions for the models that performed best on $\mathcal{L}_T$ (top rows of Table 3.6), averaged over all marked datasets. The bold black line corresponds to perfect probabilistic calibration.

When comparing the impact of different encoders, we have found that effectively encoding the process history is crucial for modeling the joint distribution of inter-arrival times and marks. However, it does not provide insights into the number of past events that truly contain useful information. To address this question, we train models using variations of both the GRU and self-attention history encoders. These encoders generate a history embedding $\boldsymbol{h}_i$ for event $\boldsymbol{e}_i$ from $\mathcal{H}^q_{t_i} = \{(t_j, k_j) \in \mathcal{H} | t_j < t_i,\ i - q \leq j < i\}$, which includes at most q events preceding $\boldsymbol{e}_i$.

In Figure 3.7, we show the evolution of $\mathcal{L}_T$ and $\mathcal{L}_M$ (when relevant), by varying the maximal history size q to which the history encoder has access during training. As a brief aside, note that while $\mathcal{L}_M$ can only take positive values due to $p^*(k|\tau; \hat{\boldsymbol{\theta}}) \in [0, 1]$, $\mathcal{L}_T$ is free to take negative or positive values. This is because the PDF $f^*(\tau; \hat{\boldsymbol{\theta}})$ is not constrained to be smaller than 1. We report our observations for the GRU encoder operating on this fixed-size window (GRU-Fixed), but similar results were obtained for a fixed self-attentive encoder (SA-Fixed). Compared to a completely masked history, which is equivalent to training the model with the CONS encoder (i.e. $q = 0$), we observe a substantial improvement in $\mathcal{L}_T$ for most models when only the last event is made available to the encoder. However, on most datasets, additional context does not yield significant improvement, with an $\mathcal{L}_T$ that often quickly stabilizes as q increases.

On the one hand, this finding suggests that real-world processes possess a Markovian property and that going far in history does not bring valuable ad-

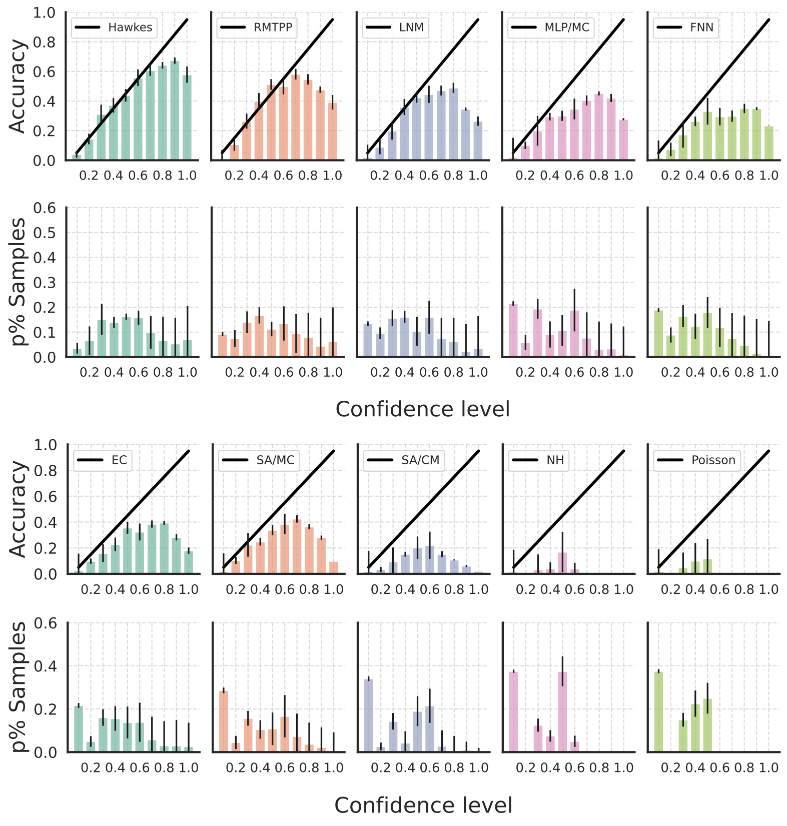
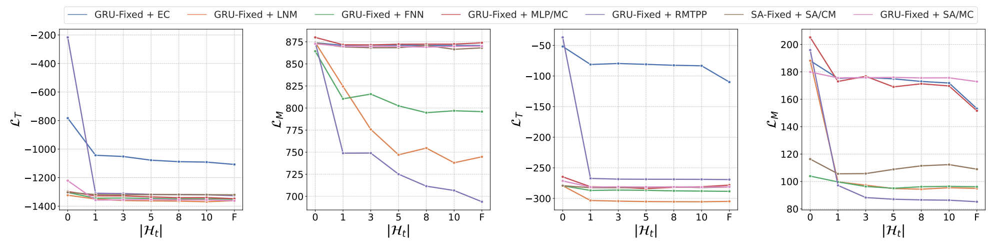


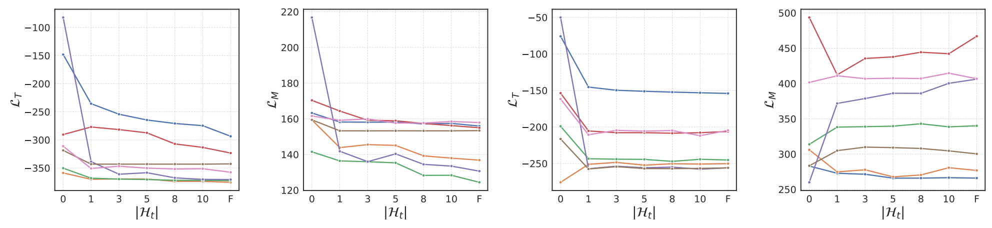
Figure 3.6: Reliability diagrams of the mark predictive distributions for various decoders, averaged over all marked datasets. Top rows depict a decoder's average accuracy per bin, while bottom rows show the average proportion of samples falling per bin. Bins aligning with the bold black line corresponds to perfect calibration. Error bars refer to the standard error.

ditional insight regarding the arrival time of the next event. On the other hand, it could also indicate that RNNs (and self-attention mechanisms) do not fully capture long-term dependencies among event occurrences in the context of MTPPs, warranting future research on better alternatives to encode the history. In essence, masking part of the history translates to a reduction in model complexity, making them effectively less prone to overfitting on the training sequences. Additionally, the EC decoder appears to be more impacted on $\mathcal{L}_T$ as q increases than other baselines. Considering that this decoder can only leverage the history embedding to define time-independent marked intensities, we hypothesize that optimization may force the encoder to extract as much information as possible from the patterns of previous marks occurrences.

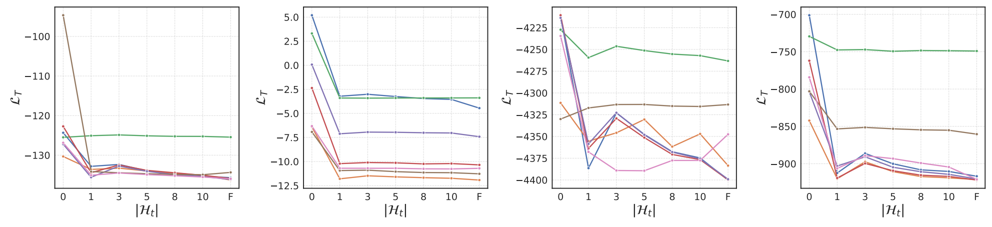
(a) From left to right: LastFM and MOOC.



(b) From left to right: Github and Wikipedia.



(c) From left to right: Taxi, Twitter, Reddit Submissions, and Reddit Comments.



(d) From left to right: PUBG, Yelp Toronto, Yelp Mississauga, and Yelp airport.

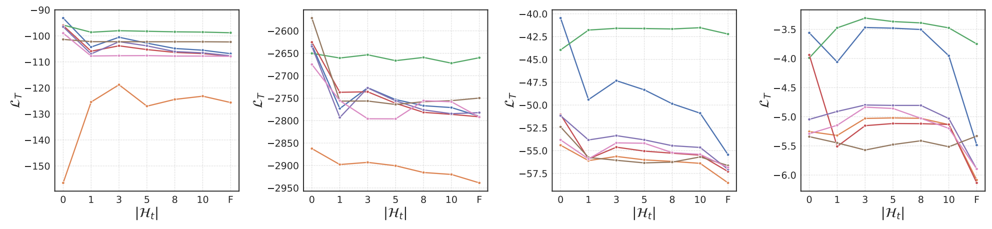
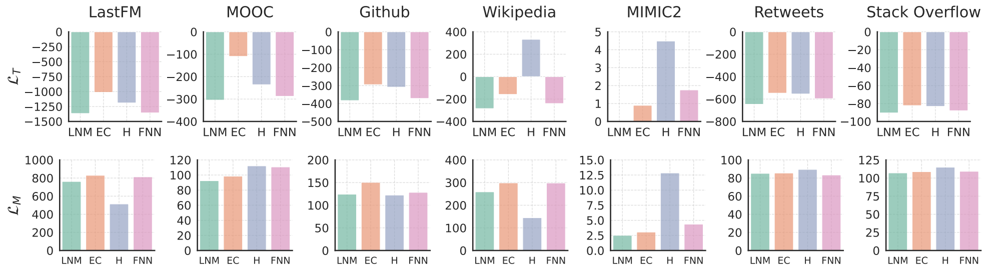


Figure 3.7: Evolution of models' performance with respect to $\mathcal{L}_T$, and $\mathcal{L}_M$ (when available), as a function of the maximal number of events used to construct $\boldsymbol{h}_i$ when using a GRU operating on a fixed-size window. 'F' refers to the unconstrained GRU, i.e. the encoder having access to the full history.

(a) Marked datasets.



(b) Unmarked datasets.

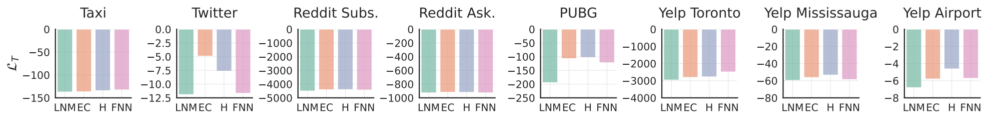


Figure 3.8: Performance of LNM, EC, Hawkes (H), and SA/CM on $\mathcal{L}_T$ and $\mathcal{L}_M$ for each dataset.

A similar statement can be made for $\mathcal{L}_M$. However, we find continuous improvement as q increases for FNN, RMTPP, and LNM on Github and LastFM. As LNM and RMTPP parametrize the PMF of marks solely from $\boldsymbol{h}_i$, they also require expressive representations of the history to perform well with respect to $\mathcal{L}_M$.

On the adequacy of MTPP datasets. A concern that is rarely addressed in the neural MTPP literature relates to the validity of the current benchmark datasets for neural MTPP models. Enguehard et al. (2020) raised some concerns regarding MIMIC2 and Stack Overflow as the simple time-independent EC decoder yielded competitive performance with more complex baselines on such datasets. Figure 3.8 reports both $\mathcal{L}_T$ and $\mathcal{L}_M$ (when available) for LNM, EC, Hawkes and FNN decoders[7] on all real-world datasets. We indeed observe that the EC decoder is competitive with LNM on $\mathcal{L}_T$ and $\mathcal{L}_M$ for MIMIC2 and Stack Overflow, supporting their recommendation that future research should show a certain degree of caution when benchmarking new methods on these datasets. We further express additional concerns about Taxi, Reddit Subs and Reddit Ask Comments, Yelp Toronto, and Yelp Mississauga, on which all decoders achieve comparable performance. All remaining datasets appear to be appropriate benchmarks for evaluating neural MTPP models, as higher variability is observed in the results.

[7]Here also, we employed the combinations on the top row of Table 3.6 to compute $\mathcal{L}_T$ on marked datasets, and the combinations of Table C.3 to compute $\mathcal{L}_T$ on unmarked datasets. $\mathcal{L}_M$ was computed using the combinations on the bottom row of Table 3.6.

Computational times. Panels (a), (b), (c), and (d) of Figure 3.9 show the evolutions of the computational time in seconds (s) and maximum GPU allocated memory in Megabytes (MB) for a single forward and backward pass on a sequence for various decoders as a function of sequence length n_l. All models are trained using a NVIDIA RTX A4000 GPU, and the results are averaged over 50 runs. Additionally, when applicable, the number of parameters is controlled such that each model is equipped with a comparable capacity. On panel (a), we observe that the time complexity of most decoders equipped with a GRU history encoder is linear in n_l, while the time complexity of both SA/CM and SA/MC decoders evolves quadratically. This aligns with the theoretical complexity of GRU and self-attention mechanisms, that respectively require $\mathcal{O}(n_l)$ and $\mathcal{O}(n_l^2)$ operations during training (Goodfellow et al., 2016; Vaswani et al., 2017). Panel (b) further illustrates that the use of a self-attentive history encoder leads to quadratic time complexity as a function of n_l. Moreover, while the time complexity of the Hawkes and NH models in panels (c) and (d) appear both linear, we find that these models are in practice very expensive to train for large n_l, as shown by the large scale differences in the y-axis compared to panels (a) and (b).

Regarding the evolution of the space complexity for panels (a) to (d), we note that the allocated GPU memory increases quadratically as a function of n_l for most models, at the exception of the Hawkes decoder which space complexity remains linear. We now turn our analysis to panel (e), which shows the evolution of the time and space complexity to perform a single forward and backward pass on a fixed-length sequence of 1000 events for increasing number of marks K. We note that while the number of marks does not have a significant impact on computational time, it leads to non negligible increases in allocated GPU memory. Finally, panel (f) presents the evolution of these metrics as a function of Monte Carlo samples n_v used in the evaluation of the compensator in (3.21) for the SA/MC and MLP/MC decoders. As observed, both computational time and GPU memory usage can grow substantially with increasing n_v, particularly for the SA/MC decoder. Hence, using MC decoders on datasets that contain long sequences and a high number of marks can quickly lead to time and memory overheads.

On Table 3.7, we report the computing time (in seconds) for each decoder averaged over 10 epochs for a single forward and backward pass on the training sequences of the MOOC dataset (standard deviation is given in parenthesis). All decoders were equipped with the GRU history encoder and the TEMWL event encoding, when relevant (i.e. not for Hawkes, Poisson, and NH), with similar hyper-parameters configurations. Additionally, the batch size was 32, and the models were trained on an NVIDIA RTX A4000. With respect to the results on both the time and mark prediction tasks, we find LNM to provide the best performance/runtime trade-off among all the decoders considered in the study. On the other hand, the NH decoder is extremely expensive to train, while generally performing worse than other neural baselines.

3.5. Conclusion

In this chapter, we conducted a large-scale experimental study of state-of-the-art neural MTPP models using multiple real-world and synthetic event sequence datasets in a carefully designed setup. Specifically, we studied the influence of major architectural components on predictive accuracy and highlighted that some specific combinations of architectural components can lead to significant improvements for both time and mark prediction tasks.

Among others, our results demonstrated that RNN encoders are in general better suited to capture the history of a process compared to their self-attentive counterparts. In this context, we found that encoders relying on a self-attention mechanism would typically require vectorial representations of arrival times and marks to achieve good performance, whereas RNN encoders are generally stable to the choice of event encoding mechanism. Additionally, we found that neural MTPP baselines would in general outperform their classical counterparts on the time prediction tasks, which aligns with expectations given their enhanced flexibility. Nonetheless, we noticed that neural MTPP models achieve in general unsatisfactory predictive accuracy on the mark prediction task, sometimes being outperformed by simpler classical models. This underscores the difficulty in optimizing these models in practice despite their inherent flexibility.

Moreover, we assessed the rarely discussed topic of probabilistic calibration for neural MTPP models. Our experiments highlighted that mark predictive distributions are often poorly calibrated, despite good calibration for the time predictive distributions. Our experiments also demonstrated that solely encoding a few of the last observed events yielded comparable performance to encoding the complete history for most datasets and decoders. Finally, we confirmed the concerns of previous research regarding the commonly used datasets for benchmarking neural MTPP models and raised concerns about others. We believe our findings will bring valuable insights to the neural MTPP research community, and we hope that they will inspire future work.

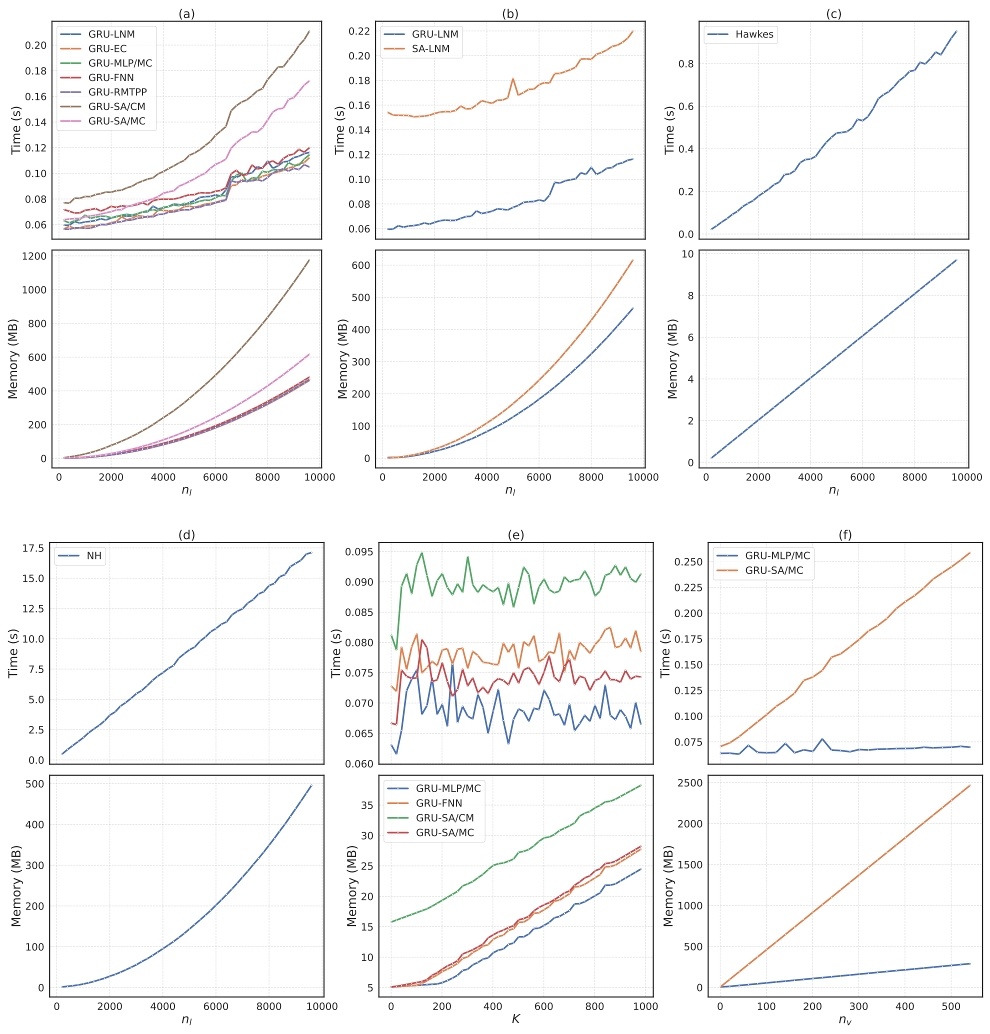


Figure 3.9: Evolution of the time in seconds (s) and maximum GPU memory allocated in megabytes (MB) for a single forward and backward pass on a sequence for different MTPP models by varying the number of events n_l (panels (a), (b), (c), and (d)), the number of marks K (panel (e)), and the number of Monte Carlo samples n_v in the evaluation of (3.21) (panel (f)).

Table 3.7: Average execution time in seconds per decoder for a single forward and backward pass on the MOOC dataset.

EC	LNM	MLP/MC	FNN	RMTPP	SA/CM	SA/MC	Hawkes	Poisson	NH
1.21 (0.06)	1.51 (0.06)	1.66 (0.07)	2.39 (0.07)	1.32 (0.07)	4.01 (0.08)	2.47 (0.06)	17.11 (0.27)	0.66 (0.01)	234.13 (1.36)

CHAPTER 4

Preventing Conflicting Gradients in Neural MTPP Models

The large-scale experimental study conducted in Chapter 3 offered a deeper understanding of the behavior of modern neural MTPP models, highlighting some of their strengths and weaknesses across a wide range of real-world scenarios. In particular, one of the key insights derived from our extensive set of experiments revealed some limitations of these neural parametrizations. Specifically, while neural MTPP models generally perform well on the time prediction task, their effectiveness on the mark prediction task appears suboptimal. Accordingly, this limitation of neural MTPP models has also been reported in related experimental studies (Lin et al., 2021).

Addressing this challenge requires first and foremost to identify the underlying causes in neural MTPP modeling that impede performance in capturing the complex dynamics of mark occurrences. For some MTPP instances, this can be partly imputed to the set of assumptions on which a given model explicitly relies. For instance, consider LNM (Shchur et al., 2020a) and RMTPP (Du et al., 2016) introduced in Section 3.1.3. Both of these models explicitly enforce independence of arrival times and marks given the history of the process. However, in a variety of real-world scenarios, we may reasonably assume that the arrival times and marks of events often exhibit complex inter-dependencies, which cannot be captured by LNM and RMTPP. This intuition has been previously confirmed in the context of EHR (Enguehard et al., 2020)—times and types of medical conditions are correlated, requiring adequate modeling of their dependencies. However, most neural MTPP models do not assume conditional independence of arrival-times and marks, indicating that the cause of poor mark predictive accuracy also stems from a different source.

In this chapter, we argue that learning a neural MTPP model can be interpreted as a two-task learning problem, where both tasks share a common set of parameters and are optimized jointly. Specifically, one task focuses on learning the distribution of the next event's arrival time conditional on historical events. The other task involves learning the distribution of the categorical mark conditional on both the event's arrival time and the historical events. As mentioned previously, we identify these tasks as the time and mark prediction tasks, respectively. While parameter sharing between tasks can sometimes enhance training efficiency (Standley et al., 2020), it may also result in performance degradation when compared to training each task separately. A major challenge in the simultaneous optimization of multi-task objectives is the issue of conflicting gradients (Liu et al., 2021b). This term describes situations where task-specific gradients point in opposite directions. When such conflicts arise, gradient updates tend to favor tasks with larger gradient magnitudes, thus hindering the learning process of other concurrent tasks and adversely affecting their performance. Although the phenomenon of conflicting gradients has been studied in various fields (Chen et al., 2018, 2020; Yu et al., 2020; Shi et al., 2023), its impact on the training of neural MTPP models remains unexplored.

We propose to address this challenge with the following contributions:

- We demonstrate that conflicting gradients frequently occur during the training of neural MTPP models. Furthermore, we show that such conflicts can significantly degrade a model's predictive performance on the time and mark prediction tasks.
- To prevent the issue of conflicting gradients, we introduce novel parametrizations for existing neural MTPP models, allowing for separate modeling and training of the time and mark prediction tasks. Inspired by the success of (Shi et al., 2023), our framework allows to prevent gradient conflicts from the root while maintaining the flexibility of the original parametrizations.
- We want to emphasize that our approach to disjoint parametrizations does not assume the independence of arrival times and marks. Unlike prior studies that assumed conditional independence (Shchur et al., 2020a; Du et al., 2016), we propose a simple yet effective parametrization for the mark conditional distribution that relaxes this assumption.
- Through a series of experiments with real-world event sequence datasets, we show the advantages of our framework over the original model formulations. Specifically, our framework effectively prevents the emergence of conflicting gradients during training, thereby enhancing the predictive accuracy of the models.

4.1. Conflicting Gradients in Two-Task Learning for Neural MTPP Models

Recall from the previous chapter that a neural MTPP model consists of three main components: (1) An *event encoder* that learns a representation $\boldsymbol{l}_i \in \mathbb{R}^{d_l}$ for each event $\boldsymbol{e}_i = (t_i, k_i)$ in a sequence $\mathcal{S}$, (2) a *history encoder* that learns a compact history embedding $\boldsymbol{h}_i \in \mathbb{R}^{d_h}$ of the history $\mathcal{H}_{t_i}$ of event $\boldsymbol{e}_i$, and (3) a *decoder* defining a function that uniquely characterizes the MTPP (e.g. $\lambda_k^*(t; \boldsymbol{\theta})$) from $\boldsymbol{h}_i$.

Let $f^*(\tau, k; \boldsymbol{\theta})$, $\lambda^*_k(t; \boldsymbol{\theta})$, or $\Lambda^*_k(t; \boldsymbol{\theta})$ be a valid model of $f^*(\tau, k)$, where the set of trainable parameters $\boldsymbol{\theta}$ lies within the parameter space Θ. To train this model, we use a dataset $\mathcal{S}_{\text{train}} = \{\mathcal{S}_1, ..., \mathcal{S}_L\}$, where each sequence $\mathcal{S}_l$ comprises n_l events with arrival times observed within the interval $[0, T]$ and $l = 1, ..., L$. As previously, the training objective is the average sequence NLL, optimized using mini-batch stochastic gradient descent (Ruder, 2017):

$$\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = -\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log f^*(\tau_{l,i}, k_{l,i}; \boldsymbol{\theta}) - \log(1 - F^*(T; \boldsymbol{\theta}))\right], \tag{4.1}$$

Consider the factorization of $f^*(\tau_{l,i}, k_{l,i}; \boldsymbol{\theta})$ into $f^*(\tau_{l,i}; \boldsymbol{\theta})$ and $p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta})$, where $f^*(\tau_{l,i}, k_{l,i}; \boldsymbol{\theta}) = f^*(\tau_{l,i}; \boldsymbol{\theta}) \cdot p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta})$. Substituting this decomposition into the NLL in (4.1) and rearranging terms, we obtain:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = &\underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log f^*(\tau_{l,i}; \boldsymbol{\theta}) - \log(1 - F^*(T; \boldsymbol{\theta}))\right]}_{\mathcal{L}_T(\boldsymbol{\theta}, \mathcal{S}_{\text{train}})} \\ &\underbrace{-\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_l}\log p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta})}_{\mathcal{L}_M(\boldsymbol{\theta}, \mathcal{S}_{\text{train}})}.\end{aligned} \tag{4.2}$$

This shows that the total objective function $\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})$ consists of two sub-objectives: $\mathcal{L}_T(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})$ and $\mathcal{L}_M(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})$, revealing that learning an MTPP model can be interpreted as a *two-task* learning problem. The first objective, $\mathcal{L}_T(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})$, relates to modeling the time predictive distribution $f^*(\tau; \boldsymbol{\theta})$, which we refer to as the *time prediction task* $\mathcal{T}_T$. The second objective, $\mathcal{L}_M(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})$, concerns modeling the conditional mark predictive distribution $p^*(k|\tau; \boldsymbol{\theta})$, which we call the *mark prediction task* $\mathcal{T}_M$.

Moreover, recall from Section 2.1.3 that a (strictly) consistent scoring rule for learning (neural) MTPP models can take the general form (Brehmer et al., 2021):

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = &-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l} S^f_{l,i}(f^*(\tau_{l,i}; \boldsymbol{\theta})) + \sum_{i=1}^{n} S^p_{l,i}(p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta}))\right. \\ &\left. - S^F_l\left(1 - F^*(T; \boldsymbol{\theta})\right)\right],\end{aligned} \tag{4.3}$$

where $S^f_{l,i}$, S^F_l, and $S^p_{l,i}$ are (strictly) consistent scoring rules for $f^*(\tau;\boldsymbol{\theta})$, $1-F^*(T;\boldsymbol{\theta})$, and $p^*(k|\tau;\boldsymbol{\theta})$, respectively. Rearranging the terms of (4.3), we find that this expression can also be interpreted as a two-tasks learning objective:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta};\mathcal{S}_{\text{train}}) = & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l} S^f_{l,i}(f^*(\tau_{l,i};\boldsymbol{\theta})) - S^F_l\left(1-F^*(T;\boldsymbol{\theta})\right)\right]}_{\mathcal{L}_T(\boldsymbol{\theta};\mathcal{S}_{\text{train}})} \\ & + \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_l} S^p_{l,i}(p^*(k_{l,i}|\tau_{l,i};\boldsymbol{\theta}))}_{\mathcal{L}_M(\boldsymbol{\theta};\mathcal{S}_{\text{train}})}. \end{aligned} \tag{4.4}$$

Although the following discussion could be extended to a broader range of scoring rules than the LogScore for $S^f_{l,i}$, S^F_l and $S^p_{l,i}$, we narrow our analysis to the objective in (4.2), as the NLL has been widely established as the standard objective for training neural MTPP models (Shchur et al., 2021b).

Let $\boldsymbol{\theta}^s$ be the value of $\boldsymbol{\theta}$ at training iteration s. Assuming that $\mathcal{L}_T(\boldsymbol{\theta}^s)$ and $\mathcal{L}_M(\boldsymbol{\theta}^s)$ are differentiable, let $\boldsymbol{g}_T = \nabla_{\boldsymbol{\theta}}\mathcal{L}_T(\boldsymbol{\theta}^s)$ and $\boldsymbol{g}_M = \nabla_{\boldsymbol{\theta}}\mathcal{L}_M(\boldsymbol{\theta}^s)$ denote the gradients of tasks $\mathcal{T}_T$ and $\mathcal{T}_M$, respectively, with respect to the shared parameters $\boldsymbol{\theta}$[1]. As discussed in (Shi et al., 2023), a small update step during optimization for $\boldsymbol{\theta}$ in the direction of negative $\boldsymbol{g}_T$ is

$$\boldsymbol{\theta}^{s+1} = \boldsymbol{\theta}^s - \eta\boldsymbol{g}_T, \tag{4.5}$$

where $\eta > 0$ is the learning rate. Consider the first order Taylor approximation of $\mathcal{L}_M$ around $\boldsymbol{\theta}^s$, i.e.

$$\mathcal{L}_M\left(\boldsymbol{\theta}^{s+1}\right) = \mathcal{L}_M(\boldsymbol{\theta}^s) + \left(\boldsymbol{\theta}^{s+1}-\boldsymbol{\theta}^s\right)^\intercal \boldsymbol{g}_M + o(\eta). \tag{4.6}$$

where $o(\eta)$ is an error term that vanishes quicker than η. Replacing $\boldsymbol{\theta}^{s+1}$ in this expression with (4.5), the impact that an update in the direction of negative $\boldsymbol{g}_T$ has on the loss of task $\mathcal{T}_M$ is measured as

$$\begin{aligned}\mathcal{L}_M\left(\boldsymbol{\theta}^{s+1}\right) - \mathcal{L}_M(\boldsymbol{\theta}^s) &= (\boldsymbol{\theta}^s - \eta\boldsymbol{g}_T - \boldsymbol{\theta}^s)^\intercal \boldsymbol{g}_M + o(\eta), & (4.7)\\ &= -\eta\boldsymbol{g}_T\cdot\boldsymbol{g}_M + o(\eta), & (4.8)\end{aligned}$$

where $\boldsymbol{g}_T\cdot\boldsymbol{g}_M$ is the dot product between $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$. Consequently, when $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$ are pointing in opposite directions, i.e. $\boldsymbol{g}_T\cdot\boldsymbol{g}_M < 0$, an update

[1]We explicitly omit the dependency of $\mathcal{L}_T$ and $\mathcal{L}_M$ on $\mathcal{S}_{\text{train}}$ to simplify notations.

step in the direction of negative $\boldsymbol{g}_T$ for $\boldsymbol{\theta}$ will increase the loss for task $\mathcal{T}_M$, provided that η is sufficiently small. Similarly, the loss of task $\mathcal{T}_T$ will increase if an update step is taken in the direction of negative $\boldsymbol{g}_M$. Such *conflicting gradients* can be formally defined as follows.

Definition 1 (Conflicting gradients (Shi et al., 2023))**.** *Let* $\phi_{TM} \in [0, 2\pi]$ *be the angle between the gradients* $\boldsymbol{g}_T$ *and* $\boldsymbol{g}_M$*. They are said to be conflicting with each other if* $\cos\phi_{TM} < 0$.

The smaller the value of $\cos\phi_{TM} \in [-1, 0]$, the more severe the conflict between the gradients. Figure (4.1) illustrates these conflicting gradients. Ideally, we want the gradients to align during optimization (i.e. $\cos\phi_{TM} > 0$) to encourage positive reinforcement between the two tasks, or to be simply orthogonal (i.e. $\cos\phi_{TM} = 0$). Conflicting gradients, especially those with significant differences in magnitude, pose substantial challenges during the optimization of multi-task learning objectives (Yu et al., 2020). Specifically, if $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$ conflict, the update step for $\boldsymbol{\theta}$ will likely be dominated by the gradient of whichever task—$\boldsymbol{g}_T$ or $\boldsymbol{g}_M$—has the greater magnitude, thereby disadvantaging the other task. The degree of similarity between the magnitudes of these two gradients can be quantified using a metric known as *gradient magnitude similarity*, defined as follows:

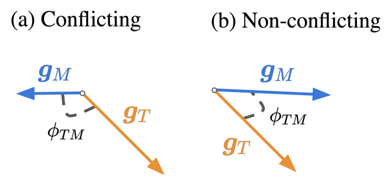


Figure 4.1: Conflicting gradients

Definition 2 (Gradient Magnitude Similarity (GMS) (Yu et al., 2020))**.** *The gradient magnitude similarity between* $\boldsymbol{g}_T$ *and* $\boldsymbol{g}_M$ *is defined as* $GMS = 2 \times \frac{||\boldsymbol{g}_T||_2 \times ||\boldsymbol{g}_M||_2}{||\boldsymbol{g}_T||_2^2 + ||\boldsymbol{g}_M||_2^2}$*, where* $|| \cdot ||_2$ *is the* l_2*-norm.*

A GMS value close to 1 indicates that the magnitudes of $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$ are similar, while a GMS value close to 0 suggests a significant difference between them. Ideally, we aim to minimize the number of conflicting gradients and maintain a GMS close to 1 to ensure balanced learning across the two tasks. However, a low GMS value among conflicting gradients does not specify which task is being prioritized. Therefore, we introduce the *time priority index* to address this issue, which is defined as follows:

Definition 3 (Time Priority Index (TPI))**.** *The time priority index between conflicting gradients* $\boldsymbol{g}_T$ *and* $\boldsymbol{g}_M$ *is defined as* $\text{TPI} = \mathbb{1}\left(||\boldsymbol{g}_T||_2 > ||\boldsymbol{g}_M||_2\right)$ *s.t.* $\cos\phi_{TM} < 0$*, where* $\mathbb{1}(\cdot)$ *is the indicator function.*

If $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$ are conflicting and optimization prioritizes task $\mathcal{T}_T$, then the TPI takes the value 1. Conversely, if task $\mathcal{T}_M$ is prioritized, the TPI takes the value 0.

Do gradients conflict in neural MTPP models? To explore this question, we perform a preliminary experiment with common neural MTPP baselines that either learn $f^*(t,k;\boldsymbol{\theta})$, $\lambda_k^*(t;\boldsymbol{\theta})$, or $\Lambda_k^*(t;\boldsymbol{\theta})$: THP (Zuo et al., 2020), SAHP (Zhang et al., 2020), FNN (Omi et al., 2019), LNM (Shchur et al., 2020a), and RMTPP (Du et al., 2016). We aim to determine whether conflicting gradients occur during their training. For this purpose, each model is trained to minimize the NLL defined in (4.2) using sequences from two real-world datasets: LastFM (Hidasi & Tikk, 2012) and MOOC (Kumar et al., 2019). For optimization, we rely on the Adam optimizer (Kingma & Ba, 2014) used by default in neural MTPP training with learning rate $\alpha = 10^{-3}$. At every gradient update, we calculate the gradients $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$ with respect to the shared parameters $\boldsymbol{\theta}$[2], recording values of cos ϕ_{TM}, GMS, and TPI. Figure 4.2 shows the distribution of cos ϕ_{TM} across all S training iterations, along with the average values of GMS and TPI, and the proportion of conflicting gradients (CG), computed as

$$\text{CG} = \frac{\sum_{s=1}^{S} \mathbb{1}\left(\cos\,\phi_{TM}^s < 0\right)}{\sum_{s=1}^{S} \cos\,\phi_{TM}^s}, \tag{4.9}$$

where ϕ_{TM}^s refers to the angle between $\boldsymbol{g}_T^s$ and $\boldsymbol{g}_M^s$ at training iteration s. Gradients that are conflicting during training correspond to the red bars. We observe that some models, such as THP and SAHP on MOOC, frequently exhibit conflicting gradients during training, as indicated by a high value of CG. Conversely, while other models show a more balanced proportion of conflicting gradients, these are generally characterized by low GMS values, which may potentially impair performance on the task with the lowest magnitude gradient. In this context, the data indicates that optimization generally tends to favor $\mathcal{T}_T$ during optimization on MOOC, LastFM and Github, as suggested by an average TPI greater than 0.5. Conversely, optimization tends to favor $\mathcal{T}_M$ on Stack Overflow and Reddit. In Section 4.4, our experiments show that the combined influence of a high proportion of conflicts and low GMS values during training can significantly deteriorate model performance on the time and mark prediction tasks.

[2]In practice, conflicts are computed layer-wise for each layer of the model (e.g the weights $\mathbf{W}$ of a fully-connected layer).

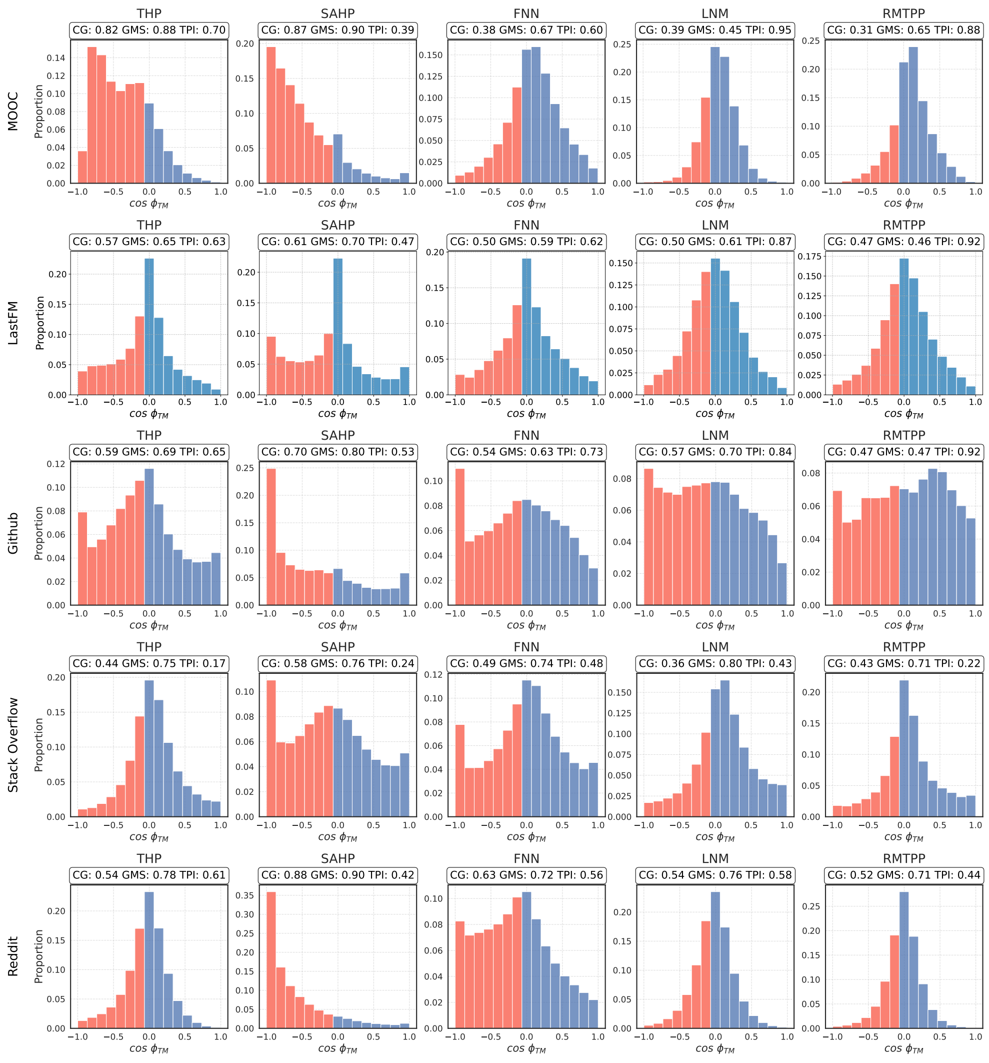


Figure 4.2: Distribution of $\cos\,\phi_{TM}$ during training for THP, SAHP, FNN, RMTPP, and LNM on all datasets. CG refers to the proportion of $\cos\,\phi_{TM} < 0$ observed during training.The distribution is obtained by pooling the values of ϕ_{TM} over 5 training runs, and gradients that are conflicting correspond to the red bars.

4.2. Related Work

Neural MTPP models. To address the limitations of simple parametric MTPP models (Hawkes, 1971; Isham & Westcott, 1979), prior studies focused on designing more flexible approaches by leveraging recent advances in deep learning techniques. Based on the parametrization chosen, these neural MTPP models can be generally classified along three main axis: density-based, intensity-based, and compensator-based approaches. Intensity-based approaches propose to model the trajectories of future arrival-times and marks by parametrizing the marked intensities $\lambda_k^*(t)$. In this line of work, past event occurrences are usually encoded into a history representation using RNNs (Du et al., 2016; Mei & Eisner, 2017; Guo et al., 2018b; Türkmen et al., 2019; Biloš et al., 2019; Zhu et al., 2020) or self-attention (SA) mechanisms (Zuo et al., 2020; Zhang et al., 2020; Zhu et al., 2021; Yang et al., 2022; Li et al., 2023). However, parametrizations of the marked intensity functions often come at the cost of being unable to evaluate the log-likelihood in closed-form, requiring Monte Carlo integration. This consideration motivated the design of compensator-based approaches that parametrize $\Lambda_k^*(t)$ using fully-connected neural networks Omi et al. (2019), or SA mechanisms (Enguehard et al., 2020), from which $\lambda_k^*(t)$ can be retrieved through differentiation. Finally, density-based approaches aim at directly modeling the joint density of (inter-)arrival times and marks $f^*(\tau, k)$. Among these, different family of distributions have been considered to model the distribution of inter-arrival times (Xiao et al., 2017b; Lin et al., 2021). Notably Shchur et al. (2020a) relies on a mixture of log-normal distributions to estimate $f^*(\tau)$, a model that then appeared in subsequent works (Sharma et al., 2021; Gupta et al., 2021). However, the original work of Shchur et al. (2020a) assumes conditional independence of inter-arrival times and marks given the history, which is alleviated in (Waghmare et al., 2022). Nonetheless, a common thread of these parametrizations is that they explicitly enforce parameter sharing between the time and mark prediction tasks. As we demonstrate, this often leads to the emergence of conflicting gradient during training, generally hindering model performance. For an overview of neural MTPP models, we refer the reader to the works of (Shchur et al., 2021b), (Lin et al., 2022) and (Bosser & Ben Taieb, 2023a).

Conflicting gradients in multi-task learning. Diverse approaches have been investigated in the literature to improve interactions between concurrent tasks in multi-task learning problems, thereby boosting performance for each task individually. In this context, a prominent line of work focuses on balancing the different tasks at hand through direct manipulation of their gradients.

These manipulations either aim at alleviating the differences in gradient magnitudes between tasks (Chen et al., 2018; Sener & Koltun, 2018; Liu et al., 2021c), or the emergence of conflicts (Sinha et al., 2018; Maninis et al., 2019; Yu et al., 2020; Chen et al., 2020; Wang et al., 2020; Liu et al., 2021a; Javaloy & Valera, 2022). Alternative approaches to task balancing have been explored based on different criteria, such as task prioritization (Guo et al., 2018a), uncertainty (Kendall et al., 2018), or learning pace (Liu et al., 2019). Another line of work refers to task clustering methods, which aim at identifying the tasks that should be learned jointly (Thrun & O'Sullivan, 1996; Zamir et al., 2018; Standley et al., 2020; Shen et al., 2021; Fifty et al., 2021). At the network level, multi-task learning methods can be partitioned into two main groups. Hard parameter sharing methods denotes methods that share common parameters between multiple tasks (Kokkinos, 2017; Long et al., 2017; Bragman et al., 2019). Conversely, soft sharing methods regroups methods where each task possess its own set of parameters, although a sharing mechanism enables communication across tasks (Misra et al., 2016; Ruder et al., 2019; Gao et al., 2019, 2020). Our methodology relates more to branched architecture search approaches (Guo et al., 2020; Bruggemann et al., 2020; Shi et al., 2023), where the aim is set on dynamically identifying which layers should or should not be shared between tasks based on a chosen criterion, e.g. the proportion of conflicting gradients. Specifically, Shi et al. (2023) recently showed that gradient surgery approaches for multi-task learning objectives, such as GradDrop (Chen et al., 2020), PCGrad (Yu et al., 2020), CAGrad (Liu et al., 2021a), and MGDA (Sener & Koltun, 2018), cannot effectively reduce the occurrence of conflicting gradients during training. Instead, they propose to address task conflicts directly from the root by turning shared layer into task-specific layers if they experience conflicting gradients too frequently. Inspired by their success, we strive to parametrize the time and mark prediction tasks on disjoint set of trainable parameters to avoid conflicts during training. We want to emphasize that our goal is not to propose a general-purpose gradient surgery method to mitigate the negative impact of conflicting gradients. Instead, we want to demonstrate that conflicts can be avoided altogether during the training of neural MTPP models by adapting their original parametrizations.

4.3. A Framework to Prevent Conflicting Gradients in Neural MTPP Models

Given the observations from Section 4.1, our goal is to prevent the occurrence of conflicting gradients during the training of neural MTPP models with the NLL objective given in (4.2). To accomplish this, we first propose in Section 4.3.1 a naive approach that leverages duplicated and disjoint instances of the same model. Then, to avoid redundancy in model specification, we introduce in Section 4.3.2 novel parametrizations for neural MTPP models that retain the flexibility of the original model formulations. We finally show in Section 4.3.3 how our parametrizations enable disjoint modeling and training of the time and mark prediction tasks.

In the previous chapter, we have seen that existing neural MTPP models generally fall into three categories based on the chosen parametrization:

1. *Intensity-based* approaches that model the marked intensity function $\lambda_k^*(t;\boldsymbol{\theta})$ (Zuo et al., 2020; Zhang et al., 2020; Mei & Eisner, 2017).
2. *Density-based* approaches that model the joint density $f^*(t,k;\boldsymbol{\theta})$ (Shchur et al., 2020a).
3. *Compensator-based* approaches that model the marked compensator $\Lambda_k^*(t;\boldsymbol{\theta})$ (Omi et al., 2019; Enguehard et al., 2020).

We denote these different approaches as *joint parametrizations* because they define a function that involves both the arrival time and the mark. To enable disjoint modeling of the time and mark prediction tasks, we consider the factorization of these functions into the products of two components: one involving a function of the arrival-times, and the other involving the conditional PMF of marks:

$$\lambda_k^*(t;\boldsymbol{\theta}) = \lambda^*(t;\boldsymbol{\theta})p^*(k|t;\boldsymbol{\theta}) = \frac{d}{dt}\left[\Lambda^*(t;\boldsymbol{\theta})\right]p^*(k|t;\boldsymbol{\theta}), \tag{4.10}$$

$$f^*(t,k;\boldsymbol{\theta}) = f^*(t;\boldsymbol{\theta})p^*(k|t;\boldsymbol{\theta}), \tag{4.11}$$

$$\Lambda_k^*(t;\boldsymbol{\theta}) = \int_{t_{i-1}}^{t} \lambda^*(s;\boldsymbol{\theta})p^*(k|s;\boldsymbol{\theta})ds, \tag{4.12}$$

Similarly to (4.2), a two-task NLL objective involving the r.h.s of expressions (4.10) and (4.12) can be derived, where task $\mathcal{T}_T$ now consists in learning either $\lambda^*(t;\boldsymbol{\theta})$ or $\Lambda^*(t;\boldsymbol{\theta})$. We provide the expression of these two-task NLL objectives in Appendix D.1.

The factorizations presented in expressions (4.10) and (4.11) show that, to enable disjoint modeling of time and mark prediction functions, we need to specify $p^*(k|t;\boldsymbol{\theta}_M)$ with parameters $\boldsymbol{\theta}_M$, and either $f^*(t;\boldsymbol{\theta}_T)$, $\lambda^*(t;\boldsymbol{\theta}_T)$, or $\Lambda^*(t;\boldsymbol{\theta}_T)$ with parameters $\boldsymbol{\theta}_T$. As a simple approach, we first explore obtaining these functions from duplicated instances of the joint parametrizations $\lambda_k^*(t;\boldsymbol{\theta})$, $\Lambda_k^*(t;\boldsymbol{\theta})$ or $f^*(t,k;\boldsymbol{\theta})$, as presented next.

4.3.1 A Simple Approach to Disjoint Parametrizations

Consider a model that parametrizes $\lambda_k^*(t;\boldsymbol{\theta})$, such as THP (Zuo et al., 2020), which marked intensity functions are given by

$$\lambda_k^*(t;\boldsymbol{\theta}) = \sigma_{S,k}\left(w_k^t \frac{t-t_{i-1}}{t_{i-1}} + (\boldsymbol{w}_k^h)^\intercal \boldsymbol{h} + b_k\right), \tag{4.13}$$

where $\sigma_{S,k}$ is the mark-wise softplus activation in (3.18), while $w_k^t \in \mathbb{R}_+$, $\boldsymbol{w}_k^h \in \mathbb{R}^{d_h}$, and $b_k \in \mathbb{R}$. For a query time $t \geq t_{i-1}$, the history representation $\boldsymbol{h}$ is defined as $\boldsymbol{h} = \text{ENC}(\{\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1}\};\boldsymbol{\theta}_h) \in \mathbb{R}^{d_h}$, where $\text{ENC}(\cdot\ ;\boldsymbol{\theta}_h)$ denotes the history encoder with parameters $\boldsymbol{\theta}_h$. The encoder $\text{ENC}(\cdot\ ;\boldsymbol{\theta}_h)$ is general and encompasses any encoder architecture typically found in the neural MTPP literature, such as recurrent neural networks (RNN) (Du et al., 2016; Shchur et al., 2020a) or self-attention mechanisms (Zuo et al., 2020; Zhang et al., 2020).

To obtain a disjoint parametrization of $\lambda^*(t;\boldsymbol{\theta}_T)$ and $p^*(k|t;\boldsymbol{\theta}_M)$, we can parametrize two identical functions $\lambda_k^*(t;\boldsymbol{\theta}_T)$ and $\lambda_k^*(t;\boldsymbol{\theta}_M)$ from the same model and for all $k \in \mathbb{K}$, where $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_M$ are disjoint set of trainable parameters. Using the relations in (2.9) and (2.10), we can finally derive $\lambda^*(t;\boldsymbol{\theta}_T)$ and $p^*(k|t;\boldsymbol{\theta}_M)$ as

$$\lambda^*(t;\boldsymbol{\theta}_T) = \sum_{k=1}^{K} \lambda_k^*\left(t;\boldsymbol{\theta}_T\right) \quad \text{and} \quad p^*(k|t;\boldsymbol{\theta}_M) = \frac{\lambda_k^*\left(t;\boldsymbol{\theta}_M\right)}{\sum_{k=1}^{K} \lambda_k^*\left(t;\boldsymbol{\theta}_M\right)}, \tag{4.14}$$

effectively defining the desired disjoint parametrization. In the presence of conflicts during training, we can show that a gradient update step for the shared model $\lambda_k^*(t;\boldsymbol{\theta})$ leads to higher loss compared to the duplicated model in (4.14) with disjoint parameters $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_M$. Indeed, suppose that $\boldsymbol{\theta}$, $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_M$ are all initialized with the same $\boldsymbol{\theta}^s$ at training iteration $s \in \mathbb{N}$. Assuming that $\mathcal{L}_T$ and $\mathcal{L}_M$ are differentiable, let

$$\boldsymbol{g}_T = \nabla_{\boldsymbol{\theta}} \mathcal{L}_T\left(\boldsymbol{\theta}^s\right) \quad \text{and} \quad \boldsymbol{g}_M = \nabla_{\boldsymbol{\theta}} \mathcal{L}_M\left(\boldsymbol{\theta}^s\right). \tag{4.15}$$

Denoting ϕ_{TM} as the angle between $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$, we have the following corollary of Theorem 4.1. from (Shi et al., 2023):

Corollary 2. *Assume that $\mathcal{L}_T$ and $\mathcal{L}_M$ are differentiable, and that the learning rate η is sufficiently small. If* $\cos\phi_{TM} < 0$, *then* $\mathcal{L}(\{\boldsymbol{\theta}_T^{s+1}, \boldsymbol{\theta}_M^{s+1}\}) < \mathcal{L}(\boldsymbol{\theta}^{s+1})$.

Proof. At training iteration $s \in \mathbb{N}$, the NLL losses for $\boldsymbol{\theta}^s$ and $\{\boldsymbol{\theta}_T^s, \boldsymbol{\theta}_M^s\}$ respectively write

$$\mathcal{L}(\boldsymbol{\theta}^s) = \mathcal{L}_T(\boldsymbol{\theta}^s) + \mathcal{L}_M(\boldsymbol{\theta}^s), \tag{4.16}$$

and

$$\mathcal{L}(\{\boldsymbol{\theta}_T^s, \boldsymbol{\theta}_M^s\}) = \mathcal{L}_T(\boldsymbol{\theta}_T^s) + \mathcal{L}_M(\boldsymbol{\theta}_M^s). \tag{4.17}$$

Let $\boldsymbol{\theta}^s = \boldsymbol{\theta}_T^s = \boldsymbol{\theta}_M^s$. Assuming that $\mathcal{L}_T$ and $\mathcal{L}_M$ are differentiable, the gradient update steps for $\boldsymbol{\theta}^s$, $\boldsymbol{\theta}_T^s$ and $\boldsymbol{\theta}_M^s$ are

$$\boldsymbol{\theta}^{s+1} = \boldsymbol{\theta}^s - \eta(\boldsymbol{g}_T + \boldsymbol{g}_M), \quad \boldsymbol{\theta}_T^{s+1} = \boldsymbol{\theta}^s - \eta\boldsymbol{g}_T \quad \text{and} \quad \boldsymbol{\theta}_M^{s+1} = \boldsymbol{\theta}^s - \eta\boldsymbol{g}_M, \tag{4.18}$$

where η is the learning rate and

$$\boldsymbol{g}_T = \nabla_{\boldsymbol{\theta}}\mathcal{L}_T(\boldsymbol{\theta}^s) = \nabla_{\boldsymbol{\theta}_T}\mathcal{L}_T(\boldsymbol{\theta}_T^s), \tag{4.19}$$

$$\boldsymbol{g}_M = \nabla_{\boldsymbol{\theta}}\mathcal{L}_M(\boldsymbol{\theta}^s) = \nabla_{\boldsymbol{\theta}_M}\mathcal{L}_M(\boldsymbol{\theta}_M^s), \tag{4.20}$$

where we used the fact that $\boldsymbol{\theta}^s = \boldsymbol{\theta}_T^s = \boldsymbol{\theta}_M^s$. Let us consider the first order Taylor approximations of the total loss $\mathcal{L} = \mathcal{L}_T + \mathcal{L}_M$ near $\boldsymbol{\theta}^{s+1}$ and $\{\boldsymbol{\theta}_T^{s+1}, \boldsymbol{\theta}_M^{s+1}\}$, respectively:

$$\mathcal{L}(\boldsymbol{\theta}^{s+1}) = \mathcal{L}(\boldsymbol{\theta}^s) + (\boldsymbol{\theta}^{s+1} - \boldsymbol{\theta}^s)^\intercal\boldsymbol{g}_T + (\boldsymbol{\theta}^{s+1} - \boldsymbol{\theta}^s)^\intercal\boldsymbol{g}_M + o(\eta), \tag{4.21}$$

and

$$\mathcal{L}(\{\boldsymbol{\theta}_T^{s+1}, \boldsymbol{\theta}_M^{s+1}\}) = \mathcal{L}(\boldsymbol{\theta}^s) + (\boldsymbol{\theta}_T^{s+1} - \boldsymbol{\theta}^s)^\intercal\boldsymbol{g}_T + (\boldsymbol{\theta}_M^{s+1} - \boldsymbol{\theta}^s)^\intercal\boldsymbol{g}_M + o(\eta), \tag{4.22}$$

where $o(\eta)$ is an error term that vanishes quicker than η. Taking the difference between $\mathcal{L}(\{\boldsymbol{\theta}_T^{s+1}, \boldsymbol{\theta}_M^{s+1}\})$ and $\mathcal{L}(\boldsymbol{\theta}^{s+1})$ yields

$$\mathcal{L}(\{\boldsymbol{\theta}_T^{s+1}, \boldsymbol{\theta}_M^{s+1}\}) - \mathcal{L}(\boldsymbol{\theta}^{s+1}) = (\boldsymbol{\theta}_T^{s+1} - \boldsymbol{\theta}^{s+1})^\intercal\boldsymbol{g}_T + (\boldsymbol{\theta}_M^{s+1} - \boldsymbol{\theta}^{s+1})^\intercal\boldsymbol{g}_M + o(\eta) \tag{4.23}$$

$$= \eta\boldsymbol{g}_M^\intercal\boldsymbol{g}_T + \eta\boldsymbol{g}_T^\intercal\boldsymbol{g}_M + o(\eta) \tag{4.24}$$

$$= 2\eta||\boldsymbol{g}_T||||\boldsymbol{g}_M||\cos\phi_{TM} + o(\eta). \tag{4.25}$$

Provided that η is sufficiently small, this difference is negative if $\cos\phi_{TM} < 0$, where ϕ_{TM} is the angle between $\boldsymbol{g}_T$ and $\boldsymbol{g}_M$. □

This result essentially indicates that a model trained with disjoint parameters leads to lower loss after a gradient update if conflicts arise during training, i.e. $\cos \phi_{TM} < 0$. Naturally, expression (4.14) and Corollary 2 remain valid for models that parametrize $\Lambda_k^*(t;\boldsymbol{\theta})$ or $f^*(\tau, k;\boldsymbol{\theta})$, as these functions can be uniquely retrieved from $\lambda_k^*(t;\boldsymbol{\theta})$.

4.3.2 Disjoint Parametrizations of Neural MTPP Models

In this section, we introduce an alternative approach to (4.14) to achieve disjoint parametrizations of $\mathcal{T}_T$ and $\mathcal{T}_M$. Specifically, we introduce novel parametrizations of existing neural MTPP models that directly parametrize $p^*(k|t;\boldsymbol{\theta}_M)$ and either $f^*(t;\boldsymbol{\theta}_T)$, $\lambda^*(t;\boldsymbol{\theta}_T)$, or $\Lambda^*(t;\boldsymbol{\theta}_T)$, thereby avoiding the unnecessary redundancy in model specification required by the method in (4.14).

A general approach to model the distribution of marks. Given a query time $t \geq t_{i-1}$ and its corresponding history representation $\boldsymbol{h}$, we propose to define the conditional PMF of marks $p^*(k|t;\boldsymbol{\theta}_M)$ using the following simple model:

$$p^*(k|t;\boldsymbol{\theta}_M) = \sigma_{\text{so}}\left(\mathbf{W}_2\sigma_R\left(\mathbf{W}_1\left[\boldsymbol{h}||\log(\tau)\right] + \boldsymbol{b}_1\right)\right) + \boldsymbol{b}_2\right), \qquad (4.26)$$

where $\tau = t - t_{i-1}$, σ_{so} is the softmax activation function, $\mathbf{W}_1 \in \mathbb{R}^{d_1\times(d_h+1)}$, $\boldsymbol{b}_1 \in \mathbb{R}^{d_1}$, $\mathbf{W}_2 \in \mathbb{R}^{K\times d_1}$, $\boldsymbol{b}_2 \in \mathbb{R}^{K}$, and $||$ means concatenation. Here, $\boldsymbol{\theta}_M = \{\mathbf{W}_1, \mathbf{W}_2, \boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{\theta}_h\}$. Despite its simplicity, this model is flexible and capable of capturing the evolving dynamics of the mark distribution between two events. Note that (4.26) effectively captures inter-dependencies between arrival times and marks. Moreover, by removing $\log(t - t_{i-1})$ from expression (4.26), we obtain a PMF of marks that is independent of time, given $\mathcal{H}_t$. Finally, while any of the mechanisms discussed in Section 3.1.1 could be used to encode the inter-arrival time τ, we empirically found that taking a logarithmic transformation would lead to the most competitive results on the mark prediction task across all datasets.

Intensity-based parametrizations. We first consider MTPP models specified by their marked intensities, namely THP (Zuo et al., 2020) and SAHP (Zhang et al., 2020). We propose revising the original model formulations to directly parametrize $\lambda^*(t;\boldsymbol{\theta}_T)$, while $p^*(k|t;\boldsymbol{\theta}_M)$ is systematically derived from expression (4.26). Furthermore, although RMTPP (Du et al., 2016) is originally defined in terms of this decomposition, we extend the model to incorporate the dependence of marks on time.

SAHP$^+$. While the original model formulation parametrizes $\lambda_k^*(t;\boldsymbol{\theta})$, we adapt its formulation to define:

$$\lambda^*(t;\boldsymbol{\theta}_T) = \mathbf{1}^\mathsf{T}\left[\sigma_S\left(\boldsymbol{\mu} - (\boldsymbol{\eta} - \boldsymbol{\mu})\exp(-\boldsymbol{\gamma}(t - t_{i-1}))\right)\right], \tag{4.27}$$

where $\mathbf{1} \in \mathbb{R}^C$ is a vector of 1's allowing to define the ground intensity as a sum over C different representations. In (4.27), $\boldsymbol{\mu} = \sigma_G(\mathbf{W}_\mu \boldsymbol{h})$, $\boldsymbol{\eta} = \sigma_S(\mathbf{W}_\eta \boldsymbol{h})$ and $\boldsymbol{\gamma} = \sigma_G(\mathbf{W}_\gamma \boldsymbol{h})$ where σ_S and σ_G are respectively the softplus and GeLU activation functions (Hendrycks & Gimpel, 2023). $\mathbf{W}_\mu, \mathbf{W}_\eta, \mathbf{W}_\gamma \in \mathbb{R}^{C\times d_h}$ are learnable parameters and $\boldsymbol{\theta}_T = \{\mathbf{W}_\mu, \mathbf{W}_\eta, \mathbf{W}_\gamma, \boldsymbol{\theta}_h\}$.

THP$^+$. Following a similar reasoning, the original formulation of THP is adapted to model $\lambda^*(t;\boldsymbol{\theta}_T)$ instead of $\lambda_k^*(t;\boldsymbol{\theta})$:

$$\lambda^*(t;\boldsymbol{\theta}_T) = \mathbf{1}^\mathsf{T}\left[\sigma_S\left(\boldsymbol{w}_t \frac{t - t_{i-1}}{t_{i-1}} + \mathbf{W}\boldsymbol{h} + \boldsymbol{b}\right)\right], \tag{4.28}$$

where $\boldsymbol{w}_t \in \mathbb{R}_+^C$, $\mathbf{W} \in \mathbb{R}^{C\times d_h}$, $\boldsymbol{b} \in \mathbb{R}^{d_1}$, and $\boldsymbol{\theta}_T = \{\mathbf{W}, \boldsymbol{w}_t, \boldsymbol{b}, \boldsymbol{\theta}_h\}$.

RMTPP$+$. We retain the original definition of $\lambda^*(t)$ proposed in RMTPP:

$$\lambda^*(t;\boldsymbol{\theta}_T) = \exp\Big(w_t(t - t_{i-1}) + \boldsymbol{w}_h^\mathsf{T}\boldsymbol{h} + b\Big), \tag{4.29}$$

where $w_t \in \mathbb{R}_+$, $\boldsymbol{w}_h \in \mathbb{R}^{d_h}$, $b \in \mathbb{R}$, and $\boldsymbol{\theta}_T = \{w_t, \boldsymbol{w}_h, b, \boldsymbol{\theta}_h\}$. A major difference between the original formulation of RMTPP and our approach lies in $p^*(k|\tau;\boldsymbol{\theta}_M)$ being defined by (4.26), which alleviates the original model assumption of marks being independent of time given $\mathcal{H}_t$.

Density-based parametrizations. The decomposition of the joint density in expression (4.11) has previously been considered by (Shchur et al., 2020a) with the LogNormMix (LNM) model. However, similar to RMTPP, this model assumes independence of marks and time, given $\mathcal{H}_t$. In *LNM*$^+$, we relax this assumption by using expression (4.26) to parametrize $p^*(k|t;\boldsymbol{\theta}_M)$, while maintaining a mixture of log-normal distributions for $f^*(\tau;\boldsymbol{\theta}_T)$:

$$f^*(\tau;\boldsymbol{\theta}_T) = \sum_{m=1}^{M} p^*(m)\frac{1}{\tau\sigma_m\sqrt{2\pi}}\exp\Big(-\frac{(\log\tau - \mu_m)^2}{2\sigma_m^2}\Big), \tag{4.30}$$

where

$$\mu_m = (\mathbf{W}_\mu \boldsymbol{h} + \boldsymbol{b}_\mu)_m, \quad \sigma_m = \exp(\mathbf{W}_\sigma \boldsymbol{h} + \boldsymbol{b}_\sigma)_m, \tag{4.31}$$

and

$$p^*(m) = \sigma_{\text{so}}\big(\mathbf{W}_p \boldsymbol{h} + \boldsymbol{b}_p\big)_m, \tag{4.32}$$

with $\mathbf{W}_p, \mathbf{W}_\mu, \mathbf{W}_\sigma \in \mathbb{R}^{M \times d_h}$ and $\boldsymbol{b}_p, \boldsymbol{b}_\mu, \boldsymbol{b}_\sigma \in \mathbb{R}^M$, M being the number of mixture components. Here $\boldsymbol{\theta}_T = \{\mathbf{W}_\mu, \mathbf{W}_p, \mathbf{W}_\sigma, \boldsymbol{b}_\mu, \boldsymbol{b}_p, \boldsymbol{b}_\sigma, \boldsymbol{\theta}_h\}$. Again, parametrizing the mark distribution using (4.26) relaxes the original assumption of marks being independent of time given $\mathcal{H}_t$.

Compensator-based parametrizations. Integrating compensator-based neural MTPP models into our framework requires to define $\Lambda^*(t; \boldsymbol{\theta}_T)$, this way retrieving the decomposition in the r.h.s. of (4.10). Specifically, we extend the improved marked FullyNN model (FNN) (Omi et al., 2019; Enguehard et al., 2020) into *FNN*$^+$, that models $\Lambda^*(t; \boldsymbol{\theta}_T)$ and $p^*(k|t; \boldsymbol{\theta}_M)$, instead of $\Lambda_k^*(t; \boldsymbol{\theta})$:

$$\Lambda^*(t; \boldsymbol{\theta}_T) = G^*(t) - G^*(t_{i-1}), \tag{4.33}$$

$$G^*(t) = \mathbf{1}^\intercal \left[\sigma_S\big(\mathbf{W}(\sigma_{GS}\big(\boldsymbol{w}_t(t - t_{i-1}) + \mathbf{W}_h \boldsymbol{h} + \boldsymbol{b}_1\big) + \boldsymbol{b}_2\big)\right], \tag{4.34}$$

where $\mathbf{W} \in \mathbb{R}^{C \times d_1}$, $\boldsymbol{w}_t, \boldsymbol{b}_1 \in \mathbb{R}^{d_1}$, $\mathbf{W}_h \in \mathbb{R}^{d_1 \times d_h}$, $\boldsymbol{b}_2 \in \mathbb{R}^C$, and σ_{GS} is the Gumbel-softplus activation function. Here, $\boldsymbol{\theta}_T = \{\mathbf{W}, \mathbf{W}_h, \boldsymbol{w}_t, \boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{\theta}_h\}$. Similarly to the previous models, we use (4.26) to define $p^*(k|t)$.

Training different history encoders. The different functions defined in this section, $p^*(k|\tau; \boldsymbol{\theta}_M)$, $f^*(\tau; \boldsymbol{\theta}_T)$, $\lambda^*(t; \boldsymbol{\theta}_T)$, and $\Lambda^*(t; \boldsymbol{\theta}_T)$, share a common set of parameters $\boldsymbol{\theta}_h$ through a common history representation $\boldsymbol{h}$. To enable fully disjoint modeling and training of the time and mark predictive functions, we define two distinct history representations:

$$\boldsymbol{h}_T = \text{ENC}_T\left[\{\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1}\}; \boldsymbol{\theta}_{T,h}\right] \in \mathbb{R}^{d_h^t} \tag{4.35}$$

$$\boldsymbol{h}_M = \text{ENC}_M\left[\{\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1}\}; \boldsymbol{\theta}_{M,h}\right] \in \mathbb{R}^{d_h^m}, \tag{4.36}$$

where $\text{ENC}_T(\cdot\ ; \boldsymbol{\theta}_{T,h})$ and $\text{ENC}_M(\cdot\ ; \boldsymbol{\theta}_{M,h})$ are the *time* and *mark* history encoders, respectively, while $\boldsymbol{\theta}_{T,h}$ and $\boldsymbol{\theta}_{M,h}$ represent the sets of *disjoint* learnable parameters of the two encoders. By using $\boldsymbol{h}_T$ for $f^*(\tau; \boldsymbol{\theta}_T)$, $\lambda^*(t; \boldsymbol{\theta}_T)$ and $\Lambda^*(t; \boldsymbol{\theta}_T)$, and $\boldsymbol{h}_M$ for $p^*(k|\tau; \boldsymbol{\theta}_M)$[3], we have defined completely disjoint parametrizations of the decompositions in (4.10) and (4.11). Using separate history encoders further enables the model to capture information from past event occurrences that are relevant to the time and mark prediction tasks separately. In this chapter, without loss of generality, we compute $\boldsymbol{h}_T$ and $\boldsymbol{h}_M$ by training two GRU encoders that sequentially process the set of event representations in $\{\boldsymbol{e}_1, ..., \boldsymbol{e}_{i-1}\}$.

[3] $\boldsymbol{\theta}_h$ is now replaced by $\boldsymbol{\theta}_{T,h}$ in $\boldsymbol{\theta}_T$, and by $\boldsymbol{\theta}_{M,h}$ in $\boldsymbol{\theta}_M$.

4.3.3 Disjoint Training of the Time and Mark Prediction Tasks.

Let us model $f^*(\tau;\boldsymbol{\theta}_T)$ from (4.30) and $p^*(k|\tau;\boldsymbol{\theta}_M)$ from (4.26), using distinct history encoders such that $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_M$ are disjoint set of trainable parameters. By injecting these expressions in (4.2), we find that the NLL is now a sum over two disjoint objectives $\mathcal{L}_T(\boldsymbol{\theta}_T, \mathcal{S}_{\text{train}})$ and $\mathcal{L}_M(\boldsymbol{\theta}_M, \mathcal{S}_{\text{train}})$, i.e.

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}_T, \boldsymbol{\theta}_M; \mathcal{S}_{\text{train}}) = & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log f^*(\tau_{l,i};\boldsymbol{\theta}_T) - \log(1 - F^*(T;\boldsymbol{\theta}_T))\right]}_{\mathcal{L}_T(\boldsymbol{\theta}_T,\mathcal{S}_{\text{train}})} \\ & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_l}\log p^*(k_{l,i}|\tau_{l,i};\boldsymbol{\theta}_M)}_{\mathcal{L}_M(\boldsymbol{\theta}_M,\mathcal{S}_{\text{train}})}, \end{aligned} \tag{4.37}$$

meaning that the associated tasks $\mathcal{T}_T$ and $\mathcal{T}_M$ can be learned separately. This contrasts with previous works, such as (Shchur et al., 2020a) and (Zuo et al., 2020), in which shared parameters between f^* and p^* does not allow for disjoint training of (4.2). In our implementation, we minimize expression (4.37) through a single pipeline by specifying different early-stopping criteria for $\mathcal{L}_T(\boldsymbol{\theta}_T, \mathcal{S}_{\text{train}})$ and $\mathcal{L}_M(\boldsymbol{\theta}_T, \mathcal{S}_{\text{train}})$. Note that a similar decomposition can be obtained from any of the parametrizations presented in Sections 4.3.1 and 4.3.2. Finally, we would like to emphasize that disjoint training of tasks $\mathcal{T}_T$ and $\mathcal{T}_M$ through (4.37) does **not** imply independence of arrival-times and marks given the history. In fact, in our parametrizations, this dependency remains systematically captured by (4.26).

4.4. Experiments

We conduct an experimental study to assess the performance of our framework in training the time and mark prediction tasks from datasets composed of multiple event sequences. Specifically, we explore the various novel neural MTPP parametrizations enabled by our framework, as detailed in Section 4.3.2. These are compared to their original parametrizations[4].

[4]For the remainder of this paper, these models will be referred to as "base models".

4.4.1 Experimental Setup

Datasets. We use five real-world marked event sequence datasets frequently referenced in the neural MTPP literature: **LastFM** (records of people listening to songs), **MOOC** (students' actions on an online learning platform), **Reddit** (users' actions of the social platform Reddit) (Kumar et al., 2019), **Github** (developers' actions on Github) (Trivedi et al., 2019), and **Stack Overflow** (badge received by users on Stack Overflow) (Du et al., 2016). For all datasets, we follow the pre-processing steps described in Section 3.3.2, and the number of sequences in Reddit is further reduced by 50% to reduce computational cost. For a detailed summary and statistical descriptions of these datasets, we direct the reader to Appendix A.

Baselines. We consider as base models **THP** in (4.13) (Zuo et al., 2020), **RMTPP** in (3.25) and (3.26) (Du et al., 2016), FullyNN (**FNN**) in (3.46) (Omi et al., 2019), LogNormMix (**LNM**) in (3.39) and (3.26) (Shchur et al., 2020a). Among our baselines, we also include **SAHP** (Zhang et al., 2020), which marked intensity functions are given by

$$\lambda_k^*(t;\boldsymbol{\theta}) = \sigma_{S,k}\left(\boldsymbol{\mu}_k - (\boldsymbol{\eta}_k - \boldsymbol{\mu}_k)\exp(-\boldsymbol{\gamma}_k(t - t_{i-1}))\right), \tag{4.38}$$

where $\boldsymbol{\mu}_k = \sigma_G(\mathbf{W}_\mu \boldsymbol{h})$, $\boldsymbol{\eta}_k = \sigma_G(\mathbf{W}_\eta \boldsymbol{h})$, and $\boldsymbol{\gamma}_k = \sigma_{S,k}(\mathbf{W}_\gamma \boldsymbol{h}_i)$ with $\mathbf{W}_\mu, \mathbf{W}_\eta, \mathbf{W}_\gamma \in \mathbb{R}^{K\times d_h}$. Finally, we also consider SMURF-THP (**STHP**) (Li et al., 2023), which parametrizes the ground intensity as

$$\lambda^*(t;\boldsymbol{\theta}) = \tanh\left[\boldsymbol{w}\left(\left[\mathbf{W}^t(t - t_{i-1}) + \mathbf{W}^h\right]\boldsymbol{h} + \boldsymbol{b}_1\right) + b_2\right], \tag{4.39}$$

and a mark PMF similarly to (4.26). In (4.39), $\boldsymbol{w}, \boldsymbol{b}_1 \in \mathbb{R}_+^{d_1}$, $\mathbf{W}^t, \mathbf{W}^h \in \mathbb{R}_+^{d_1\times d_h}$, and $b_2 \in \mathbb{R}_+$. To highlight the different components of our framework that lead to performance gains compared to the base models, we introduce the following two settings:

- *Shared History Encoders and Disjoint Decoders.* A common history embedding, denoted as $\boldsymbol{h}$ is used, while the two functional terms from equations (4.10) and (4.11) are modeled separately as detailed in Section 4.3.2. Here, the functions $f^*(\tau;\boldsymbol{\theta}_T)$, $\lambda^*(t;\boldsymbol{\theta}_T)$, $\Lambda^*(t;\boldsymbol{\theta}_T)$ and $p^*(k|\tau;\boldsymbol{\theta}_M)$ share common parameters via $\boldsymbol{h}$. Models trained in this setting are indicated with a "+" sign, e.g. THP+. In this setting, note that the time and mark prediction tasks cannot be trained independently via (4.37), and $\boldsymbol{\theta}_h$ will end up the same for both $\mathcal{T}_T$ and $\mathcal{T}_M$ after optimization.

- *Disjoint History Encoders and Disjoint Decoders.* In contrast to the previous configuration, distinct history embeddings, $\boldsymbol{h}_T$ for time and $\boldsymbol{h}_M$ for marks, are used to define $f^*(\tau; \boldsymbol{\theta}_T)$, $\lambda^*(t; \boldsymbol{\theta}_T)$, $\Lambda^*(t; \boldsymbol{\theta}_T)$, and $p^*(k|\tau; \boldsymbol{\theta}_M)$ separately. This separation allows for the independent training of the time and mark prediction tasks as described in Section 4.3.3. Models trained within this setting are labeled with a "++" symbol, e.g., THP++.

Compared to the base models, these configurations allow us to assess the impacts of (1) isolating the parameters for the decoders in the time and mark prediction tasks, and (2) using distinct history embeddings for each task, enabling fully disjoint training. A graphical illustration of these configurations is shown in Figure 4.3. We will often refer to these setups as base, base+, and base++ throughout the text. The distinction between LNM (RMTPP) and LNM+ (RMTPP+) stems from the modeling of the PMF of marks using our model in (4.26), which relaxes the conditional independence assumption of these models. Finally, we also include in our baselines the classical **Hawkes** model with exponential kernels in A.1. Note that, as the Hawkes model cannot be disentangled into distinct history encoder and decoder components, we only train it in the base setup.

Training details. For all models, we minimize the average NLL in (4.37)[5] on the training sequences using mini-batch gradient descent with the Adam optimizer (Kingma & Ba, 2014) and a learning rate of 10^{-3}. Although the emergence of conflicting gradients might be affected by the choice of optimizer and learning rate, this configuration represents a standard setup for training neural MTPP models (Shchur et al., 2020a; Zuo et al., 2020; Zhang et al., 2020). We plan to investigate the influence of alternative optimizers and learning rate on conflicts in future works.

For the base models and the base+ setup, an early-stopping protocol interrupts training if the model fails to show improvement in the total validation loss (i.e., $\mathcal{L}_T + \mathcal{L}_M$) for 50 consecutive epochs. Conversely, in the base++ setup, two distinct early-stopping protocols are implemented for the $\mathcal{L}_T$ and $\mathcal{L}_M$ terms, respectively. If one of these terms does not show improvement for 50 consecutive epochs, we freeze the parameters of the associated functions (e.g. $\boldsymbol{\theta}_T$ for $f^*(\tau; \boldsymbol{\theta}_T)$) and allow the remaining term to continue training. Training is ultimately interrupted if both early-stopping criteria are met. In all setups, the optimization process can last for a maximum of 500 epochs, and we revert the

[5]Note that for the base and base+ methods, the two terms in (4.37) are functions on shared parameters.

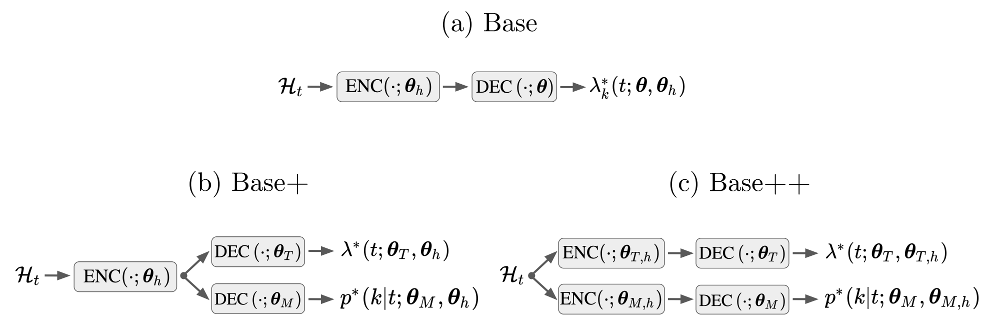

(a) Base
$\mathcal{H}_t$
ENC$(\cdot;\boldsymbol{\theta}_h)$
DEC $(\cdot;\boldsymbol{\theta})$
$\lambda_k^*(t;\boldsymbol{\theta},\boldsymbol{\theta}_h)$
(b) Base+
$\mathcal{H}_t$
ENC$(\cdot;\boldsymbol{\theta}_h)$
DEC $(\cdot;\boldsymbol{\theta}_T)$
$\lambda^*(t;\boldsymbol{\theta}_T,\boldsymbol{\theta}_h)$
DEC $(\cdot;\boldsymbol{\theta}_M)$
$p^*(k|t;\boldsymbol{\theta}_M,\boldsymbol{\theta}_h)$
(c) Base++
$\mathcal{H}_t$
ENC$(\cdot;\boldsymbol{\theta}_{T,h})$
DEC $(\cdot;\boldsymbol{\theta}_T)$
$\lambda^*(t;\boldsymbol{\theta}_T,\boldsymbol{\theta}_{T,h})$
ENC$(\cdot;\boldsymbol{\theta}_{M,h})$
DEC $(\cdot;\boldsymbol{\theta}_M)$
$p^*(k|t;\boldsymbol{\theta}_M,\boldsymbol{\theta}_{M,h})$

Figure 4.3: Graphical representation of the base, base+, and base++ setups.

model parameters to their state with the lowest validation loss after training. We then evaluate the model by computing test metrics on the test sequences of each split.

Finally, to maintain a fair comparison, we ensure that each configuration controls for the number of parameters, keeping them roughly equivalent across settings to confirm that any observed performance improvements are not merely due to increased model capacity. We provide further training details in Appendix D.2.

Metrics. To evaluate the performance of the different baselines on the time prediction task, we report the $\mathcal{L}_T$ term in (4.37) computed over all test sequences. Following Chapter 3, we also quantify the (unconditional) probabilistic calibration of the fitted models by computing the PCE. Finally, we evaluate the MAE in event inter-arrival time prediction. To this end, we predict the next $\tilde{\tau}$ computed from the median of the time predictive distribution, i.e.

$$\tilde{\tau} = (F^*)^{-1}(0.5;\hat{\boldsymbol{\theta}}_T), \tag{4.40}$$

where the quantile function $(F^*)^{-1}(\;\cdot\;;\hat{\boldsymbol{\theta}}_T)$ is estimated from the inverse transform method of Section 2.1.4 using a binary search algorithm. Similarly, for the mark prediction task, we report the average $\mathcal{L}_M$ term in (4.37), and quantify the probabilistic calibration of the mark predictive distribution by computing the ECE (Naeini et al., 2015). Additionally, by predicting the mark of the next event as

$$\tilde{k} = \underset{k\in\mathbb{K}}{\text{argmax}}\; p^*(k|\tau;\hat{\boldsymbol{\theta}}_M), \tag{4.41}$$

we can assess the quality of the point predictions by means of various classification metrics. Specifically, we compute the accuracy@n for values of n in $\{1,5\}$, the MRR (Craswell, 2009), and the F1-score of mark predictions. Lower $\mathcal{L}_T$, $\mathcal{L}_M$, PCE and ECE is better, while higher accuracy@n, MRR and F1-score is better. For both time and mark predictive distributions, we finally report their reliability diagrams marginalized over all events. Refer to Sections 2.1.5 and 3.3.4 for a reminder of these different metrics.

4.4.2 Results and Discussion

We present the $\mathcal{L}_T$ and $\mathcal{L}_M$ metrics for the base, base+, and base++ configurations across all datasets in Table 4.1. The PCE, MAE, ECE, MRR, F1-score, and Accuracy@{1,5} metrics, which reflect the subsequent discussion, are given in Appendix D.3.1.

Table 4.1: $\mathcal{L}_T$ and $\mathcal{L}_M$ results of the different setups across all datasets. The values are computed over 3 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

	$\mathcal{L}_M$				
	LastFM	MOOC	Github	Reddit	Stack O.
THP	714.0 (16.5)	93.3 (1.4)	128.3 (17.3)	**39.9 (1.3)**	105.3 (0.7)
THP+	695.9 (15.7)	76.9 (1.2)	120.8 (16.0)	42.7 (1.2)	103.3 (0.7)
THP++	**651.1 (18.1)**	**70.9 (1.0)**	**112.1 (15.4)**	40.4 (1.3)	**103.0 (0.7)**
STHP+	700.9 (16.6)	83.2 (1.3)	121.4 (15.8)	**43.7 (1.1)**	**104.0 (0.7)**
STHP++	**696.5 (15.9)**	**82.4 (1.4)**	**114.7 (14.9)**	46.2 (0.8)	105.6 (0.6)
SAHP	825.5 (25.6)	163.0 (2.2)	138.0 (19.1)	77.9 (4.7)	108.4 (1.0)
SAHP+	740.0 (24.6)	73.8 (1.0)	116.8 (15.2)	43.4 (1.2)	103.3 (0.7)
SAHP++	**654.8 (17.3)**	**71.0 (1.1)**	**114.1 (15.2)**	**40.5 (0.8)**	**103.1 (0.7)**
LNM	685.2 (15.8)	86.6 (1.2)	116.8 (15.2)	43.4 (1.2)	106.5 (0.7)
LNM+	668.6 (15.8)	77.1 (1.3)	112.3 (15.1)	41.4 (1.1)	**103.2 (0.7)**
LNM++	**637.2 (19.4)**	**73.8 (1.1)**	**111.5 (15.2)**	**40.8 (1.0)**	**103.2 (0.7)**
FNN	739.5 (25.2)	78.8 (1.3)	113.5 (15.4)	47.0 (1.4)	107.3 (0.6)
FNN+	672.1 (17.9)	72.3 (1.1)	111.6 (15.1)	41.2 (0.9)	103.2 (0.7)
FNN++	**648.6 (16.2)**	**71.8 (1.0)**	**109.5 (15.0)**	**40.1 (1.0)**	**103.1 (0.7)**
RMTPP	684.6 (15.6)	87.0 (1.2)	126.4 (18.3)	41.4 (0.9)	106.5 (0.7)
RMTPP+	681.5 (16.3)	74.9 (1.3)	118.7 (15.4)	42.0 (1.0)	**103.0 (0.7)**
RMTPP++	**654.5 (16.7)**	**71.4 (1.1)**	**112.6 (15.3)**	**40.6 (1.2)**	103.1 (0.7)
Hawkes	494.5 (17.9)	114.4 (1.6)	127.1 (25.6)	40.6 (0.7)	113.5 (1.1)

	$\mathcal{L}_T$				
	LastFM	MOOC	Github	Reddit	Stack O.
THP	-945.1 (41.3)	-135.9 (1.3)	-242.5 (38.5)	-72.3 (2.3)	-84.0 (1.4)
THP+	-994.1 (48.8)	-130.6 (1.3)	-255.7 (42.0)	-85.5 (2.3)	**-84.7 (1.4)**
THP++	**-1037.3 (47.0)**	**-136.0 (2.1)**	**-271.0 (50.8)**	**-87.9 (2.0)**	-84.2 (1.4)
STHP+	-993.3 (44.2)	-128.2 (1.0)	-208.9 (32.2)	-76.6 (2.1)	**-83.6 (1.4)**
STHP++	**-1014.9 (42.1)**	**-132.9 (1.8)**	**-237.0 (39.3)**	**-78.2 (2.1)**	-83.3 (1.4)
SAHP	-1263.8 (57.9)	-266.0 (3.5)	-346.5 (57.4)	-72.9 (1.8)	-89.7 (1.4)
SAHP+	**-1320.4 (58.4)**	-288.8 (3.5)	-358.1 (57.6)	-94.8 (2.3)	**-89.9 (1.4)**
SAHP++	-1320.0 (60.5)	**-293.8 (3.6)**	**-366.0 (59.3)**	**-95.4 (2.1)**	-77.4 (1.2)
LNM	-1326.3 (55.8)	-310.0 (3.8)	-380.4 (59.8)	**-96.4 (2.0)**	-91.0 (1.4)
LNM+	-1320.4 (60.5)	**-310.6 (3.7)**	-378.7 (59.4)	**-96.4 (2.0)**	**-91.1 (1.3)**
LNM++	**-1334.7 (58.5)**	-307.6 (3.9)	**-381.2 (59.9)**	-96.3 (2.2)	-90.5 (1.4)
FNN	-1276.2 (58.6)	-280.9 (3.2)	-363.1 (57.2)	-75.3 (2.5)	-81.2 (1.3)
FNN+	**-1324.6 (59.6)**	**-302.2 (3.6)**	-364.5 (57.0)	-94.4 (2.1)	**-90.0 (1.8)**
FNN++	-1324.2 (58.1)	-300.9 (3.7)	**-365.0 (56.7)**	**-96.1 (2.2)**	-88.9 (1.4)
RMTPP	-1052.9 (46.0)	-178.6 (1.8)	-268.0 (54.3)	**-88.0 (2.2)**	-83.3 (1.4)
RMTPP+	-1040.7 (45.8)	**-187.8 (3.1)**	-272.2 (49.0)	-87.9 (2.0)	**-83.4 (1.4)**
RMTPP++	**-1071.1 (50.3)**	-182.2 (1.9)	**-287.2 (52.6)**	-86.6 (2.0)	-82.9 (1.4)
Hawkes	-1166.3 (64.3)	-204.7 (5.4)	-341.2 (89.0)	-75.9 (3.1)	-81.1 (2.0)

Distinct decoders mitigate gradient conflicts. Based on the $\mathcal{L}_T$ and $\mathcal{L}_M$ metrics, we note a consistent improvement when moving from the base to the base+ setting for THP, SAHP, and FNN. This underscores the benefits of using two distinct decoders for the time and mark prediction tasks with base+, leading to improved predictive accuracy compared to the base models. Figure 4.4 shows the distribution of cos ϕ_{TM} during training for THP, SAHP and FNN for both base and base+ settings on all datasets, along with the average GMS and TPI for conflicting gradients. We would like too emphasize that both base and base+ share the *same* encoder architecture, which allows for a direct comparison of the distribution of cos ϕ_{TM} between the two settings during training. Appendix D.3.3 provides detailed visualizations for the remaining baselines.

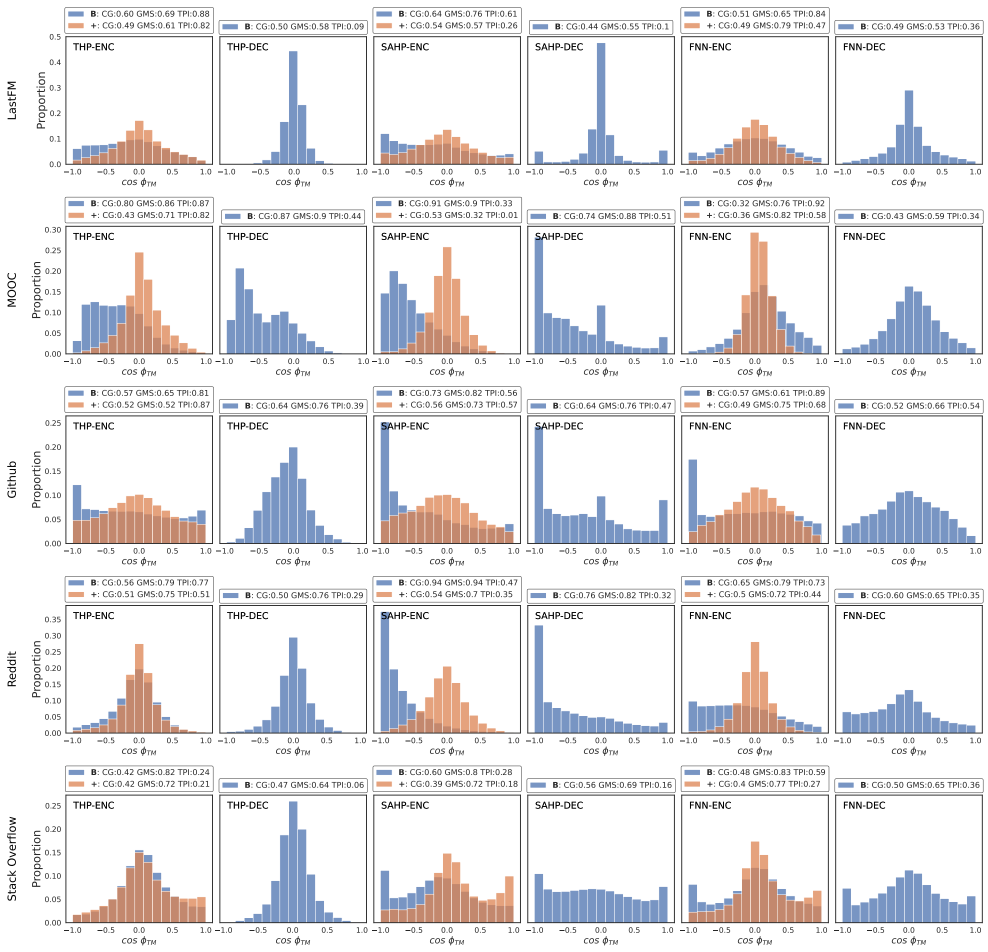


Figure 4.4: Distribution of cos ϕ_{TM} during training at the encoder (ENC) and decoder (DEC) heads for THP, SAHP and FNN in the base and base+ setups. "B" and "+" refer to the base and base+ models, respectively, and the distribution is obtained by pooling the values of ϕ_{TM} over 5 training runs. As the decoders are disjoint in the base+ setting, note that cos ϕ_{TM} is not defined.

With the base model, a significant proportion of severe conflicts (as indicated by cos ϕ_{TM} in [-1, -0.5]) is often observed for the shared parameters of both encoder and decoder heads, typically with low GMS values. Additionally, with the base model, the TPI values suggest that these conflicting gradients at the encoder heads generally favor $\mathcal{T}_T$ (i.e., TPI > 0.5). In contrast, base+ inherently prevents conflicts at the decoder by separating the parameters for each task. Moreover, during training with base+, there is a noticeable reduction in the severity of conflicting gradients for the shared encoder parameters, as evidenced by a more concentrated distribution of $\cos\phi_{TM}$ around 0. The TPI values further indicate that base+ generally achieves a more balanced training between both tasks, which further contributes to enhancing their individual performance. While we note that the GMS values do not consistently improve between the base and base+ settings, improvements with respect to $\mathcal{L}_T$ and $\mathcal{L}_M$ suggest that this effect is offset by a reduction in conflicts during training.

In addition, while LNM and RMTPP already avoid conflicts at the decoder in their base settings by decomposing the parameters, explicitly modeling the dependency of marks on time with base+ further enhances mark prediction performance. Finally, while the Hawkes decoder is generally outperformed on the time and mark prediction tasks by neural models trained in the base+ and base++ setups, we note that it still achieves top performance on the $\mathcal{L}_M$ metric on LastFM. Events in LastFM are usually highly clustered per mark, and we believe that the assumptions of strictly positive and additive influence of past events in the Hawkes model makes it more prone to capture such dynamics.

Disjoint training enhances mark predictive accuracy. Returning to Table 4.1, moving from the base+ to the base++ setting often leads to improvements in the $\mathcal{L}_M$ metric for most baselines, while the $\mathcal{L}_T$ metric generally remains comparable between the two configurations. Although conflicting gradients are typically reduced when moving from base to base+, this pattern indicates that the residual conflicts primarily hinder the mark prediction task. In contrast, base++ effectively eliminates these conflicts by using distinct history representations for each task. A significant advantage of base++ is that it allows one task to continue training after the other has reached convergence. For example, Figure 4.5 illustrates the validation losses $\mathcal{L}_T$ and $\mathcal{L}_M$ for SAHP++ on MOOC. Thanks to disjoint training, the $\mathcal{L}_M$ metric can be further optimized for additional epochs after training of the $\mathcal{L}_T$ metric ceases due to overfitting, thus achieving gains in mark prediction performance. This feature is absent in base and base+, where both $\mathcal{L}_M$ and $\mathcal{L}_T$ metrics rely

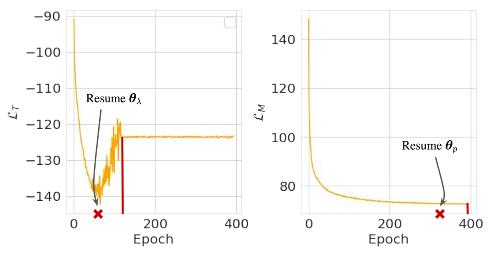
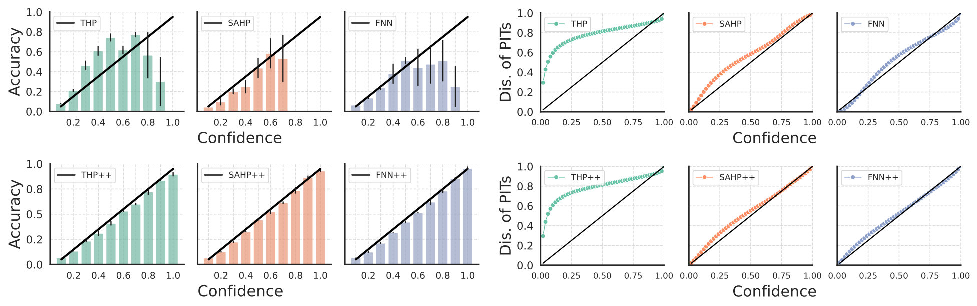


Figure 4.5: Validation curves of the $\mathcal{L}_T$ and $\mathcal{L}_M$ components for SAHP++ on MOOC.

(a) LastFM



(b) MOOC

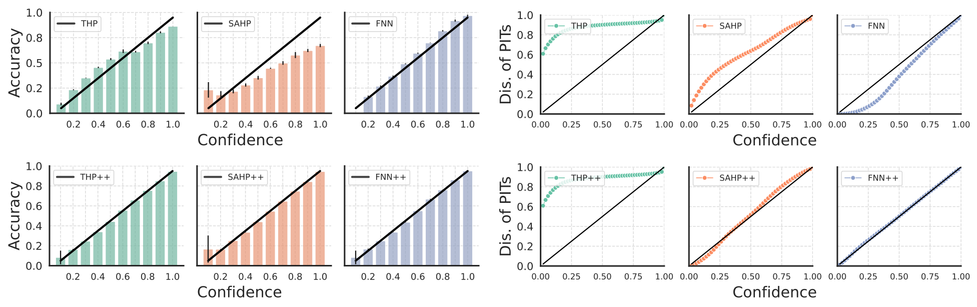


Figure 4.6: Reliability diagrams of the mark (left) and time (right) predictive distributions for THP, SAHP and FNN in the base and base++ settings on LastFM and MOOC. The cumulative distribution of PITs and accuracy aligning with the black diagonal correspond to perfect probabilistic and top-label calibration, respectively. The reliability diagrams are averaged over 5 splits, and error bars refer to the standard error.

on a shared set of parameters. In Figure 4.5, $\boldsymbol{\theta}_T$ is fixed after the vertical red line, resulting in a constant validation $\mathcal{L}_T$ for the remaining training epochs of the model.

Finally, referring back to the preliminary experiment on Figure 4.2, the results with respect to $\mathcal{L}_T$ and $\mathcal{L}_M$ in Table 4.1 suggest that conflicting gradients are mostly detrimental to model performance if (1) they are in great proportion during training (i.e. high CG) and (2) if they are associated to low GMS values. For instance, on Figure 4.2, THP and SAHP exhibit high CG associated to high GMS values, whereas the remaining models conversely show a more balanced CG, but with lower GMS values. We find that model performance often improves in both these scenarios when preventing conflicting gradients altogether in the base++ setting.

Reliability diagrams. Figure 4.6 presents the reliability diagrams of the time and mark predictive distributions for all models in the base and base++ settings on LastFM. The diagrams show that the base++ models are generally better calibrated than their base counterparts, as evidenced by the bin accuracies aligning more closely with the diagonal. This improvement aligns with the results in Table 4.1, where a lower $\mathcal{L}_M$ indicates better accuracy. However, improvements in the probabilistic calibration of arrival times between base and base++ models are less noticeable, suggesting that conflicting gradients during training predominantly affect the mark prediction task. We provide reliability diagrams for other baselines and datasets in Appendix D.3.2.

Isolating the impact of conflicts on performance. Our experiments reveal that the base+ and base++ settings result in a decrease of conflicting gradients and to enhanced performance with respect to the time and mark prediction tasks. However, for THP, SAHP and FNN, these settings do not enable us to disentangle performance gains brought by a decrease in conflicts from the ones brought by modifications of the decoder architecture. To address this limitation, we introduce the following two settings based on the duplicated model approach of Section 4.3.1:

- *Shared History Encoders and Duplicated Decoders.* The functions $\lambda^*(t;\boldsymbol{\theta}_T)/\Lambda^*(t;\boldsymbol{\theta}_T)$ and $p^*(k|t;\boldsymbol{\theta}_M)$ are obtained through (4.14) from two identical parametrizations of the same base decoder. However, similar to the base+ setting, a common history encoder $\boldsymbol{h}$ is used, meaning that these functions still share parameters via $\boldsymbol{\theta}_h$. Models trained in this setting are indicated with a "-D" sign, e.g. THP-D.

- *Disjoint History Encoders and Duplicated Decoders.* This setting differs from the previous one in the use of two distinct history embeddings $\boldsymbol{h}_T$ and $\boldsymbol{h}_M$ in (4.14), implying that $\lambda^*(t;\boldsymbol{\theta}_T)/\Lambda^*(t;\boldsymbol{\theta}_T)$ and $p^*(k|t;\boldsymbol{\theta}_M)$ are now completely disjoint parametrizations. We use the label "-DD" to denote the models trained in this setting, e.g. THP-DD.

In contrast to base+ and base++, the base-D and base-DD settings retain the same architecture as the base model, enabling us to directly evaluate the impact of conflicting gradients on performance. In Table 4.2, we report the performance with respect to $\mathcal{L}_T$ and $\mathcal{L}_M$ for THP, SAHP, and FNN trained in the base, base-D and base-DD settings. We follow the same experimental setup as before, and maintain the number of parameters comparable between the different settings for fair comparison. We almost systematically observe improvements on both tasks when moving from the base to the base-D or

Table 4.2: $\mathcal{L}_T$ and $\mathcal{L}_M$ results for the base, base-D, and base-DD settings across all datasets. The values are computed over 5 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

	$\mathcal{L}_M$ LastFM	$\mathcal{L}_M$ MOOC	$\mathcal{L}_M$ Github	$\mathcal{L}_M$ Reddit	$\mathcal{L}_M$ Stack O.
THP	714.0 (16.5)	93.3 (1.4)	128.3 (17.3)	39.9 (1.3)	105.3 (0.7)
THP-D	677.1 (19.4)	82.8 (1.2)	121.4 (16.8)	39.1 (0.9)	104.7 (0.7)
THP-DD	**607.2 (16.2)**	**79.8 (1.2)**	**113.5 (14.9)**	**38.2 (1.0)**	**104.4 (0.6)**
SAHP	825.5 (25.6)	163.0 (2.2)	138.0 (19.1)	77.9 (4.7)	108.4 (1.0)
SAHP-D	832.1 (32.0)	93.3 (1.5)	128.8 (18.5)	56.0 (1.0)	105.1 (0.7)
SAHP-DD	**692.1 (19.7)**	**89.0 (1.6)**	**115.4 (15.3)**	**51.0 (0.5)**	**104.8 (0.6)**
FNN	739.5 (25.2)	78.8 (1.3)	113.5 (15.4)	**47.0 (1.4)**	107.3 (0.6)
FNN-D	732.8 (19.7)	**76.8 (1.2)**	112.9 (15.4)	56.9 (0.9)	103.8 (0.7)
FNN-DD	**670.0 (18.1)**	79.7 (1.4)	**111.4 (15.3)**	48.8 (1.6)	**103.7 (0.7)**

	$\mathcal{L}_T$ LastFM	$\mathcal{L}_T$ MOOC	$\mathcal{L}_T$ Github	$\mathcal{L}_T$ Reddit	$\mathcal{L}_T$ Stack O.
THP	-945.1 (41.3)	-135.9 (1.3)	-242.5 (38.5)	-72.3 (2.3)	-84.0 (1.4)
THP-D	-995.1 (50.2)	**-164.8 (1.8)**	-258.5 (47.2)	-90.3 (1.9)	**-85.3 (1.4)**
THP-DD	**-1023.8 (53.0)**	-162.1 (4.3)	**-279.0 (55.2)**	**-92.6 (2.1)**	-84.8 (1.4)
SAHP	-1263.8 (57.9)	-266.0 (3.5)	-346.5 (57.4)	-72.9 (1.8)	-89.7 (1.4)
SAHP-D	-1319.1 (57.9)	-288.7 (3.7)	-348.7 (58.2)	**-92.9 (2.3)**	**-90.1 (1.3)**
SAHP-DD	**-1319.9 (59.3)**	**-294.0 (3.6)**	**-367.9 (59.1)**	-87.0 (3.5)	-87.1 (2.6)
FNN	-1276.2 (58.6)	-280.9 (3.2)	**-363.1 (57.2)**	-75.3 (2.5)	-81.2 (1.3)
FNN-D	**-1314.1 (59.7)**	**-286.8 (3.5)**	-360.0 (55.9)	-81.3 (1.7)	**-81.7 (1.3)**
FNN-DD	-1294.4 (57.5)	-286.3 (3.8)	-362.5 (56.4)	**-82.7 (2.2)**	-81.4 (1.4)

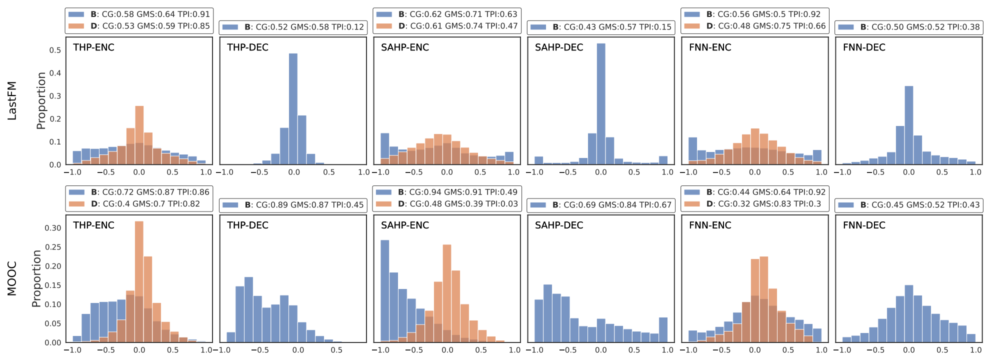
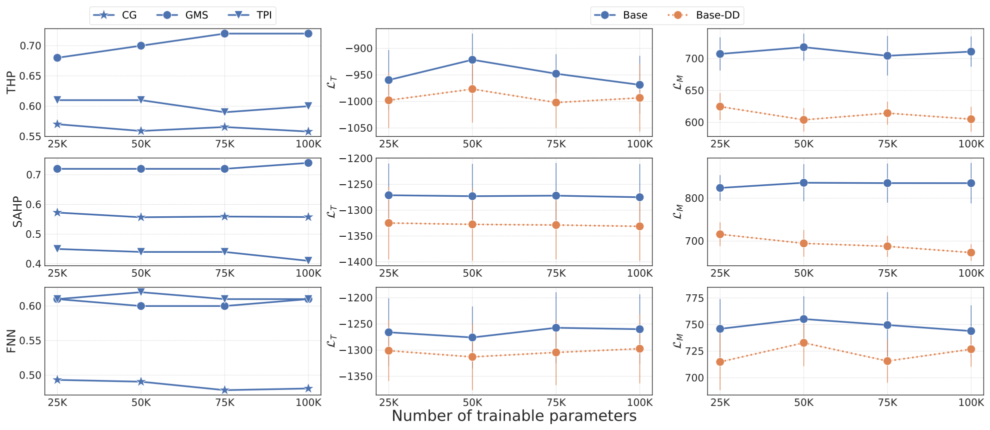


Figure 4.7: Distribution of cos ϕ_{TM} during training at the encoder (ENC) and decoder (DEC) heads for THP, SAHP and FNN in the base and base-D setup on LastFM and MOOC. "B" and "D" refer to the base and base-D models, respectively, and the distribution is obtained by pooling the values of ϕ_{TM} over 5 training runs. As the decoders are disjoint in the base-D setting, note that cos ϕ_{TM} is not defined.

base-DD settings. Moreover, these performance gains are often associated to a decrease in (severe) conflicts during training, as shown on Figure 4.7. Furthermore, when comparing the results between Tables 4.1 and 4.2, we note that our base+ and base++ parametrizations often show improved performance compared to the base-D and base-DD settings, especially on the mark prediction task. This highlights that the benefits of our parametrizations extend beyond the prevention of conflicts to achieve greater predictive performance. We provide more visualizations in Appendix D.3.4.

(a) LastFM



(b) MOOC

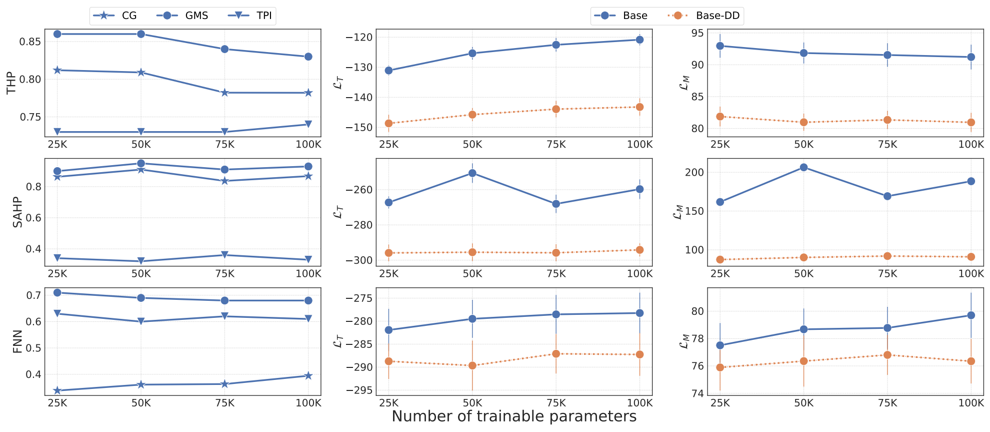


Figure 4.8: Left: Evolution of CG, GMS, TPI with increasing model capacity for the base THP, SAHP and FNN during training on LastFM and MOOC. Right: Evolution of the test $\mathcal{L}_T$ and $\mathcal{L}_M$ in the base and base-DD settings.

Conflicting gradients remain harmful as capacity increases. To assess whether conflicting gradients remain detrimental to predictive performance for higher-capacity models, we train THP, SAHP and FNN in the base and base-DD settings while progressively increasing the number of trainable parameters. Figure 4.8 shows the evolution of the proportion of CG, GMS and TPI during training for these models on LastFM and MOOC, along with the evolution of their test $\mathcal{L}_T$ and $\mathcal{L}_M$ as a function of number of parameters. For each capacity (25K, 50K, 75K and 100K parameters), we maintain the distribution of parameters between the encoder and decoder heads constant at 0.67/0.33. Note that we only report the CG, GMS, and TPI values for the base models, as the base-DD setting is by definition free of conflicts. We note that increasing a model's capacity has a limited impact on the CG, GMS and TPI values, as well as on model performance with respect to both $\mathcal{L}_T$ and $\mathcal{L}_M$. In contrast, differences in performance between the base and base-DD setups are more significant, suggesting that conflicting gradients remain harmful to performance even with increased model capacity.

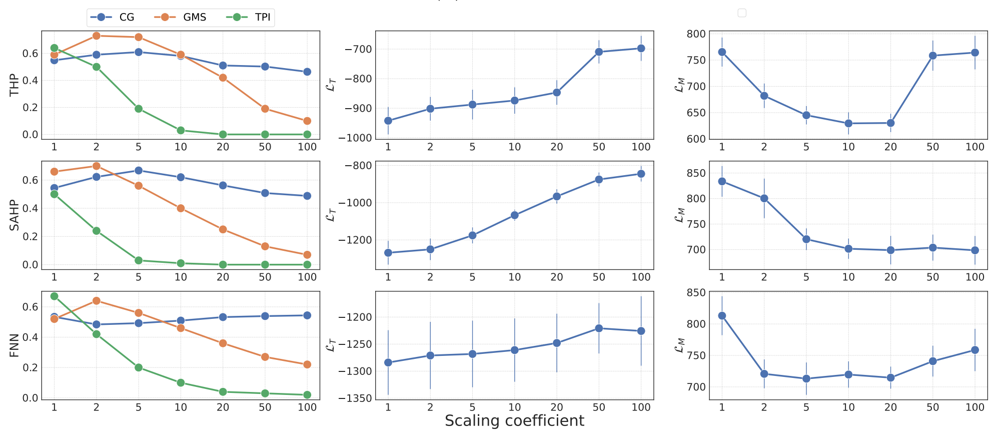
Scaling the loss does not efficiently address conflicts. Our findings in Figure 4.4 suggest that conflicting gradients generally tend to favor $\mathcal{T}_T$ at the encoder heads during optimization, as illustrated by TPI values > 0.5. To better balance tasks during training, a natural approach would consist in scaling the contribution of $\mathcal{T}_T$ in (4.2) to reduce its impact on the overall loss, i.e.

$$\mathcal{L}(\theta; \mathcal{S}_{\text{train}}) = \frac{1}{s}\mathcal{L}_T(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) + \mathcal{L}_M(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}), \tag{4.42}$$

where $s \geq 1$ is a scaling coefficient. To assess the effectiveness of this method, we train the base THP, SAHP and FNN models on the objective in (4.42) following the same experimental setup as before. For these models, we report in Figure (4.9) the evolution of the training CG, GMS and TPI values, along with their unscaled test $\mathcal{L}_T$ and $\mathcal{L}_M$. We observe that the occurrence of conflicting gradients is marginally impacted by larger values of s. However, as scaling increases, the magnitude of $\boldsymbol{g}_T = \frac{1}{s}\nabla\mathcal{L}_T(\mathcal{S}_{\text{train}}; \boldsymbol{\theta})$ diminishes, which translates into decreased values of GMS and TPI. Hence, as s grows, the contribution of $\boldsymbol{g}_T$ to the gradient update at time $s+1$, i.e. $\boldsymbol{\theta}^{s+1} = \boldsymbol{\theta}^s - \eta(\boldsymbol{g}_T + \boldsymbol{g}_M)$, becomes more and more marginal. Consequently, during training, the gradient updates will be mostly governed by $\boldsymbol{g}_M$, leading $\boldsymbol{\theta}$ to converge to a state that is primarily beneficial to the mark prediction task. Nevertheless, the crashed TPI and GMS values are at the root of significant degradation with respect to $\mathcal{L}_T$, offsetting the gains on $\mathcal{L}_M$. Although a specific value of s could lead to a trade-off between tasks, models trained in our base+ or base++ settings generally show improved performance with respect to both tasks simultaneously, as shown in Table 4.1.

Disjoint parametrizations remain efficient at all training sizes. At this stage, a question that remains unanswered relates to the efficiency of the disjoint parametrizations in the base++ setting compared to the base models when the number of training sequences is limited. To answer this, we train FNN and SAHP in the base and base++ settings on LastFM and

(a) LastFM



(b) MOOC

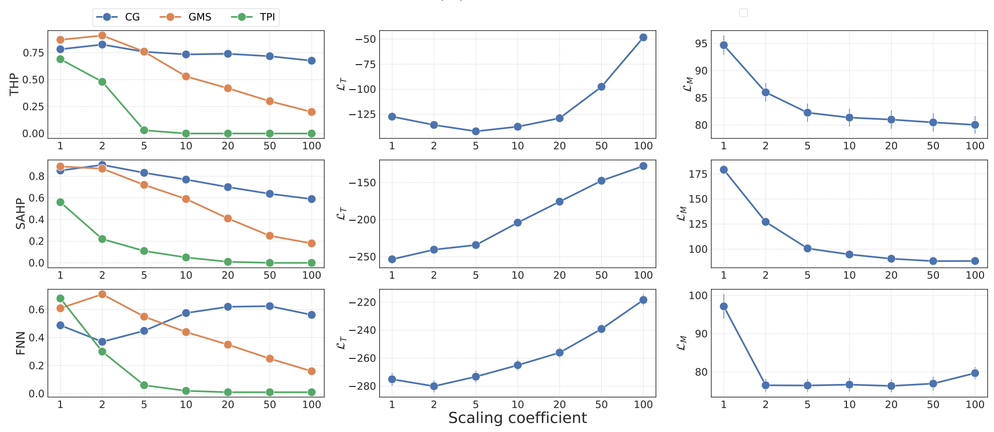


Figure 4.9: Left: Evolution of CG, GMS, TPI with scaling coefficient s for the base THP, SAHP and FNN during training on LastFM. Right: Evolution of the test $\mathcal{L}_T$ and $\mathcal{L}_M$ in the base and base-DD settings.

MOOC on increasing proportions $p\%$ of sequences to which the model has access during training. We maintain the same experimental setup detailed in Section 4.4.1, and Figure 4.10 shows the evolution of the test $\mathcal{L}_T$ and $\mathcal{L}_M$ as a function of $p\%$. As observed, models trained in the base++ setting remain systematically more efficient than their base counterparts on both the time and mark prediction tasks, even at low values of $p\%$. Moreover, all models show increasing performance as the amount of sequences available during training grows, which aligns with expectations.

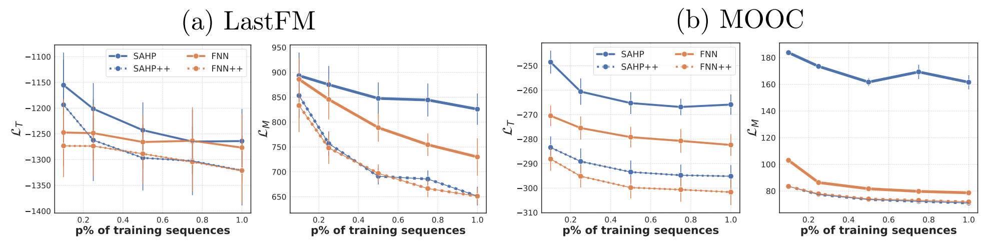


Figure 4.10: Left: Evolution of the test $\mathcal{L}_T$ and $\mathcal{L}_M$ as a function of the proportion of sequences in $\mathcal{S}_{\text{train}}$ kept during training for SAHP and FNN in the base and base++ settings on LastFM and MOOC. The results are averaged over 5 splits, and vertical bars refer to the standard error.

An alternative approach to model the joint distribution. As detailed in Section (4.3.2), our parametrization of LNM+ alleviates the assumption of conditional independence between arrival-times and marks in LNM (Shchur et al., 2020a). Relatedly, Waghmare et al. (2022) also proposed an extension of LNM that relaxes this assumption, although their approach differs from ours in some key aspects. Specifically, their work parametrizes $f^*(\tau|k;\boldsymbol{\theta}_T)$ as a distinct mixture of log-normal distributions for each mark k, and $p^*(k;\boldsymbol{\theta}_M)$ is obtained by removing the temporal dependency in (4.26)[6]. For further reference, we denote this model as LNM-Joint. Although both LNM+ and LNM-Joint aim to model the joint distribution $f^*(\tau, k)$, some conceptual differences separate the two approaches:

(1) By design, LNM-Joint cannot be trained in the base++ setup as it prevents the decomposition of the NLL into disjoint $\mathcal{L}_T$ and $\mathcal{L}_M$ terms. Indeed, suppose that we use two distinct history representations $\boldsymbol{h}_T$ and $\boldsymbol{h}_M$ to parametrize $f^*(\tau|k;\boldsymbol{\theta}_T)$ and $p^*(k;\boldsymbol{\theta}_M)$ respectively, as detailed in Section 4.3.2. Here, $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_M$ are disjoint set of learnable parameters. The NLL of a single training sequence $\mathcal{S} = \{(\tau_i, k_i)\}_{i=1}^{n}$ observed in $[0, T]$ would write

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}_T;\boldsymbol{\theta}_M) = & \underbrace{-\sum_{i=1}^{n} \log\, f^*(\tau_i|k_i;\boldsymbol{\theta}_T) - \log\, \left(1 - F^*\left(T;\boldsymbol{\theta}_T,\boldsymbol{\theta}_M\right)\right)}_{\mathcal{L}_T(\boldsymbol{\theta}_T,\boldsymbol{\theta}_M)} \\ & \underbrace{-\sum_{i=1}^{n} \log\, p^*(k_i;\boldsymbol{\theta}_M)}_{\mathcal{L}_M(\boldsymbol{\theta}_M)}, \qquad (4.43)\end{aligned}$$

[6]They rely on a common history embedding $\boldsymbol{h}$ to parametrize both predictive distributions.

Table 4.3: $\mathcal{L}_T$, $\mathcal{L}_M$ and Accuracy results for LNM+ and LNM-Joint on all datasets. The values are computed over 5 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

	LastFM	MOOC	Github	Reddit	Stack Overflow
			$\mathcal{L}_M$		
LNM+	**668.6 (15.8)**	**77.1 (1.3)**	**112.3 (15.1)**	**41.4 (1.1)**	**103.2 (0.7)**
Joint-LNM	671.3 (17.1)	127.0 (6.4)	117.3 (15.5)	42.9 (0.9)	106.6 (0.7)
			Accuracy		
LNM+	**0.24 (0.01)**	**0.52 (0.0)**	**0.67 (0.01)**	**0.82 (0.0)**	**0.49 (0.0)**
Joint-LNM	0.23 (0.01)	0.23 (0.03)	0.64 (0.01)	**0.82 (0.0)**	0.47 (0.0)
			$\mathcal{L}_T$		
LNM+	-1320.4 (60.5)	**-310.6 (3.7)**	-378.7 (59.4)	**-96.4 (2.0)**	**-91.1 (1.3)**
Joint-LNM	**-1326.2 (58.3)**	-303.9 (3.3)	**-381.0 (59.6)**	-94.3 (2.0)	-90.6 (1.4)

where $F^*(T;\boldsymbol{\theta}_T;\boldsymbol{\theta}_M) = \int_0^{T-t_n}\sum_{k=1}^K f^*(s|k;\boldsymbol{\theta}_T)p^*(k;\boldsymbol{\theta}_M)ds$ depends on both $\boldsymbol{\theta}_T$ and $\boldsymbol{\theta}_M$. Consequently, the NLL cannot be disentangled into disjoint $\mathcal{L}_T$ and $\mathcal{L}_M$ terms, proscribing disjoint training in the base++ setup. Conversely, choosing to parametrize $f^*(\tau;\boldsymbol{\theta}_T)$ and $p^*(k|\tau;\boldsymbol{\theta}_M)$ as done in our framework leads to the decomposition in (4.37) as $F^*(T;\boldsymbol{\theta}_T)$ is solely a function of $\boldsymbol{\theta}_T$.

(2) For LNM-Joint, M mixtures must be defined for each k, leading to $M \times K$ log-normal distributions in total. Conversely, in LNM+, $f^*(\tau;\boldsymbol{\theta}_T)$ does not scale with K, and $p^*(\tau|k;\boldsymbol{\theta}_M)$ requires an equivalent number of parameters as $p^*(k;\boldsymbol{\theta}_M)$ in LNM-Joint.

For completeness, we integrate LNM-Joint in our code base using the original implementation as reference. In Table (4.3), we compare its performance against LNM+ on the time and mark prediction tasks in terms of the $\mathcal{L}_T$, $\mathcal{L}_M$, and accuracy metrics, following the experimental setup detailed previously. We observe improved performance of LNM+ compared to LNM-Joint on the mark prediction task ($\mathcal{L}_M$ and accuracy), and competitive results on the time prediction task ($\mathcal{L}_T$). Despite both approaches modelling the joint distribution, our results suggest that the dependency between arrival times and marks is more accurately captured by $p^*(k|\tau;\boldsymbol{\theta}_M)$ than by $f^*(\tau|k;\boldsymbol{\theta}_T)$.

Computational time. We report in Table (4.4) the average execution time (in seconds) for a single forward and backward pass on all training sequences of all datasets . The results are given for all models in the base, base+ and

Table 4.4: Average execution time (in seconds) for a single forward and backward pass on all training sequences of the MOOC dataset. Results are averaged over 50 epochs.

	LNM			RMTPP			FNN			THP			SAHP			STHP		
	Base	+	++	Base	+	++	Base	+	++	Base	+	++	Base	+	++	Base	+	++
LastFM	2.2	2.23	3.26	1.91	1.94	2.94	4.29	3.66	4.74	2.96	2.57	3.62	4.82	3.42	4.48	3.08	\	4.17
MOOC	3.97	3.99	5.6	3.24	3.33	4.37	6.36	6.28	8.07	5.19	4.15	5.76	8.08	5.91	7.74	5.61	\	6.72
Github	0.34	0.39	0.51	0.34	0.34	0.5	0.76	0.65	0.8	0.38	0.43	0.58	0.51	0.55	0.69	0.49	\	0.64
Reddit	8.74	9.53	11.5	7.99	8.14	11.79	19.13	16.47	20.32	9.75	9.78	13.5	13.63	11.99	15.9	11.59	\	15.54
Stack O.	16.02	16.63	22.98	14.08	14.54	20.31	32.83	28.15	33.95	16.32	16.81	23.1	21.11	20.77	26.45	19.84	\	26.07

base++ setups, and are averaged over 50 epochs. We notice that the computation of two separate embeddings $\boldsymbol{h}_T$ and $\boldsymbol{h}_M$ in the base++ setup inevitably leads to an increase in execution time, which appears more pronounced for larger datasets such as Reddit and Stack Overflow. However, the increased computational complexity is generally offset by improved model performance, as detailed in Table 4.1.

4.5. Conclusion

Learning a neural MTPP model can be essentially interpreted as a two-task learning problem, in which one task is focused on learning a time predictive distribution, while the other concerns learning a mark predictive distribution. Typically, most neural MTPP models implicitly require these two tasks to share a common set of trainable parameters. In this paper, we demonstrate that this parameter sharing leads to the emergence of conflicting gradients during training, often resulting in degraded performance on each individual task. To prevent this issue, we introduce novel parametrizations of neural MTPP models that enable separate modeling and training of each task, effectively preventing the occurrence of conflicting gradients. Through extensive experiments on real-world event sequence datasets, we validate the advantages of our framework over the original model configurations, particularly in the context of mark prediction. However, we acknowledge several limitations in our study. Firstly, our focus was solely on categorical marks. Investigating conflicting gradients in more complex scenarios, such as temporal graphs (Trivedi et al., 2019; Gracious & Dukkipati, 2023) or spatio-temporal point processes (Zhou & Yu, 2023; Zhang et al., 2023), presents a promising avenue for future research. Secondly, our analysis was limited to neural MTPP models trained using the NLL. Extending our framework to other proper scoring rules (Brehmer et al., 2021) is also a potential area for future exploration.

CHAPTER 5

Distribution-Free Conformal Prediction Regions for Neural MTPP Models

In Chapters 3 and 4, we saw that defining a neural MTPP model essentially pertains to specifying a parametrization of either $f^*(\tau, k; \boldsymbol{\theta})$, $\lambda_k^*(t; \boldsymbol{\theta})$, or $\Lambda_k^*(t; \boldsymbol{\theta})$, provided that the chosen parametrization defines a valid distribution over event sequences. Once we have the set of trained parameters $\hat{\boldsymbol{\theta}}$ obtained by optimizing an appropriate objective function, the MTPP model produces an estimate that can be used for downstream prediction tasks on new test sequences. This enables us to address queries of the sort "When is the next event likely to occur?", "What will be the type of the next event, given that it occurs at a certain time t?" or "How long until an event of type k occurs?".

Providing accurate answers to the above queries can be achieved through different means, and constitutes an important inquiry that any competent MTPP model should achieve. For instance, from $f^*(\tau; \hat{\boldsymbol{\theta}})$ and $p^*(k|\tau; \hat{\boldsymbol{\theta}})$, we can predict the arrival time of the next event by computing its expected value, and its mark by taking the label associated to the highest probability at that time. Such predictions are known as *point predictions*, in the sense that they resume all possible future outcomes to a single value. While point predictions provide in general clear interpretation, they face a major limitation: in reporting a single value, they prevent us from effectively assessing the uncertainty of the model in its predictions.

In critical application domains, such as diagnostic medicine, epidemiology, or high-frequency trading, quantifying the uncertainty in the predictions promotes better-informed decision making, thereby limiting the risks of potentially disastrous consequences. From a practitioner perspective, accurate quantification of the uncertainty further contributes to making a model more reliable and interpretable, which in turn encourages trust in the predictions.

Consequently, instead of reporting point predictions for future events, we may want to construct a *prediction region* for the next arrival time, mark, or both, based on a sequence of observed historical events. This region should typically include a subset of potential values that are highly likely to occur, aligned with a predetermined probability coverage level. For instance, suppose that we want to construct a prediction interval R_τ for the next event's inter-arrival time that is guaranteed to contain the true observation with 80% probability. Given $f^*(\tau; \hat{\boldsymbol{\theta}})$, one could naively construct R_τ by taking the interval defined by 0 and the quantile 0.8 of the time predictive distribution, i.e. $R_\tau = [0, Q^*(0.8; \hat{\boldsymbol{\theta}})]$, where $Q^*(\cdot\, ; \hat{\boldsymbol{\theta}})$ is the predictive quantile function of inter-arrival times.

Unfortunately, the above interval would only be guaranteed to meet the desired coverage level if and only if the predictive distribution returned by the MTPP model $f^*(\tau; \hat{\boldsymbol{\theta}})$ matches the true distribution $f^*(\tau)$. Unfortunately, as we have seen in previous chapters, the model may provide a poor approximation of the true unknown underlying process. This limitation can be attributed to various factors, such as model misspecification or lack of training data. Consequently, prediction regions derived solely from the estimates of the model may be *unreliable*, failing to faithfully reflect the true uncertainty.

Building on the framework of conformal prediction (CP) (Vovk et al., 2005), we aim in this chapter to develop more reliable methods for uncertainty quantification in neural MTPP models. As we've seen in Section 2.2.2, CP enables the construction of distribution-free prediction regions, offering a finite-sample coverage guarantee even when the base model is unreliable. Although conformal prediction has been considered in the closely related field of survival analysis (Candès et al., 2023; Gui et al., 2023), these studies have primarily focused on univariate survival times, whereas our problem also involves categorical marks. To our knowledge, this study represents the first attempt to connect the field of neural MTPP models to conformal prediction.

In line with the standard assumption prevalent in the neural MTPP literature, we consider the setting in which a set of observed event sequences are drawn from the ground-truth process. Furthermore, for each event sequence, we construct an input-output pair, where the input is a neural vector representation of the event sequence history, and the output is a bivariate response representing the arrival time and mark of the last event of the sequence. This aligns our approach with the scenario considered in Stankeviciute et al. (2021); however, in that context, the authors focused on regular time series forecasting.

Our primary objective is to generate joint prediction regions for both the event arrival time and mark that are distribution-free and come with a finite-sample coverage guarantee. This entails developing a bivariate conformal prediction region, capable of accommodating both a strictly positive, continuous response and a categorical response with numerous marks, all without depending on distributional assumptions. Figure 5.1 gives a toy example of such prediction region. Unfortunately, the existing literature on conformal prediction for scenarios involving multi-response or mixed response types is rather limited. Moreover, many neural MTPP models typically focus on either estimating the joint density of arrival time and mark, or the marked intensity functions from which it is derived. Methods addressing these aspects in the context of CP are comparatively scarce. Notable contributions in the field of multi-response conformal prediction include Feldman et al. (2023) and Lei et al. (2013). However, Feldman et al. (2023) proposed a method centered on multi-output quantile

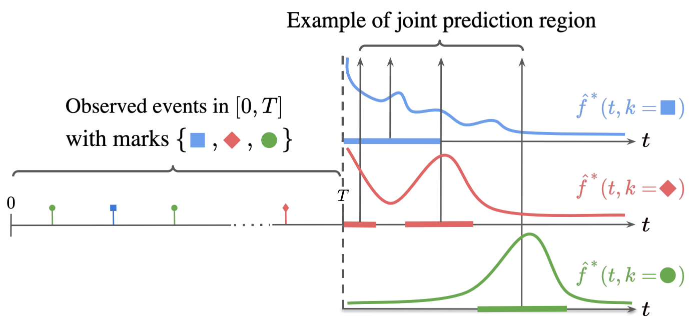


Figure 5.1: Toy illustration of a joint prediction region constructed from the joint density of arrival times and marks. The three colored curves represent predictive density functions while the horizontal bars represent prediction intervals.

regression for continuous random vectors, which does not easily align with our context. On the other hand, while Lei et al. (2013) offers a density-based conformal method, it falls short in addressing estimation problems that involve covariates.

We will first propose a naive method that, despite its simplicity, still offers a finite-sample coverage guarantee. This approach involves combining separate prediction regions for the event inter-arrival time and the mark. However, by neglecting potential dependencies between these variables, this method may be overly conservative. Consequently, it could lead to inflexible and large prediction regions. Such regions, while guaranteeing coverage, may not be tight, failing to accurately reflect the true underlying uncertainty. Next, we will adopt a more effective strategy that accounts for the dependencies between the arrival time and the mark. Specifically, we will construct a Highest Density Region (HDR) Hyndman (1996) based on their joint predictive density. To achieve a conformal coverage guarantee, we will consider a generalization of the univariate HPD-split method (Izbicki et al., 2022) for bivariate responses. In contrast to the naive approach, this method has the advantage of efficiently excluding unlikely combinations of the two variables, while still maintaining the pre-specified coverage level.

Our second objective is to explore conformal prediction methods to generate univariate prediction regions, independently for the arrival time and the mark. Considering the continuous nature of inter-arrival times, our focus will be on conformal regression techniques. We examine both symmetric and asymmetric prediction intervals using conformal quantile regression (Romano et al., 2019), as well as prediction regions derived from conformal density-based methods (Izbicki et al., 2022). Conversely for the mark, we explore conformal classification methods. Here, we intend to investigate conformal methods that involve thresholding the mark-conditional probabilities, thereby creating adaptive prediction sets (Romano et al., 2020; Angelopoulos et al., 2021).

While achieving finite-sample marginal coverage is both desirable and practically feasible, we are also interested in the stronger notion of conditional coverage which requires the desired coverage level to be met conditionally. Although this is not attainable without imposing strong distributional assumptions (Vovk, 2012; Foygel Barber et al., 2021), we will also assess the conformal prediction regions in terms of approximate notions of conditional coverage.

Finally, we will evaluate the validity and efficiency of both the bivariate and univariate conformal prediction methods through an extensive series of experiments on simulated and real-world event sequence datasets. Additionally, we will explore heuristic versions of these methods, which involve substituting the model estimate in the corresponding oracle prediction region. Our evaluation will employ metrics that quantify both the probability coverage and the sharpness of the region, as determined by its length.

5.1. Problem Formulation and Goals

In this chapter, we introduce a slight change of notations compared to the rest of the thesis. We now suppose that we have access to a dataset $\mathcal{D}^* = \{\mathcal{S}_1, ..., \mathcal{S}_n\}$, where each sequence $\mathcal{S}_i = \left\{ \boldsymbol{e}_{i,j} = (t_{i,j}, k_{i,j}))_{j=1}^{m_i} \right\}$ comprises m_i events with arrival times observed within the interval $[0, T]$ and $i = 1, ..., n$.

Moreover, we will drop the "$*$" symbol and explicitly write the dependency of a function on the history embedding $\boldsymbol{h}$, e.g. $f^*(\tau, k; \hat{\boldsymbol{\theta}}) = f(\tau, k|\boldsymbol{h}; \hat{\boldsymbol{\theta}})$. Finally, once the neural MTPP model is trained, we use the symbol "$\hat{}$" to denote an estimate of a function, e.g. $\hat{f}(\tau, k|\boldsymbol{h})$ is an estimate of the true $f(\tau, k|\boldsymbol{h})$, where the dependency on $\hat{\boldsymbol{\theta}}$ is omitted.

Finally, as commonly done in the context of neural MTPP research, we assume that the sequences are *exchangeably* drawn from the underlying ground-truth process. All remaining notations are either retained from the previous chapter, or explicitly defined throughout the text.

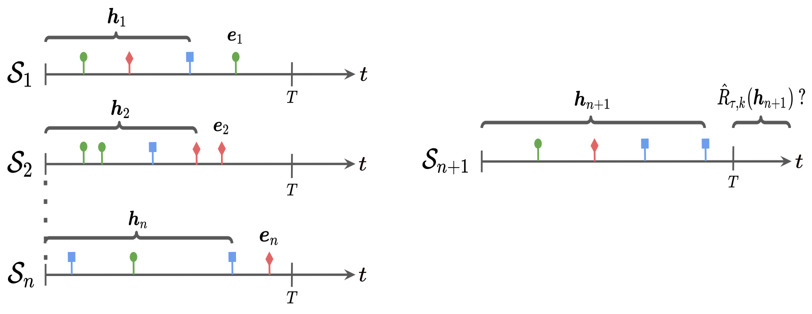


Figure 5.2: Illustration of the input-output pairs $\mathcal{D} = \{ (\boldsymbol{h}_i, \boldsymbol{e}_i) \}_{i=1}^n$ and the joint prediction region we aim to construct for $\boldsymbol{e}_{n+1}$ given a new input $\boldsymbol{h}_{n+1}$.

Let us consider the dataset $\mathcal{D}^*$, which is composed of n sequences assumed to be drawn exchangeably from the ground-truth MTPP process. For each sequence $\mathcal{S}_i \in \mathcal{D}^*$, we define the input-output pair $(\boldsymbol{h}_{i,m_i}, \boldsymbol{e}_{i,m_i})$, where $\boldsymbol{e}_{i,m_i} = (\tau_{i,m_i}, k_{i,m_i})$ is a bivariate response corresponding to the last event in $\mathcal{S}_i$, and $\boldsymbol{h}_{i,m_i}$ is the history embedding associated to it. A similar scenario has been explored by Stankeviciute et al. (2021), but for conformal time series forecasting. From these input-output pairs, we construct the dataset $\mathcal{D} = \{ (\boldsymbol{h}_{i,m_i}, \boldsymbol{e}_{i,m_i}) \}_{i=1}^n$. To simplify notation, we will henceforth denote $(\boldsymbol{h}_i, \boldsymbol{e}_i) = (\boldsymbol{h}_{i,m_i}, \boldsymbol{e}_{i,m_i})$, implying that these quantities are consistently defined for the last event of a sequence $\mathcal{S}_i$.

Given $\mathcal{D}$ and a new test input $\boldsymbol{h}_{n+1}$, our primary goal is to construct an informative distribution-free joint prediction region $\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1}) \subseteq \mathbb{R}_+ \times \mathbb{K}$ for the bivariate pair $\boldsymbol{e}_{n+1} = (\tau_{n+1}, k_{n+1})$ of $\boldsymbol{h}_{n+1}$. This prediction region must achieve finite-sample marginal coverage at level $1-\alpha$, that is

$$\mathbb{P}((\tau_{n+1}, k_{n+1}) \in \hat{R}_{\tau,k}(\boldsymbol{h}_{n+1})) \geq 1 - \alpha. \tag{5.1}$$

Here, the probability is taken over all $n+1$ observations $\mathcal{D} \cup \{(\boldsymbol{h}_{n+1}, \boldsymbol{e}_{n+1})\}$, and the condition must hold true for any chosen values of α and n. Essentially, this entails developing a joint prediction region for a bivariate response, accommodating both a continuous and a categorical response, without relying on strong distributional assumptions. Figure 5.2 illustrates our primary objective given a dataset $\mathcal{D} = \{ \mathcal{S}_i \}_{i=1}^n$ and a new test sequence $\mathcal{S}_{n+1}$.

We will also examine scenarios where we generate individual prediction regions for both the arrival time and the mark. Given that the arrival time is a continuous variable, we will use conformal regression techniques, while for the mark, a categorical variable, conformal classification methods will be used.

For the inter-arrival times, given $\mathcal{D} = \{ (\boldsymbol{h}_i, \tau_i) \}_{i=1}^{n}$ and a new test input $\boldsymbol{h}_{n+1}$, we seek to construct a prediction region $\hat{R}_\tau(\boldsymbol{h}_{n+1}) \subseteq \mathbb{R}_+$ for τ_{n+1} which achieves finite-sample marginal coverage at level $1-\alpha$, that is

$$\mathbb{P}(\tau_{n+1} \in \hat{R}_\tau(\boldsymbol{h}_{n+1})) \geq 1 - \alpha. \tag{5.2}$$

Similarly for the marks, given $\mathcal{D} = \{ (\boldsymbol{h}_i, k_i) \}_{i=1}^{n}$ and a new test input $\boldsymbol{h}_{n+1}$, we want to generate a prediction set $\hat{R}_k(\boldsymbol{h}_{n+1}) \subseteq \mathbb{K}$ for k_{n+1} which achieve finite-sample marginal coverage at level $1-\alpha$, that is

$$\mathbb{P}(k_{n+1} \in \hat{R}_k(\boldsymbol{h}_{n+1})) \geq 1 - \alpha. \tag{5.3}$$

Finally, beyond ensuring a finite-sample coverage guarantee, it is essential that the prediction regions are informative, which implies striving for the smallest possible regions. To accomplish this, we will adopt the split conformal prediction framework Papadopoulos et al. (2002), a widely used variant of CP known for its reduced computational demands. This method, which involves partitioning the data, is relatively simple but effective in transforming any heuristic notion of uncertainty into a rigorous one (Angelopoulos & Bates, 2023). It enables the construction of distribution-free prediction regions that achieve finite-sample coverage guarantees. We elaborate on this methodology in the following paragraphs.

Consider $\mathcal{D} = \{ (\boldsymbol{h}_i, \boldsymbol{y}_i) \}_{i=1}^{n}$, a dataset consisting of n exchangeable pairs. In the context of our problem setup, the response $\boldsymbol{y}_i \in \mathcal{Y}$ varies according to the scenario: it can be bivariate as $\boldsymbol{y}_i = \boldsymbol{e}_i$ with $\mathcal{Y} = \mathbb{R}_+ \times \mathbb{K}$, or univariate as either $\boldsymbol{y}_i = \tau_i$ or $\boldsymbol{y}_i = k_i$, with $\mathcal{Y} = \mathbb{R}_+$ or $\mathcal{Y} = \mathbb{K}$, respectively. Additionally, we have access to a MTPP model that provides a heuristic measure of uncertainty $\hat{g}$ for $\boldsymbol{y}$ given $\boldsymbol{h}$. Recall from Section 2.2.2 that the split conformal procedure to generate a prediction region for a new observation $\boldsymbol{y}_{n+1}$ at coverage level $1-\alpha$ can be summarized in the following steps:

1. Split $\mathcal{D}$ into two non-overlapping sets, $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{cal}}$ with $\mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{cal}} = \mathcal{D}$.

2. Fit the MTPP model to the observations in $\mathcal{D}_{\text{train}}$, yielding a heuristic measure of uncertainty $\hat{g}$ for $\boldsymbol{y}$ given $\boldsymbol{h}$.

3. Use $\hat{g}$ to define a non-conformity score function $s\,(\boldsymbol{h}, \boldsymbol{y}) \in \mathbb{R}$ that assigns larger value to worse agreement between $\boldsymbol{h}$ and $\boldsymbol{y}$.

4. Compute the calibration scores using the observations in $\mathcal{D}_{\text{cal}}$, i.e.
$$\{ s_i \}_{i=1}^{|\mathcal{D}_{\text{cal}}|} := \{ s(\boldsymbol{h}, \boldsymbol{y}) : (\boldsymbol{h}, \boldsymbol{y}) \in \mathcal{D}_{\text{cal}} \}. \tag{5.4}$$

5. Compute the $1-\alpha$ empirical quantile of these calibration scores:
$$\hat{q} = \text{Quantile}\left(s_1, ..., s_{|\mathcal{D}_{\text{cal}}|} \cup \{ \infty \} ; \frac{\lceil (|\mathcal{D}_{\text{cal}}| + 1)(1-\alpha) \rceil}{|\mathcal{D}_{\text{cal}}|} \right). \tag{5.5}$$

6. For a new test input $\boldsymbol{h}_{n+1}$, use $\hat{q}$ to construct a prediction region for $\boldsymbol{y}_{n+1}$ with a $1-\alpha$ coverage level as follows:
$$\hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_{n+1}) = \{ \boldsymbol{y} \in \mathcal{Y} : s(\boldsymbol{h}_{n+1}, \boldsymbol{y}) \leq \hat{q} \}. \tag{5.6}$$

As discussed in Section 2.2.2, we can show that this region satisfies
$$\mathbb{P}(\boldsymbol{y}_{n+1} \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_{n+1})) = \mathbb{P}(s(\boldsymbol{h}_{n+1}, \tau_{n+1}) \leq \hat{q}) \geq 1-\alpha. \tag{5.7}$$

In other words, the marginal coverage guarantees in (5.1), (5.2) and (5.3) are satisfied. Moreover, if no ties between the scores occur with probability one, we can further show that this marginal coverage is upper bounded, i.e.
$$1-\alpha \leq \mathbb{P}(\boldsymbol{y}_{n+1} \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_{n+1})) \leq 1-\alpha + \frac{1}{|\mathcal{D}_{\text{cal}}| + 1}. \tag{5.8}$$

Finally, while marginal coverage is a desirable and practically achievable property, we are also interested in the stronger notion of conditional coverage:
$$\mathbb{P}(\boldsymbol{y}_{n+1} \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_{n+1}) \mid \boldsymbol{h}_{n+1}) \geq 1-\alpha \quad \forall \, \boldsymbol{h}_{n+1}, \tag{5.9}$$

which requires the desired coverage level $1-\alpha$ to be met for all $\boldsymbol{h}_{n+1}$. Despite (5.9) not being achievable without strong distributional assumptions (Vovk, 2012; Foygel Barber et al., 2021), we still aspire for the prediction regions to achieve approximate notions of conditional coverage. To meet such desiderata, Section 5.3.1 and Section 5.3.2 explore conformal scores to achieve the guarantees in (5.2) and (5.3), respectively. Similarly, Section 5.4 seeks conformal scores to attain the joint guarantee in (5.1).

5.2. Related Work

Conformal prediction for temporal data. Our work builds upon Conformal Prediction (CP), first introduced by Vovk et al. (1999). CP is a powerful tool in machine learning for providing reliable uncertainty estimates. Angelopoulos & Bates (2023) offer a modern introduction, while Shafer & Vovk (2008) present a more classical perspective. Our research specifically focuses on the split-conformal prediction method (Papadopoulos et al., 2002). In the context of temporal data, CP has seen significant recent development. Gibbs & Candès (2021) and Zaffran et al. (2022) proposed methods to adapt CP for sequential data shifts, continuously adjusting an internal coverage target. Stankeviciute et al. (2021) extended CP to time series, and considered multi-step predictions, assuming exchangeability in individual time series. Conversely, a branch of research led by Tibshirani et al. (2019); Foygel Barber et al. (2021) and Xu & Xie (2023a) challenges the exchangeability assumption by applying weighted samples. This approach, while offering stronger uncertainty estimates by leveraging similar past instances, results in a weaker conformal guarantee. Although conformal prediction has been explored in the closely related field of survival analysis (Candès et al., 2023; Gui et al., 2023), these studies have primarily focused on univariate survival times.

For continuous variables, our work builds on Romano et al. (2019), which adjusts quantile regression estimates, and Izbicki et al. (2022), which outputs regions in the form of HDR for univariate responses. For discrete variables, we consider Sadinle et al. (2019), which minimizes the average prediction set length, and Romano et al. (2020) and Angelopoulos et al. (2021), which demonstrate good conditional coverage.

Multi-response conformal prediction. Our study also intersects with the field of multi-output CP. Sun & Yu (2023) introduced CopulaCPTS, applying CP to time series with multivariate targets and adapting the calibration set in each step based on a copula of the target variables. Feldman et al. (2023) used a deep generative model to learn a unimodal representation of the response, allowing for the application of multiple-output quantile regression on this learned lower dimensional representation. This method generates flexible and informative regions in the response space, a capability not present in earlier methods.

5.3. Individual Prediction Regions for Arrival Times and Marks

In this section, we outline the methods for generating individual prediction regions $\hat{R}_\tau(\boldsymbol{h}_{n+1})$ and $\hat{R}_k(\boldsymbol{h}_{n+1})$ for τ_{n+1} and k_{n+1}, respectively. As τ_{n+1} is a continuous variable, we rely on conformal regression techniques to construct $\hat{R}_\tau(\boldsymbol{h}_{n+1})$. Conversely, k_{n+1} being a categorical variable, we leverage conformal classification approaches to build $\hat{R}_k(\boldsymbol{h}_{n+1})$.

5.3.1 Constructing a Prediction Region for the Arrival Time

Using a dataset $\mathcal{D} = \{\ (\boldsymbol{h}_i, \tau_i)\ \}_{i=1}^n$, our objective is to construct a prediction region $\hat{R}_\tau(\boldsymbol{h}_{n+1}) \subseteq \mathbb{R}_+$ for the arrival time τ_{n+1} of a new test input $\boldsymbol{h}_{n+1}$. This prediction region must achieve finite-sample marginal coverage at level $1-\alpha$, as given in (5.2). An intuitive approach is to create an equal-tailed prediction interval using conditional quantiles at levels $\alpha/2$ and $1-\alpha/2$. Let $\hat{Q}_\tau(\cdot|\boldsymbol{h})$ be the predictive quantile function of τ given $\boldsymbol{h}$ trained using $\mathcal{D}$. We can define a symmetric prediction interval for τ_{n+1} as:

$$\hat{R}_\tau(\boldsymbol{h}_{n+1}) = [\hat{Q}_\tau(\alpha/2|\boldsymbol{h}_{n+1}), \hat{Q}_\tau(1-\alpha/2|\boldsymbol{h}_{n+1})], \tag{5.10}$$

However, as previously mentioned, there is no guarantee that the estimate $\hat{Q}_\tau(\cdot|\boldsymbol{h}_{n+1})$ is a good approximation of the true $Q_\tau(\cdot|\boldsymbol{h}_{n+1})$, resulting in no finite-sample coverage guarantee. By adjusting (5.10), Conformalized Quantile Regression (C-QR) can provide a symmetric prediction interval with a finite-sample coverage guarantee (see Theorem 2 in (Romano et al., 2019)). For a symmetric prediction interval given by (5.10), the C-QR nonconformity score can be defined as

$$s_{\text{CQR}}(\boldsymbol{h}, \tau) = \max\left(\hat{Q}_\tau(\alpha/2|\boldsymbol{h}) - \tau, \tau - \hat{Q}_\tau(1-\alpha/2|\boldsymbol{h})\right), \tag{5.11}$$

and accounts for both potential undercoverage and overcoverage from the model. Indeed, the further τ falls *outside* of the interval $\hat{R}_\tau(\boldsymbol{h}_{n+1})$ in (5.10), the greater is the positive value of s_{CQR}. Conversely, s_{CQR} decreases the further τ correctly falls *within* the interval $\hat{R}_\tau(\boldsymbol{h}_{n+1})$. After evaluating s_{CQR} on hold-out calibration samples and computing $\hat{q}$ using (5.5), we construct a valid prediction interval for τ_{n+1} as:

$$\hat{R}_{\tau,\text{CQR}}(\boldsymbol{h}_{n+1}) = [\hat{Q}_\tau(\alpha/2|\boldsymbol{h}_{n+1}) - \hat{q}, \hat{Q}_\tau(1-\alpha/2|\boldsymbol{h}_{n+1}) + \hat{q}], \tag{5.12}$$

which satisfies marginal coverage at level $1-\alpha$ since

$$\mathbb{P}(\tau_{n+1} \in \hat{R}_\tau(\boldsymbol{h}_{n+1})) = \mathbb{P}(s_{\text{CQR}}(\boldsymbol{h}_{n+1}, \tau_{n+1}) \leq \hat{q}) \geq 1-\alpha. \tag{5.13}$$

However, since the strictly positive arrival times often show a skewed distribution with a significant concentration of probability mass close to 0, this method would not encompass the high-density region between levels 0 and $\alpha/2$, and thus would lead to unnecessarily large intervals. Therefore, a more effective strategy involves generating an asymmetric interval extending from level 0 to α, where the lower bound of the interval remains fixed at 0, and the upper bound, or the right tail, is independently adjusted. This translates into the following asymmetric prediction interval for τ_{n+1}:

$$\hat{R}_\tau(\boldsymbol{h}_{n+1}) = [0, \hat{Q}_\tau(1-\alpha|\boldsymbol{h}_{n+1})], \tag{5.14}$$

for which we define a Conformalized Quantile Regression Left (C-QRL) nonconformity score, expressed as:

$$s_{\text{CQRL}}(\boldsymbol{h}, \tau) = \tau - \hat{Q}_\tau(1-\alpha|\boldsymbol{h}). \tag{5.15}$$

Naturally, this score inherits the same interpretation as the one of C-QR, and leads to the following asymmetric prediction region after estimating $\hat{q}$ on the calibration samples:

$$\hat{R}_{\tau,\text{CQRL}}(\boldsymbol{h}_{n+1}) = [0, \hat{Q}_\tau(1-\alpha|\boldsymbol{h}_{n+1}) + \hat{q}], \tag{5.16}$$

Additionally, it is worth noting that both s_{CQR} and s_{CQRL} can be directly computed from the ground compensator. Noting $\hat{\tau}_\alpha = \hat{Q}_\tau(\alpha|\boldsymbol{h})$ and supposing that t_{j-1} is the last observed arrival time in $\mathcal{H}_t$, we have

$$\begin{aligned} & F(\hat{\tau}_\alpha) = \alpha && (5.17)\\ \iff & 1 - \exp[-\Lambda(t_{j-1} + \hat{\tau}_\alpha|\boldsymbol{h})] = \alpha && (5.18)\\ \iff & \hat{\tau}_\alpha = \hat{Q}_\tau(\alpha|\boldsymbol{h}) = \hat{\Lambda}^{-1}\left(-\log\,(1-\alpha)|\boldsymbol{h}\right) - t_{j-1}. && (5.19) \end{aligned}$$

Moreover, if the estimators of the quantiles are consistent (that is, they converge to the true conditional quantiles as the sample size increases), C-QR and C-QRL have asymptotic conditional coverage, and therefore (5.9) will hold approximately if n is large (Sesia & Candès, 2020, Corollary 1). As alternatives to C-QR and C-QRL for constructing prediction intervals for τ_{n+1}, one could instead leverage the approaches of Conformalized Histogram Regression (CHR) (Sesia & Romano, 2021) or HPD-Split (Izbicki et al., 2022). While we expand further on HPD-split in a later section, we leave CHR as inquiry for future work.

5.3.2 Constructing a Prediction Set for the Mark

Using a dataset $\mathcal{D} = \{ (\boldsymbol{h}_i, k_i) \}_{i=1}^{n}$, our objective is to construct a prediction set $\hat{R}_k(\boldsymbol{h}_{n+1}) \subseteq \mathbb{K}$ for the mark k_{n+1} of a new test input $\boldsymbol{h}_{n+1}$. This prediction set must achieve finite-sample marginal coverage at level $1-\alpha$, as given in (5.3).

Let $\hat{p}(\cdot|\boldsymbol{h})$ denote the predictive PMF of k given $\boldsymbol{h}$, trained using $\mathcal{D}$. To generate a prediction set for k_{n+1}, a simple method involves ranking the marks in descending order by their associated conditional probabilities and retaining those marks where the cumulative sum of these probabilities is less than or equal to the pre-specified probability coverage. However, as previously mentioned, there is no guarantee that we have a good approximation of the true conditional probabilities $p(\cdot|\boldsymbol{h})$.

Furthermore, MTPPs often involve a large number of marks, e.g. up to $K = 50$ in our experiments. Yet in practice, only a few of these marks hold significant probability. Identifying and focusing on these high-probability marks is essential as it leads to more informative prediction sets. This is the rationale behind the method of conformal Adaptive Prediction Sets (APS) (Romano et al., 2020). We focus on the more recent method of Reguralized Adaptive Prediction Sets (RAPS) (Angelopoulos et al., 2021), which consistently generates prediction sets of smaller size than APS by introducing regularization. Specifically, noting $r(k) = |\{ k' \in \mathbb{K} : \hat{p}(k' \mid \boldsymbol{h}) \geq \hat{p}(k|\boldsymbol{h}) \}|$ as the ranking of the observed mark k among the probabilities in $\hat{p}(\cdot|\boldsymbol{h})$ and $(x)^{+}$ as the positive part of x, RAPS defines the following nonconformity score:

$$s_{\text{RAPS}}(\boldsymbol{h}, k) = \sum_{k' : \hat{p}(k'|\boldsymbol{h}) > \hat{p}(k|\boldsymbol{h})} \hat{p}(k'|\boldsymbol{h}) + u \cdot \hat{p}(k|\boldsymbol{h}) + \gamma \left(r(k) - k_{\text{reg}} \right)^{+}, \quad (5.20)$$

where u is a uniform random variable, while $\gamma, k_{\text{reg}} \geq 0$ are regularization parameters. Essentially, the uniform variable u handles the fact that the term $\sum_{k' : \hat{p}(k'|\boldsymbol{h}) > \hat{p}(k|\boldsymbol{h})} \hat{p}(k'|\boldsymbol{h})$ jumps discretely with each mark k', which would lead to too conservative sets. Conversely, the randomized term $u \cdot \hat{p}(k|\boldsymbol{h})$ allows for the random removal of the least probable mark in each prediction set, leading to exactly $1 - \alpha$ coverage (Romano et al., 2020). Moreover, in (5.20), the regularization term $\gamma \left(r(k) - k_{\text{reg}} \right)^{+}$ helps promote smaller set sizes compared to the ones returned by APS. As discussed in Angelopoulos et al. (2021), this is because APS can be sensible to the noisy probability estimates that are deep down the tail of $\hat{p}(\cdot|\boldsymbol{h})$, and whose ordering is often determined by random chance. In contrast, the regularization term in RAPS applies a strong penalty to these noisy probabilities, hence avoiding the inclusion of their associated mark to the returned prediction sets. Nonetheless, note that the nonconformity score of APS can be easily recovered by setting $\gamma = 0$ in (5.20). Minus the randomization and regularization terms, the RAPS score essentially computes the cumulative sum of mark probabilities that are greater or equal to the probability of the observed ground-truth mark k. If we had knowledge of the true PMF of marks, we could construct a prediction set for k_{n+1} as

$$\hat{R}_k(\boldsymbol{h}_{n+1}) = \left\{ k' \in \mathbb{K} : s_{\text{RAPS}}(\boldsymbol{h}_{n+1}, k') \leq 1 - \alpha \right\}, \tag{5.21}$$

that would meet the required marginal coverage of $1 - \alpha$. Instead, the split conformal procedure introduced in Section 5.1 first computes the RAPS scores on a hold-out calibration set $\mathcal{D}_{\text{cal}}$. Then, having computed the adjusted $1 - \alpha$ quantile $\hat{q}$ for these scores from (5.5), we construct the following prediction set for k_{n+1}:

$$\hat{R}_k(\boldsymbol{h}_{n+1}) = \left\{ k' \in \mathbb{K} : s_{\text{RAPS}}(\boldsymbol{h}_{n+1}, k') \leq \hat{q} \right\}, \tag{5.22}$$

which satisfies the desired marginal coverage guarantee at level $1 - \alpha$ since

$$\mathbb{P}(k_{n+1} \in \hat{R}_k(\boldsymbol{h}_{n+1})) = \mathbb{P}(s_{\text{RAPS}}(\boldsymbol{h}_{n+1}, k_{n+1}) \leq \hat{q}) \geq 1 - \alpha. \tag{5.23}$$

Finally, it is worth noting that the APS/RAPS scores can be derived from the marked intensities by recovering the marginal conditional probabilities using the following definition:

$$\hat{p}(k|\boldsymbol{h}) = \mathbb{E}_\tau \left[\hat{p}(k|\tau, \boldsymbol{h}) \right] = \mathbb{E}_\tau \left[\frac{\hat{\lambda}_k(t_{j-1} + \tau|\boldsymbol{h})}{\hat{\lambda}(t_{j-1} + \tau|\boldsymbol{h})} \right]. \tag{5.24}$$

5.4. Joint Prediction Regions for the Arrival Times and Marks

Working with a dataset $\mathcal{D} = \{ (\boldsymbol{h}_i, \boldsymbol{e}_i) \}_{i=1}^n$ where $\boldsymbol{e}_i = (\tau_i, k_i)$, our aim is to construct an informative, distribution-free bivariate joint prediction region $\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1}) \in \mathbb{R}_+ \times \mathbb{K}$ for the pair (τ_{n+1}, k_{n+1}) associated with a new test input $\boldsymbol{h}_{n+1}$. This prediction region should satisfy finite-sample marginal coverage at level $1 - \alpha$, as given by (5.1). Essentially, this involves generating a joint prediction region for a bivariate response, which integrates a continuous and a categorical variable, without relying on distributional assumptions.

In the following section, we will first explore a naive yet statistically sound method that combines individual prediction regions for the event arrival time and the mark, as outlined in Section 5.3. However, by neglecting potential dependencies between these variables, this method can be overly conservative, resulting in large prediction regions which would not reflect the true underlying uncertainty. A more effective strategy involves jointly predicting the event arrival time and the mark, which will better reflect the true distribution of these two variables. The associated joint prediction region can then exclude unlikely combinations of the two, while still attaining the pre-specified coverage level.

As discussed in Section 5.2, the body of literature on conformal prediction for multi-response scenarios is limited, with notable contributions including Feldman et al. (2023) and Lei et al. (2013). However, Feldman et al. (2023) propose a method centered on multi-output quantile regression for continuous random vectors, which is not easily applicable in our context. Additionally, while Lei et al. (2013) does present a density-based conformal method, it is not suited for estimation problems involving covariates. Instead, we explore an adaptation of the univariate HPD-split method (Izbicki et al., 2022) for bivariate responses. This method enables us to construct highest density regions using the joint predictive density of event arrival time and mark.

5.4.1 Combining Individual Conformal Prediction Regions

Let $\hat{R}_\tau(\boldsymbol{h}_{n+1}) \subseteq \mathbb{R}_+$ and $\hat{R}_k(\boldsymbol{h}_{n+1}) \subseteq \mathbb{K}$ represent the prediction regions for the arrival time τ_{n+1} and the mark k_{n+1}, respectively, for a new test input $\boldsymbol{h}_{n+1}$, as described in Section 5.3. Based on Bonferroni correction, the nominal coverage level for these two regions, specifically the right-hand side of equations (5.2) and (5.3), is set to $1 - {}^{\alpha}\!/_{2}$. Then, the joint prediction region $\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1}) = \hat{R}_\tau(\boldsymbol{h}_{n+1}) \times \hat{R}_k(\boldsymbol{h}_{n+1}) \subseteq \mathbb{R}_+ \times \mathbb{K}$ for (τ_{n+1}, k_{n+1}) obtained by combining these two regions has coverage at least $1 - \alpha$. In fact, by the union bound, we have:

$$\mathbb{P}((\tau_{n+1}, k_{n+1}) \in \hat{R}_\tau(\boldsymbol{h}_{n+1}) \times \hat{R}_k(\boldsymbol{h}_{n+1})) \tag{5.25}$$

$$= \mathbb{P}(\tau_{n+1} \in \hat{R}_\tau(\boldsymbol{h}_{n+1}) \cap k_{n+1} \in \hat{R}_k(\boldsymbol{h}_{n+1})) \tag{5.26}$$

$$= 1 - \underbrace{\mathbb{P}(\tau_{n+1} \notin \hat{R}_\tau(\boldsymbol{h}_{n+1}) \cup k_{n+1} \notin \hat{R}_k(\boldsymbol{h}_{n+1}))}_{\leq {}^{\alpha}\!/_{2} + {}^{\alpha}\!/_{2}} \tag{5.27}$$

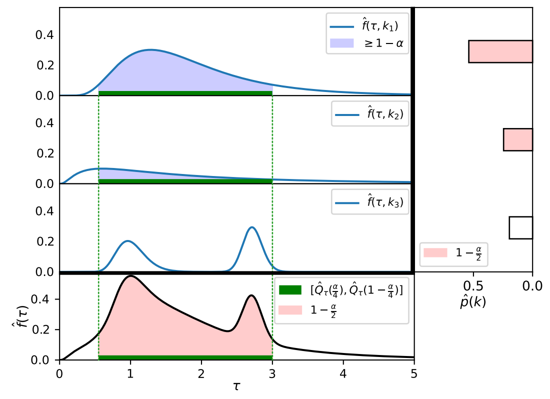
$$\geq 1 - \alpha. \tag{5.28}$$

However, this method, which treats τ and k separately, can be overly conservative, resulting in large and inflexible prediction regions. Indeed, the joint prediction region generated by this approach yields equal length prediction intervals for the arrival times across all selected marks, i.e.:

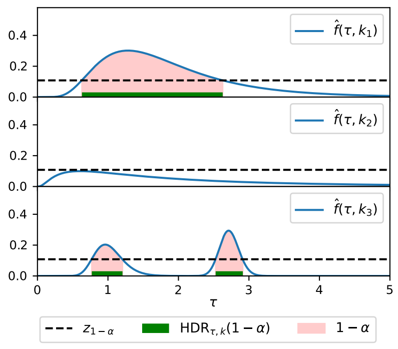
$$\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1}) = \{(\tau', k')|\tau' \in \hat{R}_{\tau}(\boldsymbol{h}_{n+1}), k' \in \hat{R}_k(\boldsymbol{h}_{n+1})\} \tag{5.29}$$

$$= \{\tau'|\tau' \in \hat{R}_{\tau}(\boldsymbol{h}_{n+1})\} \times \{k'|k' \in \hat{R}_k(\boldsymbol{h}_{n+1})\}. \tag{5.30}$$

In other words, for each of the selected marks in $\hat{R}_k(\boldsymbol{h}_{n+1})$, the same prediction interval is constructed for τ_{n+1}. In the following, we refer to this approach as Conformal Independent (C-IND). Fig. 5.3a shows an example of such prediction region, using CQR for the time and APS for the mark. First, the region $\hat{R}_{\tau}(\boldsymbol{h}_{n+1})$ is constructed for τ_{n+1} at level $1 - \frac{\alpha}{2}$ (bottom of Fig. 5.3a) and the region $\hat{R}_k(\boldsymbol{h}_{n+1})$ is constructed for k_{n+1}, also at level $1 - \frac{\alpha}{2}$ (right side of Fig. 5.3a). Finally, $\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1})$ is obtained by taking the cartesian product between the two regions (middle of Fig. 5.3a).



(a) The joint region obtained by combining individual regions, each with coverage $1 - \frac{\alpha}{2}$, has a coverage of at least $1 - \alpha$.



(b) The joint prediction region produced by HDR is composed of different regions $\hat{R}_{\tau}^{(k_1)}(\boldsymbol{h}_{n+1})$ and $\hat{R}_{\tau}^{(k_3)}(\boldsymbol{h}_{n+1})$ depending on the mark. It also demonstrates the ability to exclude marks by producing an empty region $\hat{R}_{\tau}^{(k_2)}(\boldsymbol{h}_{n+1})$.

Figure 5.3: Example of joint bivariate prediction regions with $\alpha = 0.4$ on a synthetic example with $\tau \in \mathbb{R}_+$ and marks $\mathbb{K} = \{k_1, k_2, k_3\}$.

5.4.2 Conformal Highest Joint Density Regions

A better strategy for generating a joint prediction region for the arrival time and the mark involves leveraging their joint distribution. By doing so, this approach excludes unlikely combinations of the two variables, while achieving the pre-specified coverage level. To accomplish this, we propose to compute the HDR Hyndman (1996) with a nominal coverage level of $1 - \alpha$, based on the joint density of the arrival time and mark. Let $\hat{f}(\tau, k|\boldsymbol{h}_{n+1})$ denote this predictive joint density for a new test point $\boldsymbol{h}_{n+1}$. The HDR of $\hat{f}$ with nominal coverage level $1 - \alpha$ is defined as:

$$\text{HDR}(1-\alpha|\boldsymbol{h}_{n+1}) = \left\{ (\tau, k) \;\middle|\; \hat{f}(\tau, k|\boldsymbol{h}_{n+1}) \geq z_{1-\alpha} \right\}, \tag{5.31}$$

where

$$z_{1-\alpha} = \sup \left\{ z' \;\middle|\; \mathbb{P}(\hat{f}(\tau, k|\boldsymbol{h}_{n+1}) \geq z') \geq 1 - \alpha \right\}. \tag{5.32}$$

It is important to highlight that in cases where the underlying univariate distribution exhibits multimodality, an HDR approach will result in a union of intervals that, collectively, are shorter in length than a single interval with the same coverage level. Specifically, the oracle HDR has the useful property of generating the smallest possible region that guarantees conditional coverage (Hyndman, 1996).

Moreover, in contrast to the C-IND method outlined in the previous section, an HDR approach can produce prediction regions for the arrival time that vary in length across different selected marks. Specifically, the joint HDR can be expressed as

$$\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1}) = \text{HDR}(1-\alpha|\boldsymbol{h}_{n+1}), \tag{5.33}$$

$$= \bigcup_{k' \in \hat{R}_k(\boldsymbol{h}_{n+1})} \{(\tau', k')|\tau' \in \hat{R}_\tau^{(k')}(\boldsymbol{h}_{n+1})\}, \tag{5.34}$$

where

$$\hat{R}_\tau^{(k)}(\boldsymbol{h}_{n+1}) = \{\tau'|\hat{f}(\tau', k|\boldsymbol{h}_{n+1}) \geq z_{1-\alpha}\}. \tag{5.35}$$

and

$$\hat{R}_k(\boldsymbol{h}_{n+1}) = \{k'|\exists\ \tau \in \mathbb{R}_+ : \hat{f}(\tau, k'|\boldsymbol{h}_{n+1}) \geq z_{1-\alpha}\}. \tag{5.36}$$

In simpler terms, $\hat{R}_k(\boldsymbol{h}_{n+1})$ encompasses all marks $k \in \mathbb{K}$ for which $\hat{f}(\tau, k|\boldsymbol{h}_{n+1})$ exceeds $z_{1-\alpha}$ over any non-zero interval in $\mathbb{R}_+$. Subsequently, for each $k' \in \hat{R}_k(\boldsymbol{h}_{n+1})$, $\hat{R}_\tau^{(k)}(\boldsymbol{h}_{n+1})$ contains the range of τ values where the joint distribution $\hat{f}(\tau, k|\boldsymbol{h}_{n+1})$ surpasses the threshold $z_{1-\alpha}$. This shows that the HDR is capable of adapting to the joint distribution of the arrival time and mark, leading to more tailored and potentially more efficient prediction regions than (5.30).

Figure 5.3b illustrates the HDR for a simplified example with $\mathbb{K} = \{k_1, k_2, k_3\}$. The various prediction sets constituting the HDR are as follows: $\hat{R}_k(\boldsymbol{h}_{n+1}) = \{k_1, k_3\}$, referring to the selected marks; $\hat{R}_\tau^{(k_1)}(\boldsymbol{h}_{n+1}) = [0.8, 2.6]$ and $\hat{R}_\tau^{(k_3)}(\boldsymbol{h}_{n+1}) = [0.8, 1.2] \cup [2.5, 2.9]$, referring to the arrival time intervals associated to marks k_1 and k_3, respectively. Finally, $\hat{R}_\tau^{(k_2)}(\boldsymbol{h}_{n+1}) = \emptyset$, indicating that mark k_2 does not belong to the prediction region in this scenario.

Unfortunately, the heuristic joint prediction region presented in (5.34) does not come with a finite-sample coverage guarantee. To address this, we modify the nominal coverage level $1 - \alpha$ of the HDR by using a generalization of the univariate Highest Predictive Density (HPD)-split conformal procedure (Izbicki et al., 2022). We refer to this approach as Conformal Highest Density Regions (C-HDR).

Let $\hat{q} \in [0, 1]$. For a new test input $\boldsymbol{h}_{n+1}$, and as per the definition of HDR in (5.31), it holds that

$$(\tau_{n+1}, k_{n+1}) \in \mathrm{HDR}(\hat{q}|\boldsymbol{h}_{n+1}) \iff \mathrm{HPD}(\tau_{n+1}, k_{n+1}|\boldsymbol{h}_{n+1}) \leq \hat{q},$$

where

$$\mathrm{HPD}(\tau, k|\boldsymbol{h}) = \sum_{k' \in \mathbb{K}} \int_{\{\, \tau' \,\mid\, \hat{f}(\tau', k'|\boldsymbol{h}) \geq \hat{f}(\tau, k|\boldsymbol{h}) \,\}} \hat{f}(\tau', k'|\boldsymbol{h}) d\tau,$$

effectively calculates the probability coverage of pairs (τ', k') having a higher density than (τ, k). The C-HDR method defines nonconformity scores based on HPD values,

$$s_{\mathrm{HPD}}(\boldsymbol{h}, (\tau, k)) = \mathrm{HPD}(\tau, k|\boldsymbol{h}), \tag{5.37}$$

and returns a joint HDR with an adjusted nominal coverage level $\hat{q}$, computed as the $1 - \alpha$ empirical quantile of the s_{HPD} scores evaluated on $\mathcal{D}_{\mathrm{cal}}$, i.e.

$$\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1}) = \mathrm{HDR}(\hat{q}|\boldsymbol{h}_{n+1}). \tag{5.38}$$

Furthermore, the quantile lemma ensures that this prediction region verifies the conformal guarantee at nominal level $1 - \alpha$, i.e.

$$\mathbb{P}((\tau_{n+1}, k_{n+1}) \in \hat{R}_{\tau,k}(\boldsymbol{h}_{n+1})) = \mathbb{P}(s_{\mathrm{HPD}}(\boldsymbol{h}_{n+1}, (\tau_{n+1}, k_{n+1})) \leq \hat{q}) \geq 1 - \alpha. \tag{5.39}$$

Moreover, if the estimator of $\hat{f}$ is consistent (that is, it converges to the true f as the sample size increases), the C-HDR nonconformity scores are approximately independent of the input features, which leads to asymptotic conditional coverage. That is, (5.9) will hold approximately if n is large (Izbicki et al., 2022, Theorem 27). Moreover, under these assumptions, C-HDR will also converge to the smallest prediction region that achieves conditional coverage of $1 - \alpha$ (Izbicki et al., 2022, Theorem 27).

5.5. Experiments

5.5.1 Datasets and Baselines

Datasets. We assess the validity and statistical efficiency of the prediction regions produced by various CP methods, as detailed in Sections 5.3 and 5.4. We base our evaluation on 8 real-world marked event sequence datasets that have already been considered in Chapters 3 and 4. Specifically, we use **LastFM** (Hidasi & Tikk, 2012) (records of people listening to songs), **MOOC** (Kumar et al., 2019) (students' actions on an online learning platform), **Github** (developers' actions on Github) (Trivedi et al., 2019), **Reddit** (users' actions of the social platform Reddit) (Kumar et al., 2019), **Retweets** (Zhao et al., 2015) (sequence of retweets following an initial tweet), **Stack Overflow** (badge received by users on Stack Overflow) (Du et al., 2016), **MIMIC2** (Du et al., 2016) (electronic health records of patients), and **Wikipedia** (Kumar et al., 2019) (sequences of edits to Wikipedia pages). For all datasets, we follow the pre-processing steps described in the previous chapters. For a recap on summary descriptions and dataset statistics, we refer the reader to Appendix A.

In addition to the real-world data, we also create a synthetic **Hawkes** dataset containing 14,408 sequences from a multi-dimensional Hawkes process with exponential kernels using the parameters specified in Appendix A.2. For all datasets (real and simulated), we randomly divided the sequences into training ($\mathcal{D}_{\text{train}}$), validation ($\mathcal{D}_{\text{val}}$), calibration ($\mathcal{D}_{\text{cal}}$), and test ($\mathcal{D}_{\text{test}}$) splits, in proportions of 65%, 10%, 15%, and 10%, respectively. Finally, we want to point out that the Github, MIMIC2, and Wikipedia datasets contain a short number of sequences, which will amount to a limited number of observations in the calibration and test splits.

Neural MTPP models. We consider several neural MTPP models to estimate the joint density of event arrival time and mark, represented as $\hat{f}(\tau, k|\boldsymbol{h})$. From these models, we will derive a heuristic-based measure of uncertainty. Specifically, we consider as baselines LNM+ (Bosser & Ben Taieb, 2023b), RMTPP (Du et al., 2016), FNN (Omi et al., 2019) and SAHP (Zhang et al., 2020). Refer to Sections 3.1.3 and 4.4 for a reminder on their respective parametrizations. For all models, we obtain an event representation by concatenating the sinusoidal encoding in (3.3) and the mark embedding in (3.5). Finally, we generate a history representation $\boldsymbol{h} \in \mathbb{R}^{d_h}$ by sequentially processing the events' representations using the GRU encoder in (3.10).

5.5.2 Heuristic and Conformal Prediction Methods

We first focus on creating distinct univariate prediction regions for the event arrival time and the event mark of new test inputs. This is achieved through the application of conformal regression and classification techniques, as detailed in Section 5.3. Additionally, we explore CP methods for constructing bivariate prediction regions for both the event arrival time and its associated mark, as described in Section 5.4. Finally, we also consider heuristic versions, which correspond to non-conformal versions of these methods, by simply replacing the model estimate in the corresponding oracle prediction region. We provide a summary of these methods below.

Prediction regions for the event arrival time. We explore various methods to generate a prediction region for τ_{n+1} of a test input $\boldsymbol{h}_{n+1}$, targeting marginal coverage $1-\alpha$. For the heuristic methods, the first baseline is *Heuristic Quantile Regression (H-QR)*, which constructs a symmetric interval centered at the median:

$$\hat{R}_{\tau,\text{H-QR}}(\boldsymbol{h}_{n+1}) = [\hat{Q}_{\tau}(\alpha/2|\boldsymbol{h}_{n+1}), \hat{Q}_{\tau}(1-\alpha/2|\boldsymbol{h}_{n+1})]. \tag{5.40}$$

The second baseline is *Heuristic Quantile Regression Left (H-QRL)*, which generates an asymmetrical interval with the left bound at zero:

$$\hat{R}_{\tau,\text{H-QRL}}(\boldsymbol{h}_{n+1}) = [0, \hat{Q}_{\tau}(1-\alpha|\boldsymbol{h}_{n+1})]. \tag{5.41}$$

The third baseline is *Heuristic Highest Density Regions (H-HDR)* which forms a HDR, i.e.

$$\hat{R}_{\tau,\text{H-HDR}}(\boldsymbol{h}_{n+1}) = \{\tau | \hat{f}(\tau|\boldsymbol{h}_{n+1}) \geq z_{1-\alpha}\}, \tag{5.42}$$

where $z_{1-\alpha} = \sup\left\{ z' \,\middle|\, \mathbb{P}(\hat{f}(\tau|\boldsymbol{h}_{n+1}) \geq z') \geq 1-\alpha \right\}$. Unlike the HDR defined in (5.31) for a joint prediction on the time and mark, this method represents a univariate HDR, specifically focusing on the arrival time.

We also consider conformal versions of these approaches, denoted as C-QR, C-QRL, and C-HDR, with prediction regions defined in equations (5.12), (5.16), and (5.38), respectively. Additionally, we analyze *Conformal Constant (C-CONST)*, a simple conformal baseline. Its nonconformity score is defined as $s_{\text{C}}(\boldsymbol{h}, \tau) = \tau$ and it generates prediction regions of the form $\hat{R}_\tau(\boldsymbol{h}_{n+1}) = [0, \hat{q}]$, independent of the model and history $\boldsymbol{h}$, where $\hat{q}$ is defined by the split-conformal prediction algorithm in (5.5).

Prediction sets for the event mark. We explore various methods to generate a prediction set for k_{n+1} given a test input $\boldsymbol{h}_{n+1}$. The first baseline methods, called *Heuristic Adaptive Prediction Sets (H-APS)* and *Heuristic Reguralized Adaptive Prediction Sets (H-RAPS)*, generate the following sets:

$$\hat{R}_{k,\text{H-APS}}(\boldsymbol{h}_{n+1}) = \left\{ k' \in \mathbb{K} : s_{\text{APS}}(\boldsymbol{h}_{n+1}, k') \leq 1-\alpha \right\}, \tag{5.43}$$

and

$$\hat{R}_{k,\text{H-RAPS}}(\boldsymbol{h}_{n+1}) = \left\{ k' \in \mathbb{K} : s_{\text{RAPS}}(\boldsymbol{h}_{n+1}, k') \leq 1-\alpha \right\}. \tag{5.44}$$

For their conformal counterparts, we derive prediction regions as detailed in (5.22) and the unregularized C-APS algorithm described in (Romano et al., 2020). Recall that s_{APS} is recovered from s_{RAPS} by setting $\gamma = 0$ in (5.20). Additionally, we explore the *Conformal Probability (C-PROB)* baseline, introduced in Sadinle et al. (2019). This baseline defines its nonconformity score in terms of the estimated probability mass function over the mark:

$$s_{\text{C-PROB}}(\boldsymbol{h}, k) = 1 - \hat{p}(k|\boldsymbol{h}), \tag{5.45}$$

which yields the following prediction region after computing $\hat{q}$ with the split-conformal prediction algorithm (5.5):

$$\hat{R}_{k,\text{C-PROB}}(\boldsymbol{h}_{n+1}) = \left\{ k' \in \mathbb{K} : \hat{p}(k' \mid \boldsymbol{h}) \geq 1-\hat{q} \right\}. \tag{5.46}$$

Moreover, to avoid generating empty prediction sets, the mark associated to the highest estimated probability is systematically included for all methods that we consider, namely H-APS, H-RAPS, C-PROB, C-APS and C-RAPS.

Bivariate prediction regions for the arrival time and mark. We explore two methodologies to construct a bivariate prediction region, $\hat{R}_{\tau,k}(\boldsymbol{h}_{n+1})$, for the pair (τ_{n+1}, k_{n+1}). The first method combines individual univariate prediction regions, as detailed in Section 5.4.1. For this method, we develop two variants, each based on the specific construction of $\hat{R}_{\tau}(\boldsymbol{h}_{n+1})$ and $\hat{R}_{k}(\boldsymbol{h}_{n+1})$. The first variant, C-QRL-RAPS, combines C-QRL for $\hat{R}_{\tau}(\boldsymbol{h}_{n+1})$ and C-RAPS for $\hat{R}_{k}(\boldsymbol{h}_{n+1})$. The second variant, C-HDR-RAPS, uses C-HDR for $\hat{R}_{\tau}(\boldsymbol{h}_{n+1})$, while still employing C-RAPS for $\hat{R}_{k}(\boldsymbol{h}_{n+1})$. The second method generates joint HDR regions, as described in Section 5.4.2. In parallel, we also examine their heuristic counterparts, referred to as H-QRL-RAPS, H-HDR-RAPS, and H-HDR, respectively.

5.5.3 Experimental Setup

Consistently with previous chapters, each model is trained by minimizing the average NLL in (4.1) across training sequences contained in $\mathcal{D}_{\text{train}}$. For optimization, we use mini-batch gradient descent with the Adam optimizer (Kingma & Ba, 2014) and a learning rate of $\eta = 10^{-3}$. The models are trained for at most 500 epochs, and training is interrupted through an early-stopping procedure if there is no improvement in NLL on the validation dataset $\mathcal{D}_{\text{val}}$ for 100 consecutive epochs. In such instances, the model's parameters revert to the state where the validation loss was lowest.

With a trained MTPP model, we are able to calculate the necessary prediction functions for computing prediction regions, as described in Section 5.5.2, for all test inputs within $\mathcal{D}_{\text{test}}$. For the CP methods, the non-conformity scores are computed on $\mathcal{D}_{\text{cal}}$. To compute individual prediction regions for the arrival time and the mark, it's essential to compute the predictive marginals, $\hat{f}(\tau|\boldsymbol{h})$ and $\hat{p}(k|\boldsymbol{h})$, respectively.

To derive $\hat{f}(\tau|\boldsymbol{h})$, we sum over the joint density for each mark, as follows: $\hat{f}(\tau|\boldsymbol{h}) = \sum_{k=1}^{K} \hat{f}(\tau, k|\boldsymbol{h})$. Meanwhile, $\hat{p}(k|\boldsymbol{h})$ is approximated through integration over the positive real line:

$$\hat{p}(k|\boldsymbol{h}) = \int_{\mathbb{R}^+} \hat{f}(s, k|\boldsymbol{h}_{n+1}) ds = \mathbb{E}_{\tau}[\hat{p}(k|\tau, \boldsymbol{h})] \simeq \frac{1}{N} \sum_{s=1}^{N} \hat{p}(k|\tau_s, \boldsymbol{h}), \tag{5.47}$$

where $N = 100$ samples τ_s are generated from $\hat{f}(\tau|\boldsymbol{h})$. This sampling is achieved with the inverse transform sampling method described in Section 2.1.4 from a binary search algorithm.

5.5.4 Evaluation Metrics

We assess the prediction regions $\hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)$, generated for every input $\boldsymbol{h}_i \in \mathcal{D}_{\text{test}}$, using metrics that quantify the probability coverage and the length of each region. The empirical *Marginal Coverage (MC)* is calculated as

$$\text{MC} = \hat{\mathbb{P}}_{\mathcal{D}_{\text{test}}}\left(\boldsymbol{y}_i \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)\right) = \frac{1}{|\mathcal{D}_{\text{test}}|}\sum_{i=1}^{|\mathcal{D}_{\text{test}}|} \mathbb{1}\left[\boldsymbol{y}_i \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)\right]. \tag{5.48}$$

Essentially, this metric calculates the proportion of instances where $\hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)$ contains $\boldsymbol{y}_i$ across all observations in $\mathcal{D}_{\text{test}}$. The *average length* of the prediction regions is computed as

$$\text{Length} = \frac{1}{|\mathcal{D}_{\text{test}}|}\sum_{i=1}^{|\mathcal{D}_{\text{test}}|} |\hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)|, \tag{5.49}$$

where $|\cdot|$ denotes the length of a region. Specifically, for univariate prediction regions, if $\boldsymbol{y} = \tau$, $|\cdot|$ represents the length of prediction intervals or the cumulative length in the case of union of intervals. When $\boldsymbol{y} = k$, it is the cardinality of the discrete prediction set. For bivariate prediction regions, where $\boldsymbol{y} = (\tau, k)$, the calculation differs based on the method. For naive prediction regions as defined in (5.30), the length is given by

$$|\hat{R}_{\tau,k}(\boldsymbol{h}_i)| = |\hat{R}_{\tau}(\boldsymbol{h}_i)| * |\hat{R}_{k}(\boldsymbol{h}_i)|.$$

For the bivariate HDRs as defined in (5.34), the length is calculated as:

$$|\hat{R}_{\tau,k}(\boldsymbol{h}_i)| = \sum_{k' \in \hat{R}_k(\boldsymbol{h}_i))} |\hat{R}_{\tau}^{(k')}(\boldsymbol{h}_i))|.$$

We also consider the *geometric mean of the lengths* computed on $\mathcal{D}_{\text{test}}$ as

$$\text{G. Length} = \frac{1}{|\mathcal{D}_{\text{test}}|}\sum_{i=1}^{|\mathcal{D}_{\text{test}}|} \log(|\hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)| + \epsilon), \tag{5.50}$$

where ϵ is an offset that we fix at $\epsilon = 0.01$ to avoid values of $-\infty$ when $|\hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i)| = 0$. For a better comparison, when comparing a set of M conformal methods with average lengths $L_1, \ldots, L_M$, we report the *relative length* of the ith method as:

$$\text{R. Length} = \frac{L_i}{\min_{j \in \{1,\ldots,M\}} L_j}.$$

We also consider an approximate measure of conditional coverage using the *Worst Slab Coverage (WSC)* metric, as introduced in Cauchois et al. (2021). The WSC metric evaluates the lowest coverage across all slabs $\boldsymbol{v} \in \mathbb{R}^{d_h}$, each containing at least a proportion δ of the total mass, where $0 < \delta \leq 1$. Given $\boldsymbol{v} \in \mathbb{R}^{d_h}$, $\mathrm{WSC}_{\boldsymbol{v}}$ is defined as follows:

$$\begin{aligned}\mathrm{WSC}_{\boldsymbol{v}} = &\inf_{a<b} \Bigg\{ \hat{\mathbb{P}}_{\mathcal{D}_{\text{test}}} \left(\boldsymbol{y}_i \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i) \middle| \; a \leq \boldsymbol{v}^\intercal \boldsymbol{h}_i \leq b \right) \\ &\qquad \text{s.t.} \;\; \hat{\mathbb{P}}_{\mathcal{D}_{\text{test}}}(a \leq \boldsymbol{v}^\intercal \boldsymbol{h}_i \leq b) \geq \delta \Bigg\}, \end{aligned} \tag{5.51}$$

where $a, b \in \mathbb{R}$. This quantity assesses the conditional coverage by conditioning on the history encodings $\boldsymbol{h}_i$ which have a certain level of similarity with the slab $\boldsymbol{v}$ where the similarity is measured by the dot product $\boldsymbol{v}^\intercal \boldsymbol{h}_i$. To estimate the worst slab, we follow Cauchois et al. (2021) and draw 1000 samples $\boldsymbol{v}_j \in \mathbb{R}^d$ uniformly in the simplex $\mathbb{S}^{d-1}$ and compute the slab with minimum conditional coverage as:

$$\mathrm{WSC} = \min_{\boldsymbol{v}_j \in \mathbb{S}^{d-1}} \mathrm{WSC}_{\boldsymbol{v}_j}. \tag{5.52}$$

In practice, to avoid biases due to overfitting on the test dataset, we follow Romano et al. (2020); Sesia & Romano (2021) and first divide the test set in two parts $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{test}}^{(1)} \cup \mathcal{D}_{\text{test}}^{(2)}$. Then, we compute the worst combination of a, b and $\boldsymbol{v}$ on $\mathcal{D}_{\text{test}}^{(1)}$ according to the minimum $\mathrm{WSC}(\boldsymbol{h}_i)$ metric with $\delta = 0.2$[1], and evaluate conditional coverage on $\mathcal{D}_{\text{test}}^{(2)}$.

We further assess (approximate) conditional calibration using the input space partitioning approach from the CD-split$^+$ method detailed in Izbicki et al. (2022), which we call *Conditional Coverage Error (CCE)*. Instead of the Cramér–von Mises distance, we consider the 2-Wasserstein distance, which we estimate via samples. Let Z represent the random variable corresponding to the HPD values. The 2-Wasserstein distance, comparing two random variables with quantile function $F_Z^{-1}(\cdot \mid \boldsymbol{h}_a)$ and $F_Z^{-1}(\cdot \mid \boldsymbol{h}_b)$, is expressed as:

$$d_Z(\boldsymbol{h}_a, \boldsymbol{h}_b) = \left(\int_0^1 \left| F_Z^{-1}(s \mid \boldsymbol{h}_a) - F_Z^{-1}(s \mid \boldsymbol{h}_b) \right|^2 ds \right)^{\frac{1}{2}}. \tag{5.53}$$

In practice, we approximate this distance by generating two samples $Z_1^{(a)}, \ldots, Z_m^{(a)}$ and $Z_1^{(b)}, \ldots, Z_m^{(b)}$, each with m observations conditional on $\boldsymbol{h}_a$ and $\boldsymbol{h}_b$, respectively. Based on the observation that the order statistic $Z_{(i)}$ of

[1]This is the default value employed by the authors in Cauchois et al. (2020).

a sample $Z_1, \ldots, Z_m$ approximates the quantile function $F_Z^{-1}\left(\frac{i}{m+1}\right)$, we can approximate $d_Z(\boldsymbol{h}_a, \boldsymbol{h}_b)$ using

$$d_Z(\boldsymbol{h}_a, \boldsymbol{h}_b) \simeq \left(\sum_{i=1}^{m} \left| Z_{(i)}^{(a)} - Z_{(i)}^{(b)} \right|^2 \right)^{\frac{1}{2}}. \tag{5.54}$$

Using this distance function, we calculate centroids $C_1, \ldots, C_J \in \mathbb{R}^{d_h}$ by applying the k-means++ clustering algorithm on $\mathcal{D}_{\text{cal}}$. Then, we consider a partition $\mathcal{A}$ of $\mathbb{R}^{d_h}$ defined as $\boldsymbol{h} \in A_j$ if and only if $d_Z(\boldsymbol{h}, C_j) < d_Z(\boldsymbol{h}, C_k)$ for every $k \neq j$.

In Appendix E.6, we show that we have to further adapt the distance function from d_Z to $d_{\log Z}$ since distributions with the longest tails exhibit extreme distances from other distributions, often resulting in their isolation into small clusters. By focusing on the random variable $\log Z$ instead of Z, we achieve more balanced cluster sizes, which is crucial for enhancing the accuracy of conditional coverage estimation. In our experiments, we additionally use $J = 4$ centroids to ensure an accurate estimation of conditional coverage per cluster. Finally, the CCE is defined as:

$$\text{CCE} = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{i=1}^{|\mathcal{D}_{\text{test}}|} \sum_{j=1}^{J} \left(\hat{\mathbb{P}}_{\mathcal{D}_{\text{test}}} \left(\boldsymbol{y}_i \in \hat{R}_{\boldsymbol{y}}(\boldsymbol{h}_i) \;\middle|\; \boldsymbol{h}_i \in A_j \right) - (1-\alpha) \right)^2. \tag{5.55}$$

Ideally, the MC and WSC metrics should align with the nominal coverage $1 - \alpha = 0.8$, while the relative length, geometric length, and CCE metrics should be minimized.

5.5.5 Results and Discussion

We first detail the results for individual prediction regions for the arrival time and the mark in Sections 5.5.5.1 and 5.5.5.2, respectively. Subsequently, the results for the joint prediction regions are presented in Section 5.5.5.3. Our primary focus is on a probability miscoverage level of $\alpha = 0.2$. Following this, we show the results at various other coverage levels in Section 5.5.5.4. In this section, we focus on the neural MTPP model LNM+, and on the real-world datasets LastFM, MOOC, Retweets, Reddit, and Stack Overflow. Additional results for the other neural MTPP models, as well as results on the smaller and synthetic Hawkes datasets of Section 5.5.1, are provided in Section E.1 with similar conclusions.

5.5.5.1 Prediction Regions for the Arrival Time

In Figure 5.4, the results are systematically organized in a table where each row represents a specific metric, as detailed in Section 5.5.4, and each column corresponds to one of the datasets. This figure gives the results for various methods, described in Section 5.5.2, that are used to generate prediction regions solely for the arrival time. Each heuristic method and its corresponding conformal counterpart are represented in matching colors. To differentiate them, the heuristic methods are marked with hatching patterns.

The first row of the figure demonstrates that all CP methods attain the desired marginal coverage. In contrast, heuristic methods generally undercover, which aligns with expectations. The second row focuses on the average length of the prediction regions. Here, it is evident that heuristic methods generate smaller regions compared to their conformal counterparts. While this might seem beneficial, it is important to note that these smaller regions result from undercoverage, which diminishes their practical utility.

Among the heuristic methods, H-HDR consistently produces regions of smaller or equal lengths compared to H-QR and H-QRL for each prediction instance. Consequently, H-HDR emerges as the method with the smallest average region length. H-QR, not adjusting adequately to the right-skewed nature of the distributions, tends to yield larger regions.

Focusing now on the conformal methods, we exclude heuristic methods from this analysis due to their inability to achieve marginal coverage, which can lead to arbitrarily small regions. In the second row, the variations in average region length among CP methods differ across datasets. Notably, C-HDR, unlike its heuristic counterpart H-HDR, often yields larger average region lengths, especially in the LastFM, MOOC, and Retweets datasets. This difference arises because C-HDR adjusts the initial H-HDR prediction regions adaptively based on the individual predictive distributions. In contrast, C-QR and C-QRL modify their respective heuristic initial regions by a constant amount. While C-Const generates identical regions regardless of the history $\boldsymbol{h}$, it occasionally has the smallest average region length while still maintaining marginal coverage. This occurs because C-Const does not tailor its regions to account for extreme right-skewed distributions, leading to regions that are either slightly larger or significantly smaller compared to other conformal methods. These two phenomena are exemplified in a toy example shown in Fig. 5.5.

This figure demonstrates a scenario where the average region length of C-HDR is larger than that of other conformal methods in inter-arrival time prediction.

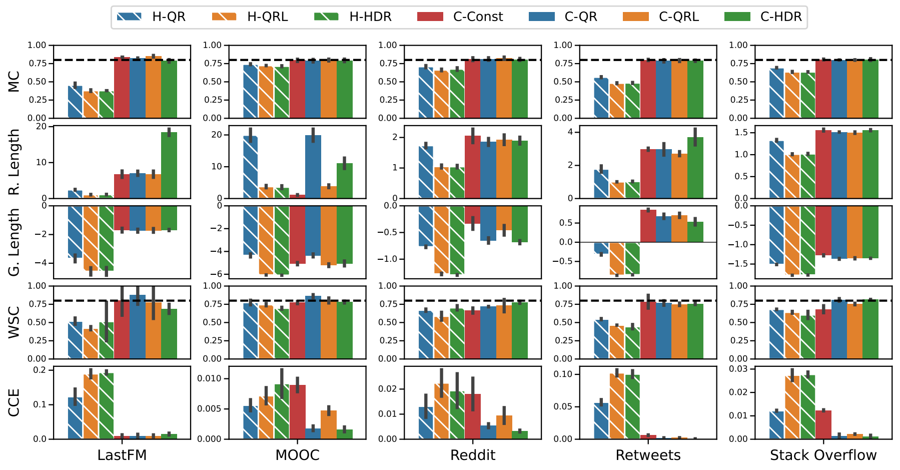


Figure 5.4: Performance of different methods producing a region for the time on real world datasets using the LNM+ model. Heuristic methods are hatched. Ideally, the MC and WSC metrics should align with the nominal coverage $1 - \alpha = 0.8$, while the relative length (R. Length), geometric length (G. Length), and CCE metrics should be minimized.

The first row shows predictive distributions in blue and their corresponding realizations as dashed lines, based on three observations from a calibration dataset. In the second row, the prediction regions for seven methods are depicted with $\alpha = 0.5$. All heuristic methods underperform, achieving a maximum coverage of only 1/3, which is less than the desired coverage of 0.5. Conformal prediction methods, in response, adjust their prediction regions to achieve coverage in at least two out of three cases. Despite H-HDR always producing shorter or equivalent lengths compared to H-QR and H-QRL, C-HDR generates larger regions on average than other conformal methods. Again, C-Const, which does not adapt to individual predictive distributions, presents the smallest average regions among the conformal methods in this particular example. C-Const however does not achieve conditional coverage even asymptotically.

Returning to Figure 5.4, the third row introduces an alternative aggregation method for region lengths – the geometric mean. This method assigns less weight to larger regions and more to smaller ones. Here, C-HDR's performance is more in line with other conformal methods, indicating that average region length might not be a reliable metric, particularly in cases of high variability in conditional distributions.

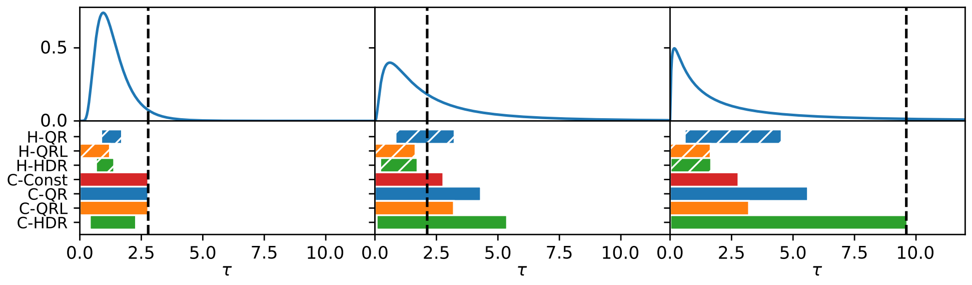


Figure 5.5: The figure showcases predictive distributions (blue) and realizations (dashed lines) in the first row, based on a calibration dataset. The second row illustrates prediction regions for various methods with $\alpha = 0.5$. It highlights the undercoverage of heuristic methods, the adaptive adjustments of conformal methods, and the notable differences between C-HDR and other methods in terms of region size. We provide an additional example with $\alpha = 0.2$ in Section E.4.

The fourth and fifth rows of the figure assess conditional coverage. WSC denotes coverage over the worst slab, with methods closer to $1 - \alpha$ being preferable, whereas CCE represents a conditional coverage error, which should be minimized. Conformal methods, already proficient in achieving marginal coverage, exhibit a conditional coverage that is usually better than heuristic methods based on the evaluated metrics. Methods capable of tailoring prediction regions to specific instances are expected to exhibit enhanced conditional coverage. Although the WSC metric reveals no marked distinction among conformal methods, the CCE metric shows that C-HDR frequently attains one of the highest levels of conditional coverage. Moreover, C-QR often outperforms C-QRL in conditional coverage. As anticipated, the CCE metric reveals that C-Const generally exhibits the poorest conditional coverage, attributable to its lack of adaptability.

5.5.5.2 Prediction Regions for the Mark

Figure 5.6 presents similar metrics as in Figure 5.4, but focuses on methods that generate prediction sets exclusively for the mark. Here, the heuristic methods H-APS and H-RAPS already meet the marginal coverage criteria, meaning that conformal prediction primarily offers theoretical backing rather than significant changes in predictions.

Turning our attention to the conformal methods, these methods show similar

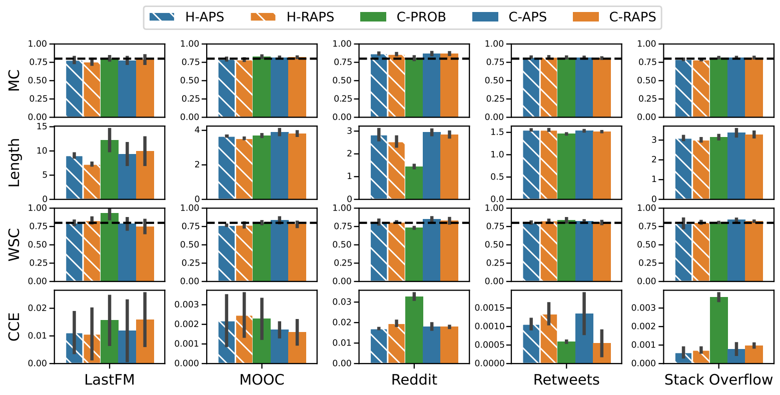


Figure 5.6: Performance of different methods producing a region for the mark on real world datasets using the LNM+ model. Heuristic methods are hatched. Ideally, the MC and WSC metrics should align with the nominal coverage $1 - \alpha = 0.8$, while the length and CCE metrics should be minimized.

region lengths across all datasets, with the exception of Reddit, where C-PROB exhibits smaller region lengths. However, on this same dataset and on Stack Overflow, C-PROB has a poor conditional coverage compared to both other conformal methods and heuristic methods. This reflects similar findings discussed in Section 5.5.5.1, where the method C-Const manages to attain short prediction regions, albeit with weak conditional coverage. This is explained by the fact that, in contrast to C-APS, C-PROB does not achieve conditional coverage asymptotically.

5.5.5.3 Joint Prediction Regions for the Arrival Time and the Mark

Figure 5.7 displays the same metrics as Figures 5.4 and 5.6, but it specifically focuses on methods that generate bivariate prediction sets for both the arrival time and the mark. Recall that we consider two main approaches. The first combines individual prediction regions, as detailed in Section 5.4.1. Under this approach, we examine two variants: QRL-RAPS, which merges QRL for time with RAPS for the mark, and HDR-RAPS, which combines HDR for time with RAPS for the mark. The second approach, outlined in Section 5.4.2, directly creates a bivariate prediction region. Conformal versions of these methods are also considered in our analysis.

In the first row, we see that conformal methods successfully achieve marginal coverage, whereas heuristic methods tend to undercover. This observation mir-

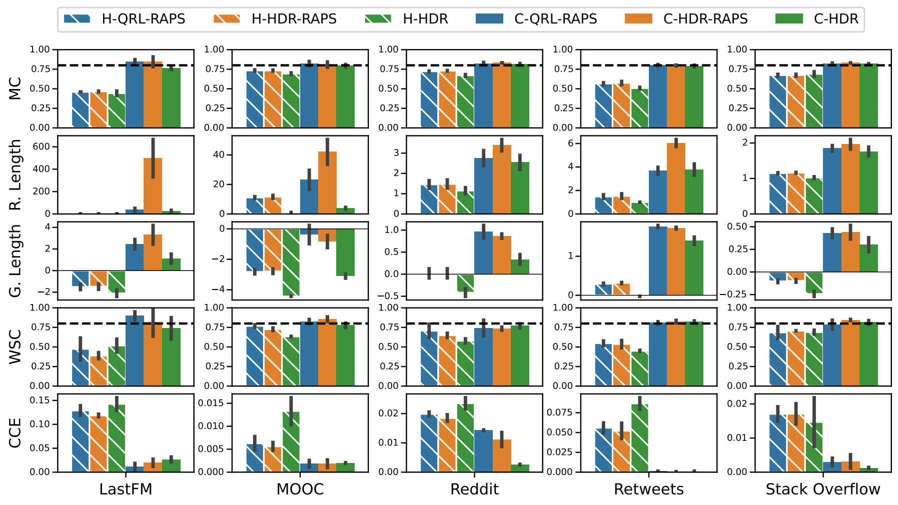


Figure 5.7: Performance of different methods producing a joint region for the time and mark on real world datasets using the LNM+ model. Heuristic methods are hatched. Ideally, the MC and WSC metrics should align with the nominal coverage $1 - \alpha = 0.8$, while the relative length (R. Length), geometric length (G. Length), and CCE metrics should be minimized.

rors the findings in the context of inter-arrival time prediction, as detailed in Section 5.5.5.1. The second row focuses on the average region length. Here, C-HDR-RAPS tends to produce larger regions on average compared to C-QRL-RAPS, consistent with the explanations provided in Section 5.5.5.1 and Fig. 5.5. Notably, C-HDR shows competitive average region lengths, outperforming C-HDR-RAPS. This highlights the benefit of creating joint regions that account for the interdependence between time and mark. In the third row, C-HDR stands out as the most effective on each dataset when using the geometric mean to average region lengths. The last two rows illustrate that conformal methods attain better conditional coverage than their heuristic counterparts, echoing the results observed in Section 5.5.5.1. C-HDR obtains a competitive conditional coverage, especially on the dataset Reddit.

For illustration, Figure 5.8 gives examples of naive bivariate prediction regions generated by C-QRL-RAPS and C-HDR-RAPS, as well as a bivariate highest density region using C-HDR. We can see that the naive approaches produce constant size intervals for each of the marks selected by the C-RAPS approach. Conversely, C-HDR is able to generate variable-length prediction intervals for each mark by taking the inter-dependencies between the two variables into account.

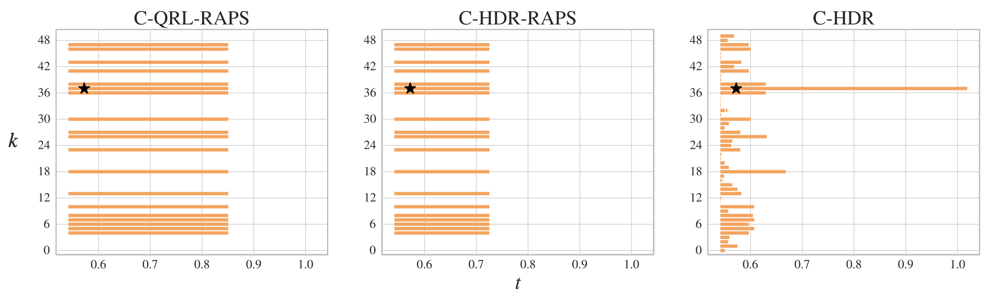


Figure 5.8: Examples of prediction regions generated by LNM+ using the C-QRL-RAPS, C-HDR-RAPS, and C-HDR methods for the last event of a test sequence of the LastFM dataset. The black star corresponds to the actual observed event.

5.5.5.4 Empirical Coverage for Different Coverage Levels

Fig. 5.9 shows the marginal coverage achieved by various methods generating joint prediction regions for both the arrival time and mark. This figure extends the analysis beyond the specific miscoverage level of $\alpha = 0.2$, as shown in the first row of Fig. 5.7, by including a range of coverage levels.

It is evident that heuristic methods generally underperform at all coverage levels, with this tendency becoming more pronounced at higher coverage levels. Conversely, conformal methods that construct individual predictions (outlined in Section Section 5.4.1) often overcover, particularly at lower coverage levels, due to their inherent conservativeness. Notably, C-HDR strikes an appropriate balance, maintaining the correct level of conservativeness across the various coverage levels. In Appendix E.3, we provide additional results for the coverage per level in the context of prediction regions for the time or for the mark.

5.6. Conclusion and Future Work

By integrating the methodologies of conformal prediction and neural MTPP models, we have established a more robust approach to uncertainty quantification in MTPPs. This is achieved by creating distribution-free joint prediction regions for the arrival time and its associated mark. The main challenge is to handle both a strictly positive, continuous response and a categorical response without distributional assumptions. We have also explored independently generating univariate prediction regions for the arrival time and the mark.

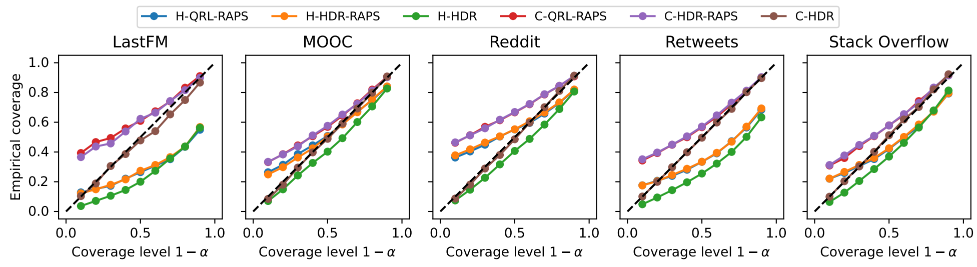


Figure 5.9: Empirical marginal coverage for different coverage levels with the LNM+ model. All conformal methods achieve marginal coverage but the naive method tends to overcover, especially for small coverage levels. The heuristic methods do not achieve coverage in most cases.

Our experiments highlight the importance of using conformal inference to ensure finite-sample marginal coverage. Indeed, heuristic methods tend to undercover in cases involving the prediction of arrival time or the simultaneous prediction of both arrival time and marks, with occasional success in predicting marks alone.

We also emphasize the significance of choosing appropriate conformal scores. C-HDR and C-QR show good conditional coverage, unlike C-Const, which lacks adaptability. The non-adaptive nature of C-Const leads to shorter average region lengths, which may appear advantageous at first glance. The same holds for C-PROB.

Our analysis underscores the importance of considering interdependence. Indeed, C-HDR, our extension of HPD-split (Izbicki et al., 2022) to bivariate outputs, outperforms C-HDR-RAPS. The superiority of C-HDR stems from its incorporation of joint regions that effectively consider and account for interdependence, whereas C-HDR-RAPS, in contrast, simplistically combines individual prediction regions through Bonferroni adjustments.

While this chapter focuses on the problem of MTPP, the techniques presented here, especially those involving C-HDR, have the potential to extend to other prediction problems where the target variable is a vector comprising a combination of continuous and categorical variables. To the best of our knowledge, this is the first time such prediction regions are explored in the context of conformal prediction.

CHAPTER 6

Conclusion

Sequences of marked events occurring in continuous time are prevalent across various application domains, ranging from social media activity and healthcare records to financial transactions and earthquake occurrences. Neural MTPP models leverage the flexibility of neural networks to model these event sequences, offering a structured approach to predict how systems will evolve over time. Since its introduction, the field of neural MTPP modeling has experienced rapid advancements, resulting in the development of numerous neural architectures that have been successfully applied to a wide range of real-world problems.

Despite its success, the field of neural MTPP modeling still faces some limitations, including inconsistent experimental setups across studies, optimization challenges that may impede model predictive performance, and unreliable prediction regions that cannot accurately capture the true uncertainty in the model predictions. In this thesis, we proposed to address these concerns by making three main contributions. First, we introduce novel neural MTPP models for probabilistic predictive modeling of continuous-time event data. Second, we present new training strategies for neural MTPP models designed to improve their predictive accuracy on the time and mark prediction tasks. Third, we develop reliable and distribution-free conformal methods for quantifying the uncertainty in the predictions extracted from neural MTPP models.

In this concluding chapter, we summarize these key contributions, and suggest potential directions for future work in the rapidly evolving field of neural MTPP modeling.

6.1. Summary of Contributions

A neural MTPP model is a structured combination of different architectural components, each tailored to capture different aspects of event sequences. To improve over existing baselines, new models generally propose alternatives to all these components simultaneously, making it challenging to identify the real sources of gains in predictive accuracy. More importantly, these new models often rely on different experimental setups, datasets and baselines for evaluation, which renders a fair comparison even harder (Shchur et al., 2021b).

To address this limitation, we conducted in Chapter 3 a large-scale experimental study of state-of-the-art neural MTPP models including 16 real-world and synthetic event sequence datasets. Specifically, our study reviewed in a unified experimental setup the influence of each main architectural component on predictive accuracy, and highlighted that some specific combinations of components can lead to significant improvements for both time and mark prediction tasks. In particular, our results revealed that self-attentive encoders required vectorial representations of time to demonstrate competitive performance with their recurrent counterparts. Also, we found that log-transformations of (inter)-arrival times are generally beneficial to most decoders in achieving good performance on the time prediction task.

Another important concept introduced in Chapter 3 relates to the probabilistic calibration of neural MTPP models. While calibration has been extensively discussed in the forecasting literature as a desirable property that any component forecaster should possess (Gneiting et al., 2007; Gneiting & Resin, 2023; Dheur & Ben Taieb, 2023), it received little attention from the neural MTPP community. We filled this gap by assessing the probabilistic calibration of neural MTPP models, which shed light on an important limitation: while the time predictive distributions are generally well calibrated, most neural MTPP models often show poor calibration for the mark predictive distributions. Overall, our experiments revealed that neural MTPP models show suboptimal results on the mark prediction task, sometimes being outperformed by simpler classical baselines.

Finally, we studied the impact of history size on model performance, and highlighted that solely encoding a few of the last observed events often resulted in comparable performance to encoding the full available history. This observation raises questions about whether some commonly used benchmark datasets are truly appropriate for the evaluation of neural MTPP models, or if alternative encoder architectures are required to capture long-term dependencies in event sequences.

The limitations of modern neural MTPP models with respect to the mark prediction task outlined in Chapter 4 motivated an investigation into the underlying causes of the problem. For some parametrizations, unsatisfactory performance in mark predictive accuracy can be directly attributed to specific design choices. For instance, LNM (Shchur et al., 2020a) and RMTPP (Du et al., 2016) make the explicit assumption of conditional independence between arrival times and marks. In Chapter 4, we have seen that relaxing this assumption by parametrizing the conditional PMF of marks as explicitly dependent on time led to improved performance on the mark prediction task.

Nonetheless, most neural MTPP parametrizations already take into account inter-dependencies between arrival times and marks, which suggested that this limitation was more deeply rooted into model training. In Chapter 4, we showed that learning a neural MTPP model can be interpreted as a two-task learning problem. Specifically, one task is focused on learning a time predictive distribution, while the other concerns learning a mark predictive distribution. Typically, these two tasks are optimized jointly on a common set of shared parameters. We demonstrated that this parameter sharing can lead to the emergence of conflicting gradients during training, often resulting in degraded performance on each individual task.

To prevent the emergence of conflicts during optimization, we introduced novel parametrizations of neural MTPP models, enabling disjoint modeling and training of the time and mark prediction tasks. Inspired from the success of (Shi et al., 2023), our framework allows to prevent gradient conflicts from the root while maintaining the flexibility of the original parametrizations. Our experiments proved that training neural MTPP models within our framework effectively prevented the occurrence of conflicts during optimization. We showed that this in turn lead to improved predictive accuracy and calibration compared to the original model formulations, especially in the context of the mark prediction task.

However, improved predictive accuracy does not guarantee that the trained model is a good approximation of the underlying unknown process. Consequently, prediction regions extracted directly from the model may be unreliable, failing to faithfully reflect on the true uncertainty in the predictions. In Chapter 5, we addressed this limitation by leveraging the sound framework of Conformal Predictions (CP) (Vovk et al., 2005), enabling the construction of reliable prediction regions that satisfy finite-sample coverage guarantees even when the base model is unreliable.

We first explored a naive method that combined individual prediction regions for events inter-arrival times and marks through Bonferroni adjustments. Despite offering finite-sample coverage guarantees, our results showed that this simple approach is highly conservative, often leading to large and inflexible prediction regions. We then set our attention to developing a more effective strategy that would take inter-dependencies between arrival times and marks into consideration. We proposed an extension of the HPD-Split approach of Izbicki et al. (2022), called C-HDR, based on bivariate HDR. Specifically, C-HDR generates predictions regions directly from the joint predictive density of inter-arrival times and marks, thereby explicitly taking dependencies into account. By excluding unlikely combinations of the two variables, this method proved to generate tighter joint prediction regions for arrival times and marks compared to the naive approach, while maintaining the desired coverage level.

Finally, we relied on approximate notions of conditional coverage to evaluate the quality of the generated prediction regions. Notably, our results revealed that a key factor to achieve good conditional coverage stemmed from the adaptivity of an approach to its input, further highlighting the benefits of C-HDR compared to simplistic, non-adaptive baselines.

6.2. Limitations and Future Work

The contributions presented in this dissertation aimed to deepen our understanding of the factors that hinder predictive accuracy in modern neural MTPP models, while also being a step forward in the development of more robust strategies to quantify the uncertainty of their predictions. Despite progress being made in this direction, we acknowledge that our work faces some limitations that deserve further investigation. In this section, we discuss such limitations and delve into several promising research directions that warrant future exploration.

The large scale experimental study conducted in Chapter 3 includes 15 real-world event sequence datasets from the neural MTPP literature. Nonetheless, one could consider including additional relevant datasets in the study, such as Reddit (Kumar et al., 2019), Financial Transactions (Du et al., 2016), Amazon (Ni et al., 2019), or Euro Email (Mei et al., 2020). The inclusion of these datasets could increase the power of the statistical comparisons conducted in Section (3.4), ultimately leading to more pronounced statistical differences between models. Additionally, a direct extension of our work could consider using versions of these datasets that were not filtered to their 50 most represented marks, enabling to assess the predictive accuracy of neural MTPP models in high-mark settings.

On a related note, the datasets mentioned in this dissertation have been used as the default benchmarks to evaluate the predictive accuracy of neural MTPP models since the early stages of the field. Consequently, it is not clear whether empirical gains over existing baselines stem from real progress in the field, or whether it is merely a byproduct of model overfitting. Moreover, as discussed in Shchur (2022), these datasets are not always derived from critical real-world applications that motivate the deployment of neural MTPP models, such as earthquake modeling or preventive medicine. We support their argument and invite future research to design high-quality and application-driven datasets for the evaluation of neural MTPP models, mirroring the standard benchmarks available in neighboring machine learning fields, such as ImageNet (Deng et al., 2009) in computer vision, or General Language Understanding Evaluation (GLUE) (Wang et al., 2018) in NLP. The suspicion raised by Enguehard et al. (2020) regarding the potential inappropriateness of some commonly used dataset, which was confirmed in our experiments, further points in that direction. Nonetheless, this challenge should not be too difficult to address in future works. For instance, the Github dataset considered in the dissertation contains a relatively short number of sequences, and is now outdated given the continuous growth of the code hosting platform since 2013 (GitHub, 2024). Hence, querying a larger and up to date Github dataset could serve as an appropriate high quality benchmark for the evaluation of neural MTPP models.

This analysis could also benefit from the inclusion of additional neural MTPP baselines, such as Attentive Neural Hawkes (A-NH) (Yang et al., 2022), WeibMix (Marin et al., 2005; Lin et al., 2021) or STHP (Li et al., 2023). In this context, since simple logarithmic transformations of inter-arrival times have shown to benefit most MLP decoders, it may be worthwhile to explore a broader range of related transformations, such as the Box-Cox transformation (Box & Cox, 1964). Additionally, predictive performance of neural MTPP models with respect to the time and mark prediction tasks was mainly evaluated using NLL scores and calibration measures. While these metrics enabled us to assess the entire predictive distributions, an extension of our experimental study could include more directly interpretable metrics, such as accuracy in event mark prediction, or MAE in event time prediction. Note that these point prediction metrics were subsequently considered in Chapter 4.

In this dissertation, we exclusively focused on the notions of probabilistic calibration and top-label calibration to quantify the reliability of the time and mark predictive distributions. While these notions have been extensively discussed (Naeini et al., 2015; Guo et al., 2017; Kuleshov et al., 2018; Laves et al., 2019; Dheur & Ben Taieb, 2023) as properties that any competent forecaster should possess, stronger notions of calibration have emerged in the literature to assess predictive distributions. As discussed in Gneiting & Resin (2023), probabilistic calibration mainly involves unconditional aspects of predictive performance and is implicitly implied by conditional notions such as distribution calibration (Song et al., 2019; Kuleshov & Deshpande, 2022), conditional exceedance calibration (Mason et al., 2007), threshold calibration (Henzi et al., 2021), and auto-calibration (Tsyplakov, 2013) being the strongest notion. Similarly, in the context of classification tasks, we may also require the predictive PMF to be calibrated with respect to all predicted probabilities, and not only the one associated to the top label (Zadrozny & Elkan, 2002; Vaicenavicius et al., 2019). We believe that exploring the calibration of modern neural MTPP models within the context of these stronger aspects constitutes an important avenue of future research.

In Chapter 3, our experiments suggested that current RNN and self-attentive history encoders struggle at efficiently capturing long-term dependencies between event occurrences, motivating the design of alternative architectures. A promising advance in the neighboring field of sequence modeling relates to Deep Discrete-time State Space Models (DDSM) (Gu et al., 2022a,b; Gupta et al., 2022; Hasani et al., 2023). In a DDSSM, outputs are parametrized as the discretization of a continuous-time state space model that specifies the relation between an output signal and a latent space function of the input. Under careful initialization of the state matrices, DDSSM have demonstrated impressive performance on tasks from the Long Range Arena (LRA) (Tay et al., 2020), a benchmark that requires reasoning over long-context scenarios. Relatedly, (Orvieto et al., 2023) recently showed that the performance and efficiency of DDSSM over long-term reasoning tasks can be matched by classical RNNs after applying a series of small modifications to their architectures. Integrating these architectures into the design of novel neural MTPP models could facilitate the capture of long-term inter-dependencies between events, hereby improving downstream predictive performance.

From an applied perspective, an exciting application of MTPP models arise in the context of *intermittent demand forecasting* (Boylan & Syntetos, 2021), where demand occurrences materialize as sporadic and lumpy time series over time, i.e. containing few and highly variable observations. These sequences bear evident resemblance with the realization of a MTPP, where events occur with irregularly-spaced intervals. While intermittent demand forecasting has been extensively studied in the times series literature (Croston, 1972; Willemain et al., 2004; Kourentzes, 2013; Karthikeswaren et al., 2021), the problem received less attention from the MTPP community. A notable exception relates to the work of Türkmen et al. (2021), which draws a parallel between the two domains, and proposes an approach based on neural renewal point processes. However, the field of MTPP modeling has witnessed significant development in the meantime, warranting the exploration of how recent approaches could be adapted to this context.

In Chapter 4, we tackled the challenge of conflicting gradients by introducing novel parametrizations of neural MTPP models that enabled disjoint modeling and training of the time and mark prediction tasks. Within the multi-task learning literature, our methodology relates most to branched architecture search approaches (Guo et al., 2020; Shi et al., 2023), where shared network layers between tasks are separated when their joint training impedes performance. However, other approaches have been proposed to handle negative transfer in multi-task learning, with a prominent line of work being gradient surgery methods, where conflicts are mitigated by direct manipulation of the tasks' gradients. Examples of gradient surgery approaches include homogenizing gradients magnitude (Chen et al., 2018) or direction (Javaloy & Valera, 2022), using their projections onto the normal plane as update rule (Yu et al., 2020), or randomly dropping updates based on conflict (Chen et al., 2020). Although gradient surgery approaches have been criticized by (Shi et al., 2023) across a selection of computer vision tasks, their effectiveness in the context of neural MTPP models training remains unexplored.

Moreover, our analysis concerned neural MTPP models trained on the NLL objective. As mentioned in Section 2.1.3, alternatives to the LogScore that are specifically tailored to the predictive distributions of inter-arrival times and marks can be considered for model training. For instance, the CRPS can be employed to evaluate the predictive distribution of inter-arrival times as done in Ben Taieb (2022), while the Brier score and the spherical score can be employed to evaluate either predictive distributions (Gneiting & Raftery, 2007). Investigating how task gradients interact in the context of alternative scoring rules, and whether their influence remains detrimental to the learning of each individual task, is an interesting direction of future work.

Our analysis conducted in Chapter 5 constitutes a first step towards the deployment of distribution-free conformal approaches in the context of neural MTPP models. However, given the inherent dependencies between event occurrences, our study focused on building prediction regions for the last event of a sequence. Consequently, extending our approach to generate predictions regions that take these dependencies into account constitutes an important direction of future research.

In the related field of time series forecasting, conformal methods can be classified into two main categories based on the setting on which they operate. Single-series approaches aim to generate prediction regions for new values based on past observations within a specific series, circumventing strict exchangeability assumptions by either dynamically adapting the coverage level (Gibbs &

Candès, 2021; Zaffran et al., 2022) or the quantile of the scores (Xu & Xie, 2021, 2023b). For multi-series, conformal approaches typically leverage the exchangeability of series to generate prediction regions that attain simultaneous coverage over entirely new trajectories (Stankeviciute et al., 2021; Yu et al., 2023; Cleaveland et al., 2024; Zhou et al., 2024). A direct parallel between these settings and the one of neural MTPP can be drawn, which warrants further exploration on the applicability of these approaches to our problem.

A more simplistic approach to handling in-sequence dependencies could consist in splitting the sequences into blocks of observations, similar to (Chernozhukov et al., 2018). Under the assumption of weak dependence between blocks of observations, the regions generated by split conformal prediction remain approximately valid (Oliveira et al., 2024), and significant deviations from exchangeability can be assessed using statistical tests, such as Saha & Ramdas (2023). Moreover, conformal prediction methods could be employed for tasks that extend beyond our definitions of the time and mark prediction tasks. For instance, in long-horizon forecasting applications of TPP models (Deshpande et al., 2021; Xue et al., 2022; Ludke et al., 2024), a common task is to predict the number of events that will occur in a specified observation window. In this context, exploring conformal approaches to construct reliable prediction regions for event counts emerges as an promising extension of our framework.

In this work, we also assumed that all sequences were drawn exchangeably from the same ground-truth process, which may not be systematically valid in practice. Suppose instead a system governed by multiple sub-processes, and that each sub-process generates sequences exchangeably. Although sequences generated by the same sub-process are exchangeable, this hypothesis is no longer verified for sequences emanating from different sub-processes. Hence, our methodology cannot be applied directly in this scenario. Fortunately, recent developments in conformal prediction for hierarchical data (Dunn et al., 2023; Lee et al., 2024) and federated learning (Plassier et al., 2023) provide tools that could be exploited to address the challenge of heterogeneous event sequences.

Finally, we would like to highlight future research directions that extend beyond the framework of MTPP models. Throughout this dissertation, we exclusively focused on MTPPs with categorical marks. Discrete marks offer a practical representation for processes where events can be classified into discrete categories, such the category of an item bought at a specific time. However, in other applications, events may instead materialize with continuous marks representing, e.g. their spatial coordinates. Such processes are called *Spatio-Temporal Point Processes (STPP)* and several works recently proposed flexible neural STPP models to capture complex spatio-temporal dependencies between event occurrences (Okawa et al., 2019; Chen et al., 2021b; Zhou et al., 2022; Zhou & Yu, 2023; Zhang et al., 2023). However, marked STPP, where events are represented by both a discrete and a continuous mark, have so far received little attention from the neural TPP community. Notable exceptions are exemplified by the works of Yeung et al. (2023) and Narayanan et al. (2023).

Another active field of research relates to *structured temporal point processes*, where the aim is either to learn a TPP model that governs the temporal evolution of nodes and edges in dynamic interaction graphs (Trivedi et al., 2017, 2019; Gracious & Dukkipati, 2023), or to learn a latent dynamic graph that governs the evolution of events' interactions over time (Zhang et al., 2021; Yang & Zha, 2024). Investigating how our parametrization framework can be extended to the contexts of marked STPP and structured TPP models, along with the development of approaches for distribution-free conformal inference in these contexts, opens the door to promising avenues of future research.

Bibliography

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., & Nahavandi, S. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, *76*, 243–297.

Abramowitz, M., & Stegun, I. A. (1965). Handbook of mathematical functions: With formulas, graphs, and mathematical tables. *Courier Corporation*.

Altieri, L., Farcomeni, A., & Fegatelli, D. A. (2023). Continuous time-interaction processes for population size estimation, with an application to drug dealing in italy. *Biometrics*, *79*(2), 1254–1267.

Anderson, T. W., & Darling, D. A. (1952). Asymptotic theory of certain "goodness of fit" criteria based on stochastic processes. *Annals of Mathematical Statistics*, *23*, 193–212.

Angelopoulos, A., Bates, S., Malik, J., & Jordan, M. I. (2021). Uncertainty sets for image classifiers using conformal prediction. *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*.

Angelopoulos, A. N., & Bates, S. (2023). Conformal prediction: A gentle introduction. *Foundations and Trends® in Machine Learning*, *16*(4), 494–591.

Bacry, E., Bompaire, M., Deegan, P., Gaïffas, S., & Poulsen, S. V. (2018). tick: a python library for statistical learning, with an emphasis on Hawkes processes and time-dependent models. *Journal of Machine Learning Research*, *18*(214), 1–5.

Bacry, E., Bompaire, M., Gaïffas, S., & Muzy, J.-F. (2020). Sparse and low-rank multivariate Hawkes processes. *Journal of Machine Learning Research (JMLR)*, *21*(50), 1–32.

Bacry, E., Mastromatteo, I., & Muzy, J.-F. (2015). Hawkes processes in finance. *ArXiv preprint*.

Bacry, E., & Muzy, J. F. (2014). Hawkes model for price and trades high-frequency dynamics. *Quantitative Finance*, *14*(7), 1147–1166.

Barlow, R. E., & Proschan, F. (1965). Mathematical theory of reliability. *Wiley*.

Bateman, H. (1910). The transformation of the electrodynamical equations. *Proceedings of the London Mathematical Society*, *8*, 223–264.

Ben Taieb, S. (2022). Learning quantile functions for temporal point processes with recurrent neural splines. *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Bhave, S., & Perotte, A. (2021). Point processes for competing observations with recurrent networks (popcorn): A generative model of ehr data.

Biloš, M., Charpentier, B., & Günnemann, S. (2019). Uncertainty on asynchronous time event prediction. *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*.

Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. (2015). Weight uncertainty in neural networks. *Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML)*.

Bosser, T., & Ben Taieb, S. (2023a). On the predictive accuracy of neural temporal point process models for continuous-time event data. *Transactions of Machine Learning Research (TMLR)*.

Bosser, T., & Ben Taieb, S. (2023b). Revisiting the mark conditional independence assumption in neural marked temporal point processes. *Proceedings of the 31st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*.

Box, G. E. P., & Cox, D. R. (1964). An analysis of transformations. *Journal of the Royal Statistical Society: Series B (Methodological)*, *6*, 211–243.

Boyd, A., Bamler, R., Mandt, S., & Smyth, P. (2020). User-dependent neural sequence models for continuous-time event data. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS).*

Boyd, A., Chang, Y., Mandt, S., & Smyth, P. (2023). Inference for mark-censored temporal point processes. *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI).*

Boylan, J. E., & Syntetos, A. A. (2021). Intermittent demand forecasting: Context, methods and applications. *Wiley*.

Bragman, F. J. S., Tanno, R., Ourselin, S., Alexander, D. C., & Cardoso, M. J. (2019). Stochastic filter groups for multi-task cnns: Learning specialist and generalist convolution kernels. *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV).*

Brehmer, J., Gneiting, T., Herrmann, M., Marzocchi, W., Schlather, M., & Strokorb, K. (2021). Using scoring functions to evaluate point process forecasts. *ArXiv preprint*.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, *78*(1), 1–3.

Brown, E. N., Barbieri, R., Ventura, V., Kass, R. E., & Frank, L. M. (2002). The time-rescaling theorem and its application to neural spike train data analysis. *Neural Computation*, *14*(2), 325–346.

Bruggemann, D., Kanakis, M., Georgoulis, S., & Gool, L. V. (2020). Automated search for resource-efficient branched multi-task networks. *Proceedings of the 31st British Machine Vision Conference (BMVC).*

Campos, M. M., Farinhas, A., Zerva, C., Figueiredo, M. A. T., & Martins, A. F. T. (2024). Conformal prediction for natural language processing: A survey. *ArXiv preprint*.

Candès, E. J., Lei, L., & Ren, Z. (2023). Conformalized survival analysis. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *85*(1), 24–45.

Cauchois, M., Gupta, S., & Duchi, J. (2020). Knowing what you know: valid and validated confidence sets in multiclass and multilabel prediction. Journal of Machine Learning Research (JMLR).

Cauchois, M., Gupta, S., & Duchi, J. C. (2021). Knowing what you know: valid and validated confidence sets in multiclass and multilabel prediction. *Journal of machine learning research (JMLR)*, *22*(1), 3681–3722.

Chen, R. T. Q., Amos, B., & Nickel, M. (2021a). Learning neural event functions for ordinary differential equations. *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*.

Chen, R. T. Q., Amos, B., & Nickel, M. (2021b). Neural spatio-temporal point processes. *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*.

Chen, S., Shojaie, A., Shea-Brown, E., & Witten, D. (2019). The multivariate Hawkes process in high dimensions: Beyond mutual excitation. *ArXiv preprint*.

Chen, Y. (2018). Thinning algorithms for simulating point processes. *Lecture notes, Florida State University.*.

Chen, Z., Badrinarayanan, V., Lee, C.-Y., & Rabinovich, A. (2018). Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. *Proceedings of the 35th International Conference on Machine Learning (ICML)*.

Chen, Z., Ngiam, J., Huang, Y., Luong, T., Kretzschmar, H., Chai, Y., & Anguelov, D. (2020). Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*.

Chernozhukov, V., Wuthrich, K., & Yinchu, Z. (2018). Exact and robust conformal inference methods for predictive machine learning with dependent data. *Proceedings of the 31st Conference On Learning Theory*.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Choi, E., Du, N., Chen, R., Song, L., & Sun, J. (2015). Constructing disease network and temporal progression model via context-sensitive Hawkes process. *Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM)*.

Clark, T. G., Bradburn, M. J., Love, S. B., & Altman, D. G. (2003). Survival analysis part i: Basic concepts and first analyses. *British Journal of Cancer*, *89*, 232–238.

Cleaveland, M., Lee, I., Pappas, G. J., & Lindemann, L. (2024). Conformal prediction regions for time series using linear complementarity programming. *Proceedings of the 38th AAAI Conference on Artificial Intelligence (AAAI)*.

Costa, M., Graham, C., Marsalle, L., & Tran, V. C. (2020). Renewal in Hawkes processes with self-excitation and inhibition. *Advances in Applied Probability*, *52*(3), 879–915.

Cox, D. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, *34*(2), 187–202.

Cox, D. R. (1955). Some statistical methods connected with series of events. *Journal of the Royal Statistical Society: Series B (Methodological)*, *17*(2), 129–157.

Cox, D. R. (1962). Renewal theory. *Methuen*.

Cox, D. R. (1966). The statistical analysis of series of events. *Springer*.

Cramér, H. (1928). On the composition of elementary errors. *Scandinavian Actuarial Journal*, *1928*, 13–74.

Craswell, N. (2009). Mean reciprocal rank. *Encyclopedia of Database Systems, Springer*.

Croston, J. (1972). Forecasting and stock control for intermittent demands. *Operational Research Quarterly*, *23*, 289–303.

Daley, D., & Vere-Jones, D. (2003). An introduction to the theory of point processes. volume i: Elementary theory and methods. *Springer*.

Daley, D. J., & Vere-Jones, D. (2008). An introduction to the theory of point processes. volume ii: General theory and structure. *Springer*.

Dawid, A. P. (1984). Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society. Series A*, *147*(2), 278–292.

De, A., Upadhyay, U., & Gomez-Rodriguez, M. (2019). Temporal point processes. *Notes for Human-Centered ML, Saarland University*.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research (JMLR)*, *7*(1), 1–30.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Deshpande, P., Marathe, K., De, A., & Sarawagi, S. (2021). Long horizon forecasting with temporal point processes. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*.

Dheur, V., & Ben Taieb, S. (2023). A large-scale study of probabilistic calibration in neural network regression. *Proceedings of the 40th International Conference on Machine Learning (ICML)*.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., & Song, L. (2016). Recurrent marked temporal point processes: Embedding event history to vector. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

Du, N., Wang, Y., He, N., Sun, J., & Song, L. (2015). Time-sensitive recommendation from recurrent user activities. *Proceedings of the 28th International Conference on Neural Information Processing Systems (NeurIPS)*.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, *12*(61), 2121–2159.

Dunn, R., Wasserman, L., & Ramdas, A. (2023). Distribution-Free prediction sets for Two-Layer hierarchical models. *Journal of the American Statistical Association*, *118*(544), 2491–2502.

Duval, C., Luçon, E., & Pouzat, C. (2022). Interacting Hawkes processes with multiplicative inhibition. *Stochastic Processes and their Applications*, *148*, 180–226.

Egesdal, M., Fathauer, C., Louie, K., Neuman, J., Mohler, G., & Lewis, E. (2010). Statistical and stochastic modeling of gang rivalries in Los Angeles. *SIAM Undergraduate Research Online*.

Eichler, M., Dahlhaus, R., & Dueck, J. (2017). Graphical modeling for multivariate hawkes processes with nonparametric link functions. *Journal of Time Series Analysis*, *38*(2), 225–242.

Embrechts, P., Liniger, T., & Lin, L. (2011). Multivariate Hawkes processes: an application to financial data. *Journal of Applied Probability*, *48*(A), 367–378.

Enguehard, J., Busbridge, D., Bozson, A., Woodcock, C., & Hammerla, N. Y. (2020). Neural temporal point processes for modelling electronic health records. *Proceedings of Machine Learning Research (PMLR)*, *136*, 85–113.

Erlang, A. K. (1909). The theory of probabilities and telephone conversations. *Nyt Tidsskrift for Matematik B*, *20*, 33.

Fatt, P., & Katz, B. (1965). Spontaneous subthreshold activity at motor nerve endings. *The Journal of Physiology*, *117*(1), 109–128.

Feldman, S., Bates, S., & Romano, Y. (2023). Calibrated multiple-output quantile regression with representation learning. *Journal of machine learning research (JMLR)*, *24*(24), 1–48.

Feller, W. (1949). On the theory of stochastic processes, with particular reference to applications. *Proceedings of the 1949 Berkeley Symposium on Mathematical Statistics and Probability*.

Fifty, C., Amid, E., Zhao, Z., Yu, T., Anil, R., & Finn, C. (2021). Efficiently identifying task groupings for multi-task learning. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Foygel Barber, R., Candès, E. J., Ramdas, A., & Tibshirani, R. J. (2021). The limits of distribution-free conditional predictive inference. *Information and Inference: A Journal of the IMA*, *10*(2), 455–482.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, *32*(200), 675–701.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *Annals of Mathematical Statistics*, *11*(1), 86–92.

Fukushima, K. (1969). Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, *5*(4), 322–333.

Gal, Y., & Ghahramani, Z. (2016). Dropout as a bayesian approximation:

Representing model uncertainty in deep learning. *Proceedings of the 33rd International Conference on Machine Learning (ICML).*

Gao, Y., Bai, H., Jie, Z., Ma, J., Jia, K., & Liu, W. (2020). MTL-NAS: Task-agnostic neural architecture search towards general-purpose multi-task learning. *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Gao, Y., Ma, J., Zhao, M., Liu, W., & Yuille, A. L. (2019). NDDR-CNN: Layerwise feature fusing in multi-task CNNs by neural discriminative dimensionality reduction. *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

García, S., & Herrera, F. (2008). An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research (JMLR)*, *9*(89), 2677–2694.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A., Triebel, R., Jung, P., Roscher, R., Shahzad, M., Yang, W., Bamler, R., & Zhu, X. X. (2023). A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, *56*, 1513–1589.

Gebetsbergera, M., Messner, J. W., Mayr, G. J., & Zeileis, A. (2018). Estimation methods for nonhomogeneous regression models: Minimum continuous ranked probability score versus maximum likelihood. *Monthly Weather Review*, *146*(12), 4323–4338.

Ghanem, R., Higdon, D., & Owhadi, H. (2017). Handbook of uncertainty quantification. *Springer*.

Gibbs, I., & Candès, E. (2021). Adaptive conformal inference under distribution shift. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS).*

GitHub (2024). The 2024 state of the octoverse - community report. *Available at: https://octoverse.github.com/.*

Gneiting, T. (2012). Making and evaluating point forecasts. *Journal of the American Statistical Association*, *106*, 746–762.

Gneiting, T., Balabdaoui, F., & E. Raftery, A. (2007). Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *69*(2), 243–268.

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, *102*(477), 359–378.

Gneiting, T., & Resin, J. (2023). Regression diagnostics meets forecast evaluation: Conditional calibration, reliability diagrams, and coefficient of determination. *Electronic Journal of Statistics*, *17*(2), 3226–3286.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning. *MIT Press*, *1*.

Gracious, T., & Dukkipati, A. (2023). Dynamic representation learning with temporal point processes for higher-order interaction forecasting. *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI)*.

Gruber, C., Schenk, P. O., Schierholz, M., Kreuter, F., & Kauermann, G. (2023). Sources of uncertainty in machine learning – a statisticians' view. *ArXiv preprint*.

Gu, A., Goel, K., & Ré, C. (2022a). Efficiently modeling long sequences with structured state spaces. *Proceedings of the 2022 International Conference on Learning Representation (ICLR)*.

Gu, A., Gupta, A., Goel, K., & Ré, C. (2022b). On the parameterization and initialization of diagonal state space models. *Proceedings of the 36th International Conference on Neural Information Processing Systems (NeurIPS)*.

Gui, Y., Hore, R., Ren, Z., & Barber, R. F. (2023). Conformalized survival analysis with adaptive cutoffs. *Biometrika*, *11*(2), 459–477.

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

Guo, F., Blundell, C., Wallach, H., & Heller, K. (2015). The bayesian echo chamber: Modeling social influence via linguistic accommodation. *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Guo, M., Haque, A., Huang, D.-A., Yeung, S., & Fei-Fei, L. (2018a). Dynamic task prioritization for multitask learning. *Proceedings of the 2018 European Conference on Computer Vision (ECCV)*.

Guo, P., Lee, C.-Y., & Ulbricht, D. (2020). Learning to branch for multi-task learning. *Proceedings of the 37th International Conference on Machine Learning (ICML).*

Guo, R., Li, J., & Liu, H. (2018b). Initiator: Noise-contrastive estimation for marked temporal point process. *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI).*

Gupta, A., Gu, A., & Berant, J. (2022). Diagonal state spaces are as effective as structured state spaces. *Proceedings of the 36th International Conference on Neural Information Processing Systems (NeurIPS).*

Gupta, V., Bedathur, S. J., Bhattacharya, S., & De, A. (2021). Learning temporal point processes with intermittent observations. *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS).*

Ha, D., Dai, A., & Le, Q. V. (2016). Hypernetworks. *ArXiv preprint.*

Hasani, R., Lechner, M., Wang, T.-H., Chahine, M., Amini, A., & Rus, D. (2023). Liquid structural state-space models. *Proceedings of the 2023 International Conference on Learning Representations (ICLR).*

Hawkes, A. G. (1971). Point spectra of some mutually exciting point processes. *Journal of the Royal Statistical Society: Series B (Methodological)*, *33*(3), 438–443.

Hawkes, A. G. (2018). Hawkes processes and their applications to finance: A review. *Quantitative Finance*, *18*(2), 193–198.

Hendrycks, D., & Gimpel, K. (2023). Gaussian error linear units (GELUs). *ArXiv preprint.*

Henzi, A., Ziegel, J. F., & Gneiting, T. (2021). Isotonic distributional regression. *Journal of the Royal Statistical Society: Series B (Methodological)*, *83*(5), 963–993.

Hidasi, B., & Tikk, D. (2012). Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback. *Proceedings of the 2012 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD).*

Hinton, G. (2018). Neural networks for machine learning. *Coursera, lecture 6a.*

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, *6*(2), 65–70.

Hyndman, R. J. (1996). Computing and graphing highest density regions. *The American Statistician*, *50*(2), 120–126.

Hüllermeier, E., & Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, *110*(3), 457–506.

Isham, V., & Westcott, M. (1979). A self-correcting point process. *Stochastic Processes and Their Applications*, *8*(3), 335–347.

Izbicki, R., Shimizu, G., & Stern, R. B. (2022). CD-split and HPD-split: Efficient conformal regions in high dimensions. *Journal of machine learning research (JMLR)*, *23*(87), 1–32.

Javaloy, A., & Valera, I. (2022). Rotograd: Gradient homogenization in multitask learning. *Proceedings of the 2022 International Conference on Learning Representations (ICLR).*

Kalbfleisch, J. D., & Prentice, R. L. (2002). The statistical analysis of failure time data. *John Wiley & Sons*.

Kalos, M. H., & Whitlock, P. A. (2008). Monte carlo methods. *John Wiley & Sons*.

Karthikeswaren, R., Kayathwal, K., Dhama, G., & Arora, A. (2021). A survey on classical and deep learning based intermittent time series forecasting methods. *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN).*

Kato, Y., Tax, D. M., & Loog, M. (2023). A review of nonconformity measures for conformal prediction in regression. *Proceedings of the 12th Symposium on Conformal and Probabilistic Prediction with Applications*.

Kazemi, S. M., Goel, R., Eghbali, S., Ramanan, J., Sahota, J., Thakur, S., Wu, S., Smyth, C., Poupart, P., & Brubaker, M. (2019). Time2vec: Learning a vector representation of time. *ArXiv preprint*.

Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *Proceedings of the 2014 International Conference on Learning Representations (ICLR).*

Kingman, J. F. (1992). Poisson processes. *Clarendon Press.*

Kleiber, C., & Kotz, S. (2003). Statistical size distributions in economics and actuarial sciences. *Wiley.*

Klein, J. P., van Houwelingen, H. C., Ibrahim, J. G., & Scheike, T. H. (2020). Handbook of survival analysis. *Chapman & Hall.*

Kokkinos, I. (2017). UberNet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. *Proceedings of the 2017 IEEE International Conference on Computer Vision and Pattern Recognition (CVPR).*

Kourentzes, N. (2013). Intermittent demand forecasts with neural networks. *International Journal of Production Economics*, *143*(1), 198–206.

Kuleshov, V., & Deshpande, S. (2022). Calibrated and sharp uncertainties in deep learning via density estimation. *Proceedings of the 39th International Conference on Machine Learning (ICML).*

Kuleshov, V., Fenner, N., & Ermon, S. (2018). Accurate uncertainties for deep learning using calibrated regression. *Proceedings of the 35th International Conference on Machine Learning (ICML).*

Kumar, S., Zhang, X., & Leskovec, J. (2019). Predicting dynamic embedding trajectory in temporal interaction networks. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).*

Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS).*

Laub, P. J., Taimre, T., & Pollett, P. K. (2015). Hawkes processes. *ArXiv preprint*.

Laves, M.-H., Ihler, S., Kortmann, K.-P., & Ortmaier, T. (2019). Well-calibrated model uncertainty with temperature scaling for dropout variational inference. *Proceedings of the 4th Workshop on Bayesian Deep Learning (NeurIPS)*.

Lee, Y., Barber, R. F., & Willett, R. (2024). Distribution-free inference with hierarchical data. *ArXiv preprint*.

Lee, Y., Lim, K. W., & Ong, C. S. (2016). Hawkes processes with stochastic excitations. *Proceedings of The 33rd International Conference on Machine Learning (ICML)*.

Lei, J., G'Sell, M., Rinaldo, A., & Tibshirani, R. J. (2018). Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, *113*(523), 1094–1111.

Lei, J., Robins, J., & Wasserman, L. (2013). Distribution-free prediction sets. *Journal of the American Statistical Association*, *108*(501), 278–287.

Lewis, P. (1972). Stochastic point processes: Statistical analysis, theory and applications. *Wiley-Interscience*.

Lewis, P. A. W., & Shedler, G. S. (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, *26*(3), 403–413.

Li, S., Xiao, S., Zhu, S., Du, N., Xie, Y., & Song, L. (2018). Learning temporal point processes via reinforcement learning. *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*.

Li, Z., Xu, Y., Zuo, S., Jiang, H., Zhang, C., Zhao, T., & Zha, H. (2023). SMURF-THP: Score matching-based uncertainty quantification for transformer Hawkes process. *Proceedings of the 40th International Conference on Machine Learning (ICML)*.

Lin, H., Tan, C., Wu, L., Gao, Z., & Li, S. Z. (2021). An empirical study: Extensive deep temporal point process. *ArXiv preprint*.

Lin, H., Wu, L., Zhao, G., Liu, P., & Li, S. Z. (2022). Exploring generative neural temporal point process. *Transactions on Machine Learning Research (TMLR)*.

Liniger, T. J. (2009). Multivariate Hawkes processes. *PhD Thesis, ETH Zurich*.

Liu, B., Liu, X., Jin, X., Stone, P., & Liu, Q. (2021a). Conflict-averse gradient descent for multi-task learning. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Liu, B., Liu, X., Jin, X., Stone, P., & Liu, Q. (2021b). Conflict-averse gradient descent for multi-task learning. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Liu, L., Li, Y., Kuang, Z., Xue, J.-H., Chen, Y., Yang, W., Liao, Q., & Zhang, W. (2021c). Towards impartial multi-task learning. *Proceedings of the 2021 International Conference on Learning Representations (ICLR)*.

Liu, S., Johns, E., & Davison, A. J. (2019). End-to-end multi-task learning with attention. *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Long, M., Cao, Z., Wang, J., & Yu, P. S. (2017). Learning multiple tasks with multilinear relationship networks. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*.

Ludke, D., Bilos, M., Shchur, O., Lienen, M., & Günnemann, S. (2024). Add and thin: Diffusion for temporal point processes. *Proceedings of the 37th International Conference on Neural Information Processing Systems (NeurIPS)*.

Lukasik, M., Srijith, P. K., Vu, D., Bontcheva, K., Zubiaga, A., & Cohn, T. (2016). Hawkes processes for continuous time sequence classification: an application to rumour stance classification in Twitter. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Maninis, K.-K., Radosavovic, I., & Kokkinos, I. (2019). Attentive single-tasking of multiple tasks. *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Manning, C. D., Raghavan, P., & Schutze, H. (2008). Introduction to information retrieval. *Cambridge University Press*.

Marin, J. M., Rodríguez-Bernal, M. T., & Wiper, M. P. (2005). Using Weibull mixture distributions to model heterogeneous survival data. *Communications in Statistics - Simulation and Computation*, *34*(3), 673–684.

Mason, S. J., Galpin, J. S., Goddard, L., Graham, N. E., , & Rajartnam, B. (2007). Conditional exceedance probabilities. *Monthly Weather Review*, *135*(2), 363–372.

Massey, F. J. (1951). The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, *46*, 68–78.

Mehrtash, A., Wells, W. M., Tempany, C. M., Abolmaesumi, P., & Kapur, T. (2020). Confidence calibration and predictive uncertainty estimation for deep medical image segmentation. *IEEE Transactions on Medical Imaging*, *39*(12), 3868–3878.

Mei, H., & Eisner, J. (2017). The neural Hawkes process: A neurally self-modulating multivariate point process. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS)*.

Mei, H., Qin, G., & Eisner, J. (2019). Imputing missing events in continuous-time event streams. *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

Mei, H., Wan, T., & Eisner, J. (2020). Noise-contrastive estimation for multivariate point processes. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*.

Mises, R. V. (1928). Wahrscheinlichkeit, statistik und wahrheit. *Julius Springer*.

Misra, I., Shrivastava, A., Gupta, A., & Hebert, M. (2016). Cross-stitch networks for multi-task learning. *Proceedings of the 2016 IEEE/CVF International Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mohler, G. O., Short, M. B., Brantingham, P. J., Schoenberg, F. P., & Tita, G. E. (2011). Self-exciting point process modeling of crime. *Journal of the American Statistical Association*, *106*(493), 100–108.

Morse, P. M. (1958). Queues, inventories, and maintenance : The analysis of operational system with variable demand and supply. *Wiley*.

Mortier, T., Bengs, V., Hüllermeier, E., Luca, S., & Waegeman, W. (2023). On the calibration of probabilistic classifier sets. *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Naeini, M. P., Cooper, G. F., & Hauskrecht, M. (2015). Obtaining well calibrated probabilities using bayesian binning. *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*.

Narayanan, S., Kosmidis, I., & Dellaportas, P. (2023). Flexible marked spatio-temporal point processes with applications to event sequences from association football. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, *72*(5), 1095–1126.

Ni, J., Li, J., & McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. *Proceedings of the 22nd International Conference on Machine Learning (ICML)*.

Oakes, D. (1975). The markovian self-exciting process. *Journal of Applied Probability*, *12*(1), 69–77.

Ogata, Y. (1981). On Lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, *27*(1), 23–31.

Ogata, Y. (1988). Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical Association*, *83*(401), 9–27.

Ogata, Y. (1998). Space-time point-process models for earthquake occurrences. *Annals of the Institute of Statistical Mathematics*, *50*, 379–402.

Ogata, Y., & Vere-Jones, D. (1984). Inference for earthquake models: A self-correcting model. *Stochastic Processes and their Applications*, *17*(2), 337–347.

Okawa, M., Iwata, T., Kurashima, T., Tanaka, Y., Toda, H., & Ueda, N. (2019). Deep mixture point processes: Spatio-temporal event prediction with rich contextual information. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

Oliveira, R. I., Orenstein, P., Ramos, T., & Romano, J. V. (2024). Split conformal prediction and non-exchangeable data. *Journal of Machine Learning Research (JMLR)*, *25*(225), 1–38.

Omi, T., Ueda, N., & Aihara, K. (2019). Fully neural network based model for general temporal point processes. *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS).*

Orvieto, A., Smith, S. L., Gu, A., Fernando, A., Gulcehre, C., Pascanu, R., & De, S. (2023). Resurrecting recurrent neural networks for long sequences. *Proceedings of the 40th International Conference on Machine Learning (ICML).*

Pandey, M. D., & van der Weide, J. (2017). Stochastic renewal process models for estimation of damage cost over the life-cycle of a structure. *Structural Safety*, *67*, 27–38.

Papadopoulos, H., Proedrou, K., Vovk, V., & Gammerman, A. (2002). Inductive confidence machines for regression. *Proceedings of the 13th European Conference on Machine Learning (ECML).*

Pinson, P., & Hagedorn, R. (2011). Verification of the ecmwf ensemble forecasts of wind speed against analyses and observations. *Meteorological Applications*, *19*(4), 385–512.

Plassier, V., Makni, M., Rubashevskii, A., Moulines, E., & Panov, M. (2023). Conformal prediction for federated uncertainty quantification under label shift. *Proceedings of the 40th International Conference on Machine Learning (ICML).*

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). Numerical recipes 3rd edition: The art of scientific computing. *Cambridge University press.*

Rasmussen, J. G. (2018). Lecture notes: Temporal point processes and the conditional intensity function. *ArXiv preprint.*

Rizoiu, M.-A., Lee, Y., Mishra, S., & Xie, L. (2017a). A tutorial on Hawkes processes for events in social media. *In Frontiers of Multimedia Research*, (pp. 191–218).

Rizoiu, M.-A., Mishra, S., Kong, Q., Carman, M., & Xie, L. (2018). SIR-Hawkes: Linking epidemic models and hawkes processes to model diffusions in finite populations. *Proceedings of the 27th International Conference on World Wide Web (WWW).*

Rizoiu, M.-A., Xie, L., Sanner, S., Cebrian, M., Yu, H., & Van Hentenryck, P.

(2017b). Expecting to be HIP: Hawkes intensity processes for social media popularity. *Proceedings of the 26th International Conference on World Wide Web (WWW).*

Romano, Y., Patterson, E., & Candes, E. (2019). Conformalized quantile regression. *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS).*

Romano, Y., Sesia, M., & Candès, E. J. (2020). Classification with valid and adaptive coverage. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS).*

Ross, S. M. (2012). Simulation. *Academic Press.*

Rotondi, R., & Varini, E. (2019). Failure models driven by a self-correcting point process in earthquake occurrence modeling. *Stochastic Environmental Research and Risk Assessment*, *33*, 709–724.

Ruder, S. (2017). An overview of gradient descent optimization algorithms. *ArXiv preprint.*

Ruder, S., Bingel, J., Augenstein, I., & Sogaard, A. (2019). Latent multi-task architecture learning. *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI).*

Sadinle, M., Lei, J., & Wasserman, L. (2019). Least ambiguous set-valued classifiers with bounded error levels. *Journal of the American Statistical Association*, *114*(525), 223–234.

Saha, A., & Ramdas, A. (2023). Testing exchangeability by pairwise betting. *ArXiv preprint.*

Sahani, M., Bohner, G., & Meyer, A. (2016). Score-matching estimators for continuous-time point-process regression models. *Proceedings of the 26th IEEE International Workshop on Machine Learning for Signal Processing (MLSP).*

Schoenberg, F., & Bolt, B. (2000). Short-term exciting, long-term correcting models for earthquake catalogs. *Bulletin of the Seismological Society of America*, *90*(4), 849–858.

Sener, O., & Koltun, V. (2018). Multi-task learning as multi-objective optimization. *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS).*

Sesia, M., & Candès, E. J. (2020). A comparison of some conformal quantile regression methods. *Stat*, *9*(1).

Sesia, M., & Romano, Y. (2021). Conformal prediction using conditional histograms. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Shafer, G., & Vovk, V. (2008). A tutorial on conformal prediction. *Journal of Machine Learning Research (JMLR)*, *9*, 371–421.

Sharma, K., Zhang, Y., Ferrara, E., & Liu, Y. (2021). Identifying coordinated accounts on social media through hidden influence and group behaviours. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

Shchur, O. (2022). Modeling continuous-time event data with neural temporal point processes. *PhD Thesis, TUM School of Computation, Information and Technology, Technische Universitat Munchen*.

Shchur, O., Biloš, M., & Günnemann, S. (2020a). Intensity-free learning of temporal point processes. *Proceedings of the 2020 International Conference on Learning Representations (ICLR)*.

Shchur, O., Gao, N., Biloš, M., & Günnemann, S. (2020b). Fast and flexible temporal point processes with triangular maps. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*.

Shchur, O., Türkmen, A. C., Januschowski, T., Gasthaus, J., & Günnemann, S. (2021a). Detecting anomalous event sequences with temporal point processes. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Shchur, O., Türkmen, A. C., Januschowski, T., & Günnemann, S. (2021b). Neural temporal point processes: A review. *Proceedings of 13th Joint Conference on Artificial Intelligence (IJCAI)*.

Shen, J., Zhen, X., Worring, M., & Shao, L. (2021). Variational multi-task learning with gumbel-softmax priors. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Shi, G., Li, Q., Zhang, W., Chen, J., & Wu, X.-M. (2023). Recon: Reducing conflicting gradients from the root for multi-task learning. *Proceedings of the 2023 International Conference on Learning Representations (ICLR)*.

Sinha, A., Chen, Z., Badrinarayanan, V., & Rabinovich, A. (2018). Gradient adversarial training of neural networks. *ArXiv preprint*.

Soen, A., Mathews, A., Grixti-Cheng, D., & Xie, L. (2021). Unipoint: Universally approximating point processes intensities. *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)*.

Song, H., Diethe, T., Kull, M., & Flach, P. (2019). Distribution calibration for regression. *Proceedings of the 36th International Conference on Machine Learning (ICML)*.

Standley, T., Zamir, A. R., Chen, D., Guibas, L., Malik, J., & Savarese, S. (2020). Which tasks should be learned together in multi-task learning? *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Stankeviciute, K., M Alaa, A., & van der Schaar, M. (2021). Conformal time-series forecasting. *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS)*.

Stein, R. B. (1965). A theoretical analysis of neuronal variability. *Biophysical Journal*, *5*(2), 173–194.

Sun, S., & Yu, R. (2023). Copula conformal prediction for multi-step time series forecasting. *Proceedings of the 2023 International Conference on Learning Representations (ICLR)*.

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., & Metzler, D. (2020). Long range arena: A benchmark for efficient transformers. *Proceedings of the 2020 International Conference on Learning Representations (ICLR)*.

Thrun, S., & O'Sullivan, J. (1996). Discovering structure in multiple learning tasks: The tc algorithm. *Proceedings of the 13th International Conference on Machine Learning (ICML)*.

Thulasidasan, S., Chennupati, G., Bilmes, J., Bhattacharya, T., & Michalak, S. (2019). On mixup training: Improved calibration and predictive uncertainty for deep neural networks. *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*.

Tibshirani, R. (2023). Conformal prediction. *Advanced Topics in Statistical Learning, University of California Berkeley*.

Tibshirani, R. J., Barber, R. F., Candes, E. J., & Ramdas, A. (2019). Conformal prediction under covariate shift. *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS).*

Trivedi, R., Dai, H., Wang, Y., & Song, L. (2017). Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. *Proceedings of the 34th International Conference on Machine Learning (ICML).*

Trivedi, R., Farajtabar, M., Biswal, P., & Zha, H. (2019). Representation learning over dynamic graphs. *Proceedings of the 2019 International Conference on Learning Representations (ICLR).*

Tsyplakov, A. (2013). Evaluation of probabilistic forecasts: Proper scoring rules and moments. *Munich Personal RePEc Archive.*

Türkmen, A. C., Januschowski, T., Wang, Y., & Cemgil, A. T. (2021). Forecasting intermittent and sparse time series: A unified probabilistic framework via deep renewal processes. *PLOS One*, *16*(11).

Türkmen, A. C., Wang, B., & Smola, A. (2019). Fastpoint: Scalable deep point processes. *Proceedings of the 2019 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD).*

Upadhyay, U., De, A., & Gomez-Rodriguez, M. (2018). Deep reinforcement learning of marked temporal point processes. *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS).*

Utpala, S., & Rai, P. (2021). Quantile regularization: Towards implicit calibration of regression models. *ArXiv preprint.*

Vaicenavicius, J., Widmann, D., Andersson, C., Lindsten, F., Roll, J., & Schön, T. B. (2019). Evaluating model calibration in classification. *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS).*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS).*

Vovk, V. (2012). Conditional validity of inductive conformal predictors. *Proceedings of the 2012 Asian Conference on Machine Learning.*

Vovk, V., Gammerman, A., & Saunders, C. (1999). Machine-Learning applications of algorithmic randomness. *Proceedings of the 16th International Conference on Machine Learning (ICML).*

Vovk, V., Gammerman, A., & Shafer, G. (2005). Algorithmic learning in a random world. *Springer*.

Waghmare, G., Debnath, A., Asthana, S., & Malhotra, A. (2022). Modeling inter-dependence between time and mark in multivariate temporal point processes. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management (CIKM).*

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. R. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Wang, Y., Xie, B., Du, N., & Song, L. (2016). Isotonic hawkes processes. *Proceedings of the 33rd International Conference on Machine Learning (ICML).*

Wang, Z., Tsvetkov, Y., Firat, O., & Cao, Y. (2020). Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *Proceedings of the 2020 International Conference on Learning Representations (ICLR).*

Wen, Y., Tran, D., & Ba, J. (2020). Batchensemble: An alternative approach to efficient ensemble and lifelong learning. *Proceedings of the 2020 International Conference on Learning Representations (ICLR).*

Wenger, J., Kjellström, H., & Triebel, R. (2020). Non-parametric calibration for classification. *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS).*

Widmann, D., Lindsten, F., & Zachariah, D. (2019). Calibration tests in multiclass classification: A unifying framework. *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS).*

Wiegrebe, S., Kopper, P., Sonabend, R., Bischl, B., & Bender, A. (2024). Deep learning for survival analysis: A review. *Artificial Intelligence Review*, *57*(3).

Wienke, A. (2010). Frailty models in survival analysis. *Chapman and Hall/CRC*.

Willemain, T. R., Smart, C. N., & Schwarz, H. F. (2004). A new approach to forecasting intermittent demand for service parts inventories. *International Journal of Forecasting*, *20*(3), 375–387.

Xiao, S., Farajtabar, M., Ye, X., Yan, J., Song, L., & Zha, H. (2017a). Wasserstein learning of deep generative point process models. *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS)*.

Xiao, S., Xu, H., Yan, J., Farajtabar, M., Yang, X., Song, L., & Zha, H. (2018). Learning conditional generative models for temporal point processes. *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*.

Xiao, S., Yan, J., Chu, S. M., Yang, X., & Zha, H. (2017b). Modeling the intensity function of point process via recurrent neural networks. *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAI)*.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., & Liu, T.-Y. (2020). On layer normalization in the transformer architecture. *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Xu, C., & Xie, Y. (2021). Conformal prediction interval for dynamic time-series. *Proceedings of the 38th International Conference on Machine Learning (ICML)*.

Xu, C., & Xie, Y. (2023a). Conformal prediction for time series. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(10), 11575–11587.

Xu, C., & Xie, Y. (2023b). Sequential predictive conformal inference for time series. *Proceedings of the 40th International Conference on Machine Learning (ICML)*.

Xu, H., Luo, D., Chen, X., & Carin, L. (2017). Benefits from superposed Hawkes processes. *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Xue, S., Shi, X., Zhang, J. Y., & Mei, H. (2022). HYPRO: A hybridly normalized probabilistic model for long-horizon prediction of event sequences. *Proceedings of the 36th International Conference on Neural Information Processing Systems (NeurIPS)*..

Yan, J. (2019a). Recent advance in temporal point process : from machine learning perspective. *SJTU Technical Report*.

Yan, J. (2019b). Temporan point processes. *Tutorial Presentation on Learning Temporal Point Process for the International Joint Conference on Artificial Intelligence (IJCAI)*.

Yan, J., Liu, X., Shi, L., Li, C., & Zha, H. (2018). Improving maximum likelihood estimation of temporal point process via discriminative and adversarial learning. *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*.

Yang, C., Mei, H., & Eisner, J. (2022). Transformer embeddings of irregularly spaced events and their participants. *Proceedings of the 2022 International Conference on Learning Representations (ICLR)*.

Yang, S., & Zha, H. (2024). A variational autoencoder for neural temporal point processes with dynamic latent graphs. *Proceedings of the 38th AAAI Conference on Artificial Intelligence (AAAI)*.

Yeung, C. C. K., Sit, T., & Fujii, K. (2023). Transformer-based neural marked spatio temporal point process model for football match events analysis. *ArXiv preprint*.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., & Finn, C. (2020). Gradient surgery for multi-task learning. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*.

Yu, X., Zhao, Y., Yin, X., & Lindemann, L. (2023). Signal temporal logic control synthesis among uncontrollable dynamic agents with conformal prediction. *ArXiv preprint*.

Yuan, Y., Ding, J., Shao, C., Jin, D., & Li, Y. (2023). Spatio-temporal diffusion point processes. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*.

Zadrozny, B., & Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

Zaffran, M., Feron, O., Goude, Y., Josse, J., & Dieuleveut, A. (2022). Adaptive conformal predictions for time series. *Proceedings of the 39th International Conference on Machine Learning (ICML)*.

Zamir, A., Sax, A., Shen, W., Guibas, L., Malik, J., & Savarese, S. (2018). Taskonomy: Disentangling task transfer learning. *Proceedings of the 2018*

IEEE/CVF International Conference on Computer Vision and Pattern Recognition (CVPR).

Zeiler, M. D. (2012). Adadelta: An adaptive learning rate method. *ArXiv preprint*.

Zhang, Q., Lipani, A., Kirnap, O., & Yilmaz, E. (2020). Self-attentive hawkes processes. *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Zhang, Q., Lipani, A., & Yilmaz, E. (2021). Learning neural point processes with latent graphs. *Proceedings of the 30th International Conference on World Wide Web (WWW)*.

Zhang, Y., Kong, Q., & Zhou, F. (2023). Integration-free training for spatio-temporal multimodal covariate deep kernel point processes. *Proceedings of the 37th International Conference on Neural Information Processing Systems (NeurIPS)*.

Zhao, Q., Erdogdu, M. A., He, H. Y., Rajaraman, A., & Leskovec, J. (2015). SEISMIC: A self-exciting point process model for predicting tweet popularity. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*.

Zhao, S., Ma, T., & Ermon, S. (2020). Individual calibration with randomized forecasting. *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Zhou, K., Zha, H., & Song, L. (2013). Learning triggering kernels for multi-dimensional Hawkes processes. *Proceedings of the 30th International Conference on Machine Learning (ICML)*.

Zhou, T., Li, Y., Wu, Y., & Carlson, D. (2021a). Estimating uncertainty intervals from collaborating networks. *Journal of Machine Learning Research (JMLR)*, *22*, 11645 – 11691.

Zhou, T., Li, Y., Wu, Y., & Carlson, D. (2021b). Estimating uncertainty intervals from collaborating networks. *Journal of Machine Learning Research (JMLR)*, *22*(257), 1–47.

Zhou, Y., Lindemann, L., & Sesia, M. (2024). Conformalized adaptive forecasting of heterogeneous trajectories. *Proceedings of the 41st International Conference on Machine Learning (ICML)*.

Zhou, Z., Yang, X., Rossi, R., Zhao, H., & Yu, R. (2022). Neural point process for learning spatio-temporal event dynamics. *Proceedings of the 4th Annual Conference on Learning for Dynamics and Control*.

Zhou, Z., & Yu, R. (2023). Automatic integration for spatio-temporal neural point processes. *Proceedings of the 37th International Conference on Neural Information Processing Systems (NeurIPS)*.

Zhu, S., Yuchi, H. S., & Xie, Y. (2020). Adversarial anomaly detection for marked spatio-temporal streaming data. *Proceedings of the 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Zhu, S., Zhang, M., Ding, R., & Xie, Y. (2021). Deep fourier kernel for self-attentive point processes. *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Zuo, S., Jiang, H., Li, Z., Zhao, T., & Zha, H. (2020). Transformer Hawkes process. *Proceedings of the 37th International Conference on Machine Learning (ICML)*.

Appendices

APPENDIX A

Datasets

A.1. Real-World Event Sequence Datasets

In this section, we review the different real-world and synthetic datasets datasets employed throughout this thesis.

A.1.1 Marked Datasets

LastFM [1] (Hidasi & Tikk, 2012): This dataset comprises records of people listening to songs over time. Each sequence relates to a user, and each mark corresponds to the artist of the song.

MOOC [1] (Kumar et al., 2019): This dataset captures the activities of students on a *massive open online course* (MOOC) platform. A sequence corresponds to a student, and the mark refers to the type of activity carried out by the student, e.g. watching a video or answering a quiz.

Wikipedia [1] (Kumar et al., 2019) : Contains records of edits made to Wikipedia pages in the course of a month. Each sequence corresponds to a given page, and the marks relate to specific editors.

MIMIC2 [2] (Du et al., 2016) : Electronic health records (HER) of patients in an intensive care units for seven years. A sequence corresponds to a patient, and the marks are the types of diseases.

Github [3] (Trivedi et al., 2019) : This dataset captures the activity records of public account owners on the open-source platform GitHub, covering the period from January 2013 to December 2013. Each sequence corresponds to an account, and the marks are the types of action performed, i.e. "Watch", "Star", "Fork", "Push", "Issue", "Comment Issue", "Pull Request", "Commit".

Stack Overflow [5] (Du et al., 2016) : Contains records of the times users received a badge on the question-answering platform Stack Overflow between 2012 and 2014. Each sequence corresponds to a user, and the marks are the types of badges received, e.g. "Stellar Question", "Guru", "Great Answer".

Retweets [2] (Zhao et al., 2015) : This dataset comprises streams of retweet events following the creation of an original tweet. Each sequence corresponds to a tweet, and marks refer to the category to which the retweeter belongs based off his/her popularity, i.e. small, medium, and large number of followers.

[1]https://github.com/srijankr/jodie/
[2]https://github.com/babylonhealth/neuralTPPs
[3]https://github.com/uoguelph-mlrg/LDG

Table A.1: Marked datasets' statistics after pre-processing. MSL corresponds to the average number of events per sequence.

	Sequences	**Events**	**MSL**	**Max Len.**	**Min Len.**	**Marks**
Wikipedia	590	30472	51.6	1163	2	50
MOOC	7047	351160	49.8	416	2	50
LastFM	856	193441	226.0	6396	2	50
MIMIC2	599	1812	3.0	32	2	43
Github	173	20657	119.4	4698	3	8
Stack Overflow	7959	569688	71.6	735	40	22
Retweets	24000	2610102	108.8	264	50	3

Reddit [5] (Kumar et al., 2019): Records of one-month submissions made to various sub-reddits on the social platform Reddit. Each sequence corresponds to a user, and an event's mark refers to the specific sub-reddit to which the user responds.

A.1.2 Unmarked Datasets

Twitter [4] (Shchur et al., 2020b): This dataset contains records of tweets made over several years.

PUBG [4] (Shchur et al., 2020b): Records of players' death in the online game PUBG. Each sequence corresponds to a game, and a timestamp refers to the death of a given player.

Yelp Airport [4] (Shchur et al., 2020b): This dataset contains customers check-in times to 319 businesses on the platform Yelp at the McCarran International Airport. Each sequence corresponds to a business.

Yelp Mississauga [4] (Shchur et al., 2020b): Similar to the Yelp Airport dataset, but for 319 businesses in the city of Mississauga.

Yelp Toronto [5] (Shchur et al., 2020a) : This dataset contains sequences of reviews made by customers on the platform Yelp for 300 restaurants in the city of Toronto. Each sequence corresponds to a restaurant.

[4]https://github.com/shchur/triangular-tpp
[5]https://github.com/shchur/ifl-tpp

Table A.2: Unmarked datasets' statistics after pre-processing. MSL corresponds to the average number of events per sequence.

	Sequences	**Events**	**MSL**	**Max Len.**	**Min Len.**
Reddit Submissions	1094	1235128	1129.0	2658	362
Reddit Comments	1355	400933	295.9	2137	4
Taxi	182	17904	98.4	140	12
Twitter	1804	29862	16.6	169	2
Yelp Toronto	300	215146	717.2	2868	424
Yelp Airport	319	9716	30.5	55	9
Yelp Mississauga	319	17621	55.2	107	3
PUBG	3001	229703	76.5	97	26

Reddit Comments [4](Shchur et al., 2020b) : Records of comments in reply to Reddit discussion threads within 24hrs of the original post submission. The data is recorded between 2018 and 2020, and each sequence corresponds to a discussion thread.

Reddit Submissions [4](Shchur et al., 2020b) : This dataset comprises records of submissions made to a political sub-Reddit between 2017 and 2020. Each sequence corresponds to a 24 hours window.

Taxi [4] (Shchur et al., 2020b) : Contains the records of taxi pick-ups in the South of Manhattan. Each sequence corresponds to a taxi, and timestamps refer to the times at which passengers were taken on board.

A.2. Simulated Datasets

In Chapters 3 and 5, we used a synthetic dataset generated from a multidimensional Hawkes process with exponential kernels, i.e.

$$\lambda_k^*(t) = \lambda_k + \sum_{k'=1}^{K} \sum_{(t_j,k_j)\in\mathcal{H}_t^{k'}} \gamma_{k,k'} \exp\left(-\beta_{k,k'}(t - t_j)\right), \tag{A.1}$$

where

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_K \end{bmatrix} \in \mathbb{R}_+^K, \qquad \boldsymbol{\gamma} = \begin{bmatrix} \gamma_{1,1} & \cdots & \gamma_{1,K} \\ \vdots & \ddots & \vdots \\ \gamma_{K,1} & \cdots & \gamma_{K,K} \end{bmatrix} \in \mathbb{R}_+^{K\times K}, \tag{A.2}$$

and

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_{1,1} & \cdots & \beta_{1,K} \\ \vdots & \ddots & \vdots \\ \beta_{K,1} & \cdots & \beta_{K,K} \end{bmatrix} \in \mathbb{R}_+^{K\times K}, \tag{A.3}$$

and $\mathcal{H}_t^{k'} = \{(t_j, k_j) \in \mathcal{S} \mid t_j < t, k_j = k'\}$. For simulation, we set the value of the parameters $\boldsymbol{\lambda}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\beta}$ to

$$\boldsymbol{\lambda} = \begin{pmatrix} 0.2 \\ 0.6 \\ 0.1 \\ 0.7 \\ 0.9 \end{pmatrix} \boldsymbol{\gamma} = \begin{pmatrix} 0.25 & 0.13 & 0.13 & 0.13 & 0.13 \\ 0.13 & 0.35 & 0.13 & 0.13 & 0.13 \\ 0.13 & 0.13 & 0.2 & 0.13 & 0.13 \\ 0.13 & 0.13 & 0.13 & 0.3 & 0.13 \\ 0.13 & 0.13 & 0.13 & 0.13 & 0.25 \end{pmatrix} \boldsymbol{\beta} = \begin{pmatrix} 4.1 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 2.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 6.2 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 4.9 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 4.1 \end{pmatrix},$$

where the matrix $\boldsymbol{\gamma}$ is scaled to have a spectral radius of approximately 0.8, guaranteeing stationarity of the process (Bacry et al., 2020). The process essentially corresponds to a marked process with $K = 5$ marks. All simulations are carried out with the library *tick* [6] (Bacry et al., 2018).

[6]https://x-datainitiative.github.io/tick/index.html

APPENDIX B

Supplementary Material for Chapter 2

Suppose that $\boldsymbol{e}_{i-1} = (t_{i-1}, k_{i-1})$ is the last observed event in a sequence $\mathcal{S}$. Let $f^*(t,k)$ be the joint PDF of arrival times and marks, and let $\lambda_k^*(t)$ be the marked intensity functions of the process. For $t > t_{i-1}$, $f^*(t,k)$ and $\lambda_k^*(t)$ verify

$$f^*(t,k) = \lambda_k^*(t)\exp\left(-\Lambda^*(t)\right), \tag{B.1}$$

where $\Lambda^*(t) = \sum_{k=1}^K \int_{t_{i-1}}^t \lambda^*(s)ds$ is the ground compensator of the process.

Proof. By definition of $\lambda^*(t)$ Rasmussen (2018), we have:

$$\lambda^*(t) = \sum_{k=1}^K \lambda_k^*(t) = \frac{f^*(t)}{1 - F^*(t)} \tag{B.2}$$

$$= \frac{\frac{d}{dt}F^*(t)}{1 - F^*(t)} \tag{B.3}$$

$$= -\frac{d}{dt}\log\left(1 - F^*(t)\right). \tag{B.4}$$

Integrating both sides from t_{i-1} to t, we get

$$\Lambda^*(t) = \int_{t_{i-1}}^t \lambda^*(s)ds = \int_{t_{i-1}}^t -d\log\left(1 - F^*(s)\right) \tag{B.5}$$

$$= -\log\left(1 - F^*(t)\right) + \log\left(1 - \underbrace{F^*(t_{i-1})}_{=0}\right), \tag{B.6}$$

where $F^*(t_{i-1}) = 0$ results from the point process being simple, i.e. two events occur simultaneously with probability 0. Rearranging the terms in (B.6), we find

$$F^*(t) = 1 - \exp\left(-\Lambda^*(t)\right) = 1 - \exp\left(-\sum_{k=1}^K \Lambda_k^*(t)\right). \tag{B.7}$$

Differentiating (B.7) with respect to t gives

$$f^*(t) = \frac{d}{dt}F^*(t) = \lambda^*(t)\exp\left(-\Lambda^*(t)\right). \tag{B.8}$$

Finally, given that $\lambda_k^*(t) = \lambda^*(t)p^*(k|t)$, we have

$$f^*(t,k) = f^*(t)p^*(k|t) = \lambda_k^*(t)\exp\left(-\Lambda^*(t)\right) \tag{B.9}$$

□

APPENDIX C

Supplementary Material for Chapter 3

C.1. Details on Statistical Tests

in Chapter 3, we introduced statistical tests to conduct pairwise comparisons between models for each metric separately. This section, which is mainly inspired from Demšar (2006) and García & Herrera (2008), briefly presents the Friedman and Holm's post-hoc tests that we employed in Section 3.3.4.

Friedman test. The Friedman test (Friedman, 1937, 1940) is a non-parametric statistical test that enables to determine differences between the performance of multiple models measured on multiple datasets. For each dataset separately, the Friedmann test first ranks all models based on their respective performance with respect to a metric of interest, say $\mathcal{L}_T$. Then, for a given dataset, the best performing model on $\mathcal{L}_T$ is assigned rank 1, the second rank 2, and so on.

Suppose that we have M models evaluated on D datasets. Denoting r_m^d as the rank of model m on dataset d, the Friedman test then computes the average ranks $\hat{r}_m = \frac{1}{D}\sum_{d=1}^{D} r_m^d$ of all M models and evaluate the statistic

$$\chi_F^2 = \frac{12D}{M(M+1)}\left[\sum_{m=1}^{M}\hat{r}_m^2 - \frac{M(M+1)^2}{4}\right]. \tag{C.1}$$

Under the null hypothesis, which states that all average ranks $\hat{r}_m$ are equal, the statistic χ_F^2 follows a chi-squared distribution with $M-1$ degrees of freedom. For a given significance level α, rejecting the null hypothesis implies that at least one statistically significant difference exists among the average ranks. In such case, we can proceed with an appropriate post-hoc test to compare pairwise differences between models, adjusting for multiple comparisons. A common choice is Holm's post-hoc test (Holm, 1979).

Holm's post-hoc test allows to control the family-wise error rate when comparing the performance of all models against each other. For any two models m_i and m_j with average ranks $\hat{r}_i$ and $\hat{r}_j$, respectively, the test computes the following statistic

$$z_{ij} = \frac{\hat{r}_i - \hat{r}_j}{\sqrt{\frac{M(M+1)}{6D}}}. \tag{C.2}$$

The $n = M(M-1)/2$ values z_{ij} (one for each pairwise comparison) are then used to obtain the corresponding p-values from a table of normal distributions. To account for multiple hypothesis testing, Holm's post-hoc test adjusts the significance level α using a step-down procedure. Let $p_1 \geq p_2 \geq \ldots \geq p_n$ be the ordered set of n p-values. To accept or reject the null hypothesis associated to p_1, Holm's post-hoc test compares p_1 with the adjusted significance level α/n. If $p_1 < \alpha/n$, the null hypothesis is rejected and we pursue by comparing p_2 with $\alpha/(n-1)$. Once we reach a null hypothesis that cannot be rejected, we stop, and all subsequent null hypothesis are kept. From this point, we can conclude of a statistical difference at significance level α for all pairwise comparisons whose null hypothesis have been rejected by the procedure.

C.2. Results for Unmarked Datasets

We report the comparison of different event encoding mechanisms and history encoders for unmarked datasets in Table C.1 and Table C.2, respectively, where we also added the worst score per variation of component. The results of the combinations that performed best with respect to the $\mathcal{L}_T$ can be found in Table C.3. Overall, our findings regarding unmarked datasets align with the ones described in Section 3.4. However, we note that most decoders show a lower PCE, and hence, improved calibration with respect to the distribution of inter-arrival time compared to marked datasets. The reliability diagrams displayed in Figure C.1 for unmarked datasets do indeed confirm this observation. In Figure C.2, we present the CD diagrams illustrating the pairwise differences between all combinations of Table C.3 for the $\mathcal{L}_T$ and PCE metrics. While there is a noticeable distinction between the average ranks of the considered decoders, the data does not provide sufficient evidence to conclude of statistical differences at the $\alpha = 0.05$ significance level. As for marked datasets, this can be explained by a large number of pairwise comparisons, resulting in high adjusted p-values.

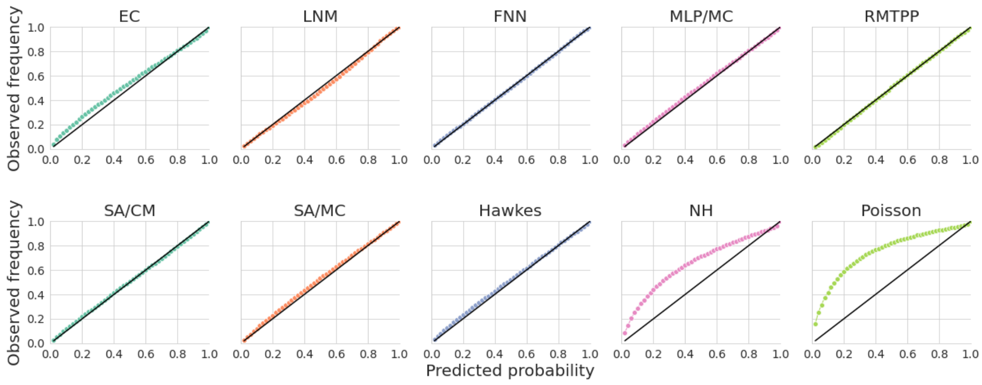


Figure C.1: Reliability diagrams for the time predictive distributions of the models of Table C.3, averaged over all unmarked datasets. The bold black line corresponds to perfect probabilistic calibration.

Table C.1: Mean, median, and worst scores, as well as average ranks per decoder and variation of event encoding, for unmarked datasets. Refer to Section 3.3.5 for details on the aggregation procedure. Best scores are highlighted in bold.

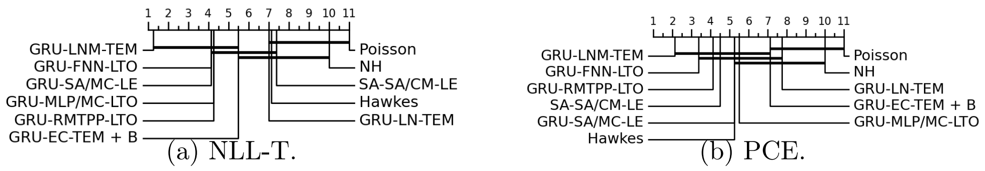
	Unmarked Datasets							
	$\mathcal{L}_T$				PCE			
	Mean	Median	Worst	Rank	Mean	Median	Worst	Rank
EC-TO	0.25	0.24	0.6	3.5	0.07	0.06	0.16	3.25
EC-LTO	0.25	0.26	0.64	3.5	0.07	0.06	0.16	3.62
EC-TEM	**-0.21**	**-0.26**	**0.31**	**1.0**	**0.05**	**0.03**	**0.14**	**1.25**
EC-LE	-0.09	-0.04	0.46	2.0	0.05	0.03	0.15	1.88
LNM-TO	-1.15	-0.5	-0.26	3.0	**0.01**	**0.01**	**0.01**	3.12
LNM-LTO	-1.17	-0.57	-0.25	3.38	0.01	0.01	0.01	2.25
LNM-TEM	**-1.46**	**-0.79**	**-0.51**	**1.12**	0.01	0.01	0.01	**1.5**
LNM-LE	-0.98	-0.59	-0.44	2.5	0.01	0.01	0.01	3.12
FNN-TO	0.7	0.6	1.06	2.62	0.09	0.09	0.15	2.62
FNN-LTO	**-0.19**	**-0.25**	1.69	**1.25**	**0.02**	**0.01**	**0.06**	**1.0**
FNN-LE	0.56	0.5	**1.03**	2.12	0.08	0.07	0.15	2.38
MLP/MC-TO	-0.02	0.0	0.1	3.75	0.05	0.04	0.08	3.62
MLP/MC-LTO	-0.14	-0.17	0.06	2.5	**0.03**	0.03	0.08	**1.88**
MLP/MC-TEM	-0.23	-0.21	**-0.04**	2.12	0.04	0.03	0.09	2.62
MLP/MC-LE	**-0.3**	**-0.33**	-0.04	**1.62**	0.03	**0.02**	**0.07**	**1.88**
RMTPP-TO	-0.07	-0.07	0.11	3.75	0.04	0.03	0.08	3.75
RMTPP-LTO	-0.21	-0.13	0.2	2.75	**0.02**	**0.02**	**0.03**	2.0
RMTPP-TEM	**-0.37**	**-0.32**	**-0.25**	**1.25**	0.03	0.02	0.07	**1.88**
RMTPP-LE	-0.3	-0.28	-0.14	2.25	0.03	0.02	0.08	2.38
SA/CM-TO	0.71	0.24	4.19	2.38	0.05	0.03	0.14	2.62
SA/CM-LTO	1.08	0.23	4.93	2.12	0.05	0.03	0.14	2.25
SA/CM-LE	**-0.22**	**-0.18**	**0.19**	**1.5**	**0.02**	**0.01**	**0.05**	**1.12**
SA/MC-TO	0.78	0.66	1.17	3.5	0.09	0.08	0.16	3.38
SA/MC-LTO	0.68	0.69	1.09	3.5	0.09	0.1	0.15	3.38
SA/MC-TEM	-0.42	-0.39	**-0.18**	1.88	**0.02**	**0.01**	**0.07**	**1.62**
SA/MC-LE	**-0.5**	**-0.49**	0.17	**1.12**	0.03	0.02	0.08	**1.62**

Table C.2: Mean, median, and worst scores, as well as average ranks per decoder and variation of history encoder, for unmarked datasets. Refer to Section 3.3.5 for details on the aggregation procedure. Best scores are highlighted in bold.

	Unmarked Datasets							
	$\mathcal{L}_T$				PCE			
	Mean	Median	Worst	Rank	Mean	Median	Worst	Rank
CONS-EC	1.03	0.87	2.14	3.0	0.1	0.09	0.2	3.0
SA-EC	0.48	0.54	0.85	2.0	0.08	0.07	0.17	2.0
GRU-EC	**-0.38**	**-0.4**	**0.25**	**1.0**	**0.04**	**0.02**	**0.14**	**1.0**
CONS-LNM	-0.83	-0.16	0.13	2.75	**0.01**	**0.01**	0.02	2.25
SA-LNM	-0.69	-0.26	-0.14	2.25	**0.01**	**0.01**	0.02	2.25
GRU-LNM	**-1.7**	**-0.92**	**-0.61**	**1.0**	**0.01**	**0.01**	**0.01**	**1.5**
CONS-FNN	0.63	0.53	1.26	2.62	**0.06**	**0.05**	0.13	2.12
SA-FNN	0.46	0.35	1.06	2.25	**0.06**	0.06	**0.1**	**1.62**
GRU-FNN	**0.25**	**0.16**	**0.94**	**1.12**	**0.06**	0.06	0.11	2.25
CONS-MLP/MC	0.46	0.5	0.74	2.88	0.06	0.06	0.11	2.88
SA-MLP/MC	0.15	0.17	0.33	2.12	0.05	0.04	0.09	2.0
GRU-MLP/MC	**-0.49**	**-0.47**	**-0.33**	**1.0**	**0.03**	**0.02**	**0.06**	**1.12**
CONS-RMTPP	0.31	0.26	0.79	2.88	0.04	0.03	**0.06**	2.38
SA-RMTPP	0.04	0.01	0.29	2.12	0.04	0.03	0.07	2.38
GRU-RMTPP	**-0.52**	**-0.54**	**-0.33**	**1.0**	**0.02**	**0.01**	**0.06**	**1.25**
CONS-SA/CM	1.03	0.68	2.87	2.88	0.05	0.04	0.11	2.62
SA-SA/CM	0.53	0.26	**2.7**	1.62	**0.04**	**0.03**	**0.1**	1.88
GRU-SA/CM	**0.51**	**0.2**	3.0	**1.5**	**0.04**	**0.03**	**0.1**	**1.5**
CONS-SA/MC	0.52	0.49	0.86	2.88	0.07	**0.06**	0.11	2.62
SA-SA/MC	0.17	0.21	**0.34**	1.75	**0.06**	0.07	**0.08**	**1.5**
GRU-SA/MC	**0.1**	**0.14**	0.38	**1.38**	**0.06**	0.07	**0.08**	1.88

Table C.3: Mean, median, worst scores, and average ranks of the best combinations per decoder on $\mathcal{L}_T$ across all unmarked datasets. Best results are highlighted in bold.

	Unmarked Datasets							
	$\mathcal{L}_T$				PCE			
	Mean	Median	Worst	Rank	Mean	Median	Worst	Rank
GRU-EC-TEM + B	-0.41	-0.46	0.16	5.5	0.04	0.02	0.13	7.12
GRU-LNM-TEM	**-1.96**	**-0.95**	**-0.62**	**1.25**	**0.01**	**0.01**	**0.01**	**2.12**
GRU-LN-TEM	-0.25	-0.25	0.41	7.0	0.03	0.03	0.07	7.75
GRU-FNN-LTO	-0.48	-0.59	1.48	4.12	**0.01**	**0.01**	0.05	3.38
GRU-MLP/MC-LTO	-0.57	-0.56	-0.44	4.25	0.02	**0.01**	0.05	5.5
GRU-RMTPP-LTO	-0.6	-0.55	-0.28	4.25	**0.01**	**0.01**	0.03	4.12
SA-SA/CM-LE	-0.23	-0.15	0.0	7.38	0.02	**0.01**	0.04	4.5
GRU-SA/MC-LE	-0.56	-0.55	0.4	4.12	0.03	0.02	0.1	5.25
Hawkes	-0.22	-0.16	0.16	7.12	0.02	**0.01**	0.06	5.25
Poisson	4.23	2.54	12.29	11.0	0.25	0.26	0.47	11.0
NH	2.29	1.11	8.2	10.0	0.16	0.15	0.3	10.0



(a) NLL-T. (b) PCE.

Figure C.2: Critical Distance (CD) diagrams per metric at the $\alpha = 0.05$ significance level for all models of Table C.3. Average ranks are displayed on top, and a bold line joins models that are not statistically different.

C.3. Additional Results

In Table C.4, we report the results with respect to all time-related metrics ($\mathcal{L}_T$, PCE) for all models of Table 3.6 on marked datasets individually, while results with respect to marked related metrics ($\mathcal{L}_M$, ECE, F1-score) are given in Table C.5. In turn, results with respect to time-related metrics for models of Table C.3 on unmarked datasets are reported in Table C.6.

Finally, Table C.7 summarizes all the variations of event encoder, history encoder and decoder that have been considered in the experimental study of Chapter 3. The column "Unmarked" indicates whether the combination was appropriate for training on unmarked datasets.

Table C.4: Results with respect to the $\mathcal{L}_T$ and PCE for all models in Table 3.6 on each marked dataset. Standard error across all splits is reported in parenthesis, and best results are highlighted in bold.

				$\mathcal{L}_T$				
	LastFM	MOOC	Wikipedia	Github	MIMIC2	Hawkes	SO	Retweets
GRU-EC-LE	-1102.45 (49.95)	-105.23 (3.43)	-153.79 (14.4)	-290.58 (55.37)	0.91 (0.05)	-78.05 (0.78)	-82.38 (1.34)	-546.95 (2.86)
GRU-LNM-TO	**-1363.43 (59.03)**	-275.49 (3.53)	-242.45 (30.64)	-378.09 (57.75)	**0.13 (0.33)**	-75.62 (1.01)	-79.21 (2.32)	-613.77 (14.25)
GRU-LN-LEWL	-1351.97 (60.26)	-275.49 (3.53)	-226.75 (31.05)	-369.38 (57.05)	0.66 (0.07)	-73.68 (0.76)	-74.24 (1.42)	-582.47 (2.47)
GRU-FNN-LCONCAT	-1355.58 (59.91)	-287.65 (3.56)	-239.87 (29.03)	-371.57 (58.27)	1.76 (0.11)	-78.33 (0.8)	**-87.99 (1.41)**	-596.29 (2.65)
GRU-MLP/MC-LCONCAT	-1338.38 (56.91)	-275.43 (3.55)	-186.58 (47.99)	-325.3 (57.09)	1.82 (0.14)	-78.49 (0.79)	-87.82 (1.15)	-577.8 (1.7)
GRU-RMTPP-LCONCAT	-1331.75 (58.87)	-268.91 (3.25)	**-267.41 (24.59)**	**-382.4 (61.05)**	1.66 (0.08)	-78.49 (0.79)	-83.99 (1.47)	-570.73 (2.18)
GRU-SA/CM-LE	-1299.86 (57.52)	-280.73 (3.8)	-200.89 (47.85)	-331.96 (60.72)	1.23 (0.23)	-78.01 (0.79)	-84.05 (1.3)	-545.31 (13.15)
GRU-SA/MC-LE	-1349.8 (60.0)	-277.66 (3.03)	-207.39 (51.79)	-348.82 (56.8)	0.47 (0.12)	-78.5 (0.76)	-85.17 (1.33)	-581.05 (2.59)
Hawkes	-1189.48 (55.2)	-235.9 (3.09)	332.83 (93.92)	-308.42 (57.77)	4.49 (0.24)	**-78.66 (0.82)**	-83.18 (1.4)	-554.6 (2.15)
Poisson	-747.35 (46.13)	-51.74 (1.01)	-57.32 (16.36)	-142.58 (26.06)	2.98 (0.12)	-71.6 (0.7)	-73.6 (1.11)	-234.81 (0.5)
NH	-788.33 (33.0)	-76.64 (1.21)	-102.72 (12.9)	-168.39 (29.39)	1.73 (0.11)	-72.54 (0.67)	-73.79 (1.06)	-236.29 (4.38)
GRU-EC-TEMWL	-1002.13 (37.43)	-97.11 (1.01)	-144.57 (14.46)	-269.0 (51.2)	1.82 (0.11)	-78.0 (0.76)	-82.78 (1.42)	-535.75 (1.39)
GRU-LNM-CONCAT	-1363.78 (59.77)	**-289.3 (3.06)**	-261.79 (37.08)	-367.41 (56.35)	0.59 (0.22)	-76.57 (1.18)	-85.12 (3.03)	**-621.33 (22.55)**
GRU-LN-CONCAT	-1348.27 (57.29)	-280.3 (3.15)	-228.31 (31.13)	-367.41 (56.35)	0.97 (0.05)	-73.68 (0.76)	-81.42 (1.34)	-583.42 (2.36)
GRU-MLP/MC-TEMWL	-1192.81 (46.5)	-232.97 (1.31)	-135.52 (20.11)	-285.31 (51.34)	1.41 (0.11)	-78.59 (0.79)	-85.9 (1.42)	-577.61 (2.41)
GRU-RMTPP-TEMWL + B	-1118.02 (52.48)	-202.08 (9.0)	-136.01 (22.52)	-276.45 (51.41)	1.34 (0.14)	-78.48 (0.79)	-82.96 (1.36)	-565.46 (3.72)
GRU-SA/CM-LEWL	-1274.49 (50.07)	-268.28 (5.07)	-174.11 (29.42)	-316.81 (51.14)	1.5 (0.18)	-77.52 (0.65)	-80.23 (1.5)	-555.59 (9.08)
GRU-SA/MC-TEMWL	-1243.21 (47.96)	-257.48 (3.25)	-158.02 (27.61)	-322.72 (53.94)	1.77 (0.11)	-78.04 (0.73)	-86.24 (1.29)	-569.34 (4.58)

				PCE				
	LastFM	MOOC	Wikipedia	Github	MIMIC2	Hawkes	SO	Retweets
GRU-EC-LE	0.27 (0.0)	0.37 (0.01)	0.35 (0.01)	0.19 (0.01)	0.08 (0.01)	0.02 (0.0)	0.02 (0.0)	0.09 (0.0)
GRU-LNM-TO	**0.02 (0.0)**	0.04 (0.0)	0.22 (0.07)	0.03 (0.01)	0.05 (0.01)	0.02 (0.01)	0.03 (0.01)	**0.01 (0.0)**
GRU-LN-LEWL	0.03 (0.0)	0.04 (0.0)	0.28 (0.05)	0.05 (0.0)	0.07 (0.01)	0.03 (0.0)	0.05 (0.0)	**0.01 (0.0)**
GRU-FNN-LCONCAT	**0.02 (0.0)**	**0.02 (0.0)**	**0.07 (0.01)**	**0.02 (0.0)**	0.06 (0.0)	**0.0 (0.0)**	**0.0 (0.0)**	**0.01 (0.0)**
GRU-MLP/MC-LCONCAT	0.08 (0.01)	0.13 (0.01)	0.27 (0.03)	0.14 (0.01)	0.05 (0.01)	0.01 (0.0)	0.01 (0.0)	0.02 (0.0)
GRU-RMTPP-LCONCAT	0.06 (0.0)	0.07 (0.0)	0.1 (0.01)	**0.02 (0.0)**	0.04 (0.0)	0.01 (0.0)	0.01 (0.0)	0.03 (0.0)
GRU-SA/CM-LE	0.05 (0.01)	**0.02 (0.0)**	0.18 (0.05)	0.09 (0.01)	0.05 (0.01)	**0.0 (0.0)**	**0.0 (0.0)**	**0.01 (0.0)**
GRU-SA/MC-LE	0.06 (0.01)	0.13 (0.0)	0.24 (0.04)	0.12 (0.01)	0.04 (0.01)	0.01 (0.0)	0.01 (0.0)	**0.01 (0.0)**
Hawkes	0.21 (0.0)	0.2 (0.0)	0.19 (0.01)	0.15 (0.01)	0.06 (0.0)	**0.0 (0.0)**	0.01 (0.0)	0.03 (0.0)
Poisson	0.38 (0.01)	0.4 (0.0)	0.34 (0.01)	0.33 (0.02)	0.07 (0.0)	0.04 (0.0)	0.04 (0.0)	0.36 (0.0)
NH	0.35 (0.01)	0.4 (0.0)	0.35 (0.01)	0.3 (0.02)	0.07 (0.0)	0.04 (0.0)	0.05 (0.0)	0.36 (0.0)
GRU-EC-TEMWL	0.29 (0.01)	0.39 (0.0)	0.35 (0.01)	0.19 (0.01)	0.06 (0.0)	0.02 (0.0)	0.02 (0.0)	0.12 (0.01)
GRU-LNM-CONCAT	**0.02 (0.01)**	0.03 (0.01)	0.16 (0.06)	0.05 (0.01)	**0.03 (0.01)**	0.01 (0.01)	0.03 (0.01)	**0.01 (0.0)**
GRU-LN-CONCAT	0.04 (0.01)	0.04 (0.0)	0.28 (0.06)	0.05 (0.01)	0.05 (0.01)	0.03 (0.0)	0.04 (0.0)	**0.01 (0.0)**
GRU-MLP/MC-TEMWL	0.21 (0.01)	0.24 (0.01)	0.33 (0.01)	0.19 (0.01)	**0.03 (0.0)**	0.01 (0.0)	0.01 (0.0)	0.02 (0.0)
GRU-RMTPP-TEMWL + B	0.25 (0.0)	0.27 (0.01)	0.32 (0.01)	0.18 (0.01)	**0.03 (0.0)**	0.01 (0.0)	0.01 (0.0)	0.04 (0.0)
GRU-SA/CM-LEWL	0.09 (0.01)	0.09 (0.02)	0.28 (0.02)	0.11 (0.01)	0.05 (0.0)	0.01 (0.0)	0.02 (0.01)	**0.01 (0.0)**
GRU-SA/MC-TEMWL	0.17 (0.01)	0.19 (0.01)	0.33 (0.01)	0.13 (0.01)	0.05 (0.0)	0.01 (0.0)	0.01 (0.0)	0.03 (0.0)

Table C.5: Results with respect to the $\mathcal{L}_M$, ECE and F1-score for all models in Table 3.6 on each marked dataset. Standard error across all splits is reported in parenthesis, and best results are highlighted in bold.

				$\mathcal{L}_M$				
	LastFM	MOOC	Wikipedia	Github	MIMIC2	Hawkes	SO	Retweets
GRU-EC-LE	870.89 (30.27)	153.58 (1.94)	272.21 (23.78)	155.36 (21.1)	4.43 (0.16)	111.67 (0.55)	120.66 (1.17)	89.74 (0.16)
GRU-LNM-TO	872.47 (30.81)	151.54 (1.4)	297.42 (29.5)	153.92 (21.54)	4.3 (0.15)	111.67 (0.55)	122.73 (0.87)	88.71 (0.45)
GRU-LN-LEWL	872.63 (31.02)	151.54 (1.4)	289.27 (28.85)	156.31 (21.1)	4.31 (0.15)	111.66 (0.55)	121.66 (0.67)	87.91 (0.25)
GRU-FNN-LCONCAT	813.09 (30.86)	110.71 (5.25)	298.11 (33.11)	128.36 (18.2)	4.36 (0.14)	110.87 (0.59)	109.31 (0.79)	83.38 (0.23)
GRU-MLP/MC-LCONCAT	827.77 (30.3)	128.44 (9.32)	638.25 (187.65)	154.69 (21.38)	4.58 (0.24)	111.4 (0.61)	111.98 (1.42)	83.37 (0.3)
GRU-RMTPP-LCONCAT	834.05 (40.33)	89.92 (1.67)	327.16 (77.34)	122.9 (16.42)	**2.29 (0.23)**	110.34 (0.56)	107.33 (0.34)	**82.63 (0.18)**
GRU-SA/CM-LE	871.21 (30.14)	171.64 (2.7)	377.53 (63.96)	154.8 (21.11)	4.39 (0.16)	111.69 (0.54)	121.9 (0.8)	90.35 (0.29)
GRU-SA/MC-LE	869.85 (30.36)	177.49 (2.28)	407.99 (74.55)	156.81 (22.86)	4.55 (0.17)	111.71 (0.54)	117.3 (0.91)	88.5 (0.21)
Hawkes	**514.13 (16.19)**	112.14 (1.26)	**144.79 (11.89)**	**122.44 (15.5)**	12.84 (0.23)	110.83 (0.54)	114.99 (0.75)	89.5 (0.14)
Poisson	888.49 (27.18)	188.13 (2.37)	534.95 (131.14)	168.37 (21.04)	11.25 (0.46)	113.46 (0.54)	132.93 (0.74)	92.11 (0.15)
NH	869.55 (30.47)	184.55 (2.35)	272.66 (21.64)	161.84 (21.59)	4.66 (0.11)	111.99 (0.54)	131.2 (0.74)	91.1 (0.15)
GRU-EC-TEMWL	830.42 (33.2)	98.53 (1.65)	298.61 (38.42)	150.4 (21.71)	3.06 (0.18)	110.06 (0.53)	108.8 (0.93)	85.55 (0.9)
GRU-LNM-CONCAT	752.71 (25.47)	92.15 (1.81)	267.73 (17.66)	139.91 (21.89)	2.5 (0.12)	110.18 (0.61)	**107.24 (0.73)**	84.33 (1.08)
GRU-LN-CONCAT	744.04 (29.37)	92.58 (1.3)	277.51 (22.56)	139.91 (21.89)	2.8 (0.31)	110.37 (0.52)	108.25 (1.11)	82.83 (0.2)
GRU-MLP/MC-TEMWL	837.78 (41.47)	136.16 (8.43)	454.48 (134.47)	149.36 (20.32)	3.81 (0.29)	111.59 (0.47)	109.07 (1.4)	86.36 (0.85)
GRU-RMTPP-TEMWL + B	750.69 (32.06)	**88.58 (1.71)**	264.53 (41.57)	136.6 (18.94)	2.43 (0.23)	**109.99 (0.55)**	108.73 (1.21)	83.94 (0.38)
GRU-SA/CM-LEWL	871.49 (30.22)	165.08 (7.25)	281.76 (22.54)	154.18 (21.25)	4.17 (0.19)	111.7 (0.55)	120.48 (1.8)	87.73 (0.79)
GRU-SA/MC-TEMWL	866.4 (29.6)	157.69 (6.6)	340.59 (49.61)	147.83 (22.54)	3.2 (0.21)	111.32 (0.6)	110.87 (0.56)	87.76 (0.29)
				ECE				
	LastFM	MOOC	Wikipedia	Github	MIMIC2	Hawkes	SO	Retweets
GRU-EC-LE	0.5 (0.0)	0.44 (0.01)	0.48 (0.01)	0.31 (0.01)	0.38 (0.0)	0.44 (0.0)	0.23 (0.02)	0.38 (0.01)
GRU-LNM-TO	0.5 (0.0)	0.45 (0.01)	0.49 (0.0)	0.4 (0.01)	0.41 (0.02)	0.44 (0.0)	0.28 (0.02)	0.28 (0.03)
GRU-LN-LEWL	0.5 (0.0)	0.45 (0.01)	0.5 (0.0)	0.42 (0.01)	0.4 (0.02)	0.44 (0.0)	0.26 (0.01)	0.23 (0.02)
GRU-FNN-LCONCAT	0.43 (0.05)	0.31 (0.04)	0.5 (0.01)	0.15 (0.0)	0.46 (0.0)	0.39 (0.01)	**0.03 (0.0)**	**0.06 (0.0)**
GRU-MLP/MC-LCONCAT	0.42 (0.02)	0.21 (0.04)	0.47 (0.02)	0.14 (0.03)	0.32 (0.03)	0.44 (0.01)	**0.03 (0.01)**	**0.06 (0.0)**
GRU-RMTPP-LCONCAT	0.43 (0.03)	**0.06 (0.01)**	0.36 (0.06)	0.15 (0.01)	0.16 (0.02)	0.4 (0.0)	0.15 (0.02)	0.08 (0.02)
GRU-SA/CM-LE	0.5 (0.0)	0.45 (0.0)	0.5 (0.0)	0.43 (0.01)	0.46 (0.0)	0.44 (0.0)	0.26 (0.04)	0.41 (0.0)
GRU-SA/MC-LE	0.5 (0.0)	0.42 (0.01)	0.49 (0.0)	0.22 (0.02)	0.33 (0.01)	0.44 (0.0)	0.15 (0.01)	0.27 (0.02)
Hawkes	**0.03 (0.0)**	0.14 (0.0)	**0.11 (0.03)**	**0.07 (0.02)**	0.19 (0.01)	**0.34 (0.01)**	0.09 (0.02)	0.19 (0.0)
Poisson	0.5 (0.0)	0.5 (0.0)	0.5 (0.0)	0.49 (0.01)	0.47 (0.0)	0.48 (0.0)	0.46 (0.0)	0.46 (0.0)
NH	0.5 (0.0)	0.5 (0.0)	0.5 (0.0)	0.47 (0.01)	0.46 (0.0)	0.48 (0.0)	0.46 (0.0)	0.45 (0.01)
GRU-EC-TEMWL	0.48 (0.01)	0.21 (0.03)	0.49 (0.0)	0.27 (0.04)	0.17 (0.02)	0.4 (0.0)	0.21 (0.01)	0.12 (0.02)
GRU-LNM-CONCAT	0.35 (0.03)	0.18 (0.02)	0.36 (0.07)	0.31 (0.04)	0.15 (0.01)	0.4 (0.0)	0.12 (0.02)	0.08 (0.01)
GRU-LN-CONCAT	0.29 (0.05)	0.13 (0.02)	0.46 (0.03)	0.31 (0.04)	0.2 (0.05)	0.39 (0.01)	0.13 (0.03)	0.07 (0.01)
GRU-MLP/MC-TEMWL	0.4 (0.05)	0.23 (0.01)	0.44 (0.05)	0.22 (0.02)	0.18 (0.03)	0.44 (0.0)	**0.03 (0.0)**	0.17 (0.04)
GRU-RMTPP-TEMWL + B	0.31 (0.02)	**0.06 (0.01)**	0.34 (0.08)	0.19 (0.02)	**0.1 (0.01)**	0.4 (0.0)	0.13 (0.02)	0.07 (0.01)
GRU-SA/CM-LEWL	0.5 (0.0)	0.42 (0.02)	0.5 (0.0)	0.43 (0.01)	0.39 (0.03)	0.44 (0.0)	0.34 (0.03)	0.23 (0.04)
GRU-SA/MC-TEMWL	0.49 (0.01)	0.36 (0.02)	0.35 (0.06)	0.25 (0.01)	0.27 (0.02)	0.44 (0.0)	0.08 (0.04)	0.26 (0.03)
				F1-score				
	LastFM	MOOC	Wikipedia	Github	MIMIC2	Hawkes	SO	Retweets
GRU-EC-LE	0.0 (0.0)	0.06 (0.0)	0.02 (0.0)	0.4 (0.03)	0.22 (0.02)	0.14 (0.01)	0.29 (0.0)	0.53 (0.0)
GRU-LNM-TO	0.01 (0.0)	0.06 (0.01)	0.01 (0.0)	0.4 (0.03)	0.22 (0.02)	0.14 (0.01)	0.28 (0.01)	0.54 (0.01)
GRU-LN-LEWL	0.01 (0.0)	0.06 (0.01)	0.01 (0.0)	0.4 (0.03)	0.23 (0.02)	0.15 (0.01)	0.28 (0.0)	0.55 (0.0)
GRU-FNN-LCONCAT	0.02 (0.01)	0.23 (0.04)	0.02 (0.01)	0.49 (0.02)	0.22 (0.02)	0.23 (0.0)	**0.33 (0.0)**	**0.6 (0.0)**
GRU-MLP/MC-LCONCAT	0.02 (0.0)	0.24 (0.03)	0.03 (0.02)	0.4 (0.03)	0.43 (0.09)	0.2 (0.01)	0.32 (0.0)	**0.6 (0.0)**
GRU-RMTPP-LCONCAT	0.03 (0.01)	0.38 (0.01)	0.31 (0.11)	0.52 (0.01)	**0.74 (0.01)**	0.25 (0.01)	**0.33 (0.0)**	**0.6 (0.0)**
GRU-SA/CM-LE	0.0 (0.0)	0.04 (0.0)	0.0 (0.0)	0.4 (0.03)	0.22 (0.02)	0.13 (0.01)	0.28 (0.0)	0.45 (0.04)
GRU-SA/MC-LE	0.0 (0.0)	0.03 (0.0)	0.01 (0.0)	0.4 (0.03)	0.24 (0.03)	0.16 (0.01)	0.31 (0.0)	0.55 (0.0)
Hawkes	**0.3 (0.0)**	0.29 (0.0)	**0.66 (0.02)**	**0.54 (0.01)**	0.63 (0.0)	**0.28 (0.0)**	0.32 (0.0)	0.56 (0.0)
Poisson	0.0 (0.0)	0.01 (0.0)	0.01 (0.0)	0.4 (0.03)	0.2 (0.02)	0.11 (0.0)	0.26 (0.0)	0.33 (0.0)
NH	0.0 (0.0)	0.01 (0.0)	0.0 (0.0)	0.4 (0.03)	0.22 (0.02)	0.11 (0.0)	0.26 (0.0)	0.33 (0.0)
GRU-EC-TEMWL	0.02 (0.0)	0.28 (0.01)	0.01 (0.0)	0.41 (0.03)	0.53 (0.03)	0.27 (0.0)	0.32 (0.0)	0.58 (0.01)
GRU-LNM-CONCAT	0.1 (0.03)	0.36 (0.01)	0.22 (0.1)	0.47 (0.03)	0.64 (0.01)	0.26 (0.0)	**0.33 (0.0)**	0.59 (0.01)
GRU-LN-CONCAT	0.1 (0.02)	0.36 (0.01)	0.02 (0.01)	0.47 (0.03)	0.53 (0.09)	0.26 (0.0)	0.32 (0.0)	**0.6 (0.0)**
GRU-MLP/MC-TEMWL	0.02 (0.01)	0.25 (0.01)	0.07 (0.04)	0.42 (0.02)	0.59 (0.02)	0.18 (0.01)	**0.33 (0.0)**	0.57 (0.01)
GRU-RMTPP-TEMWL + B	0.11 (0.01)	**0.39 (0.0)**	0.46 (0.09)	0.48 (0.02)	0.73 (0.03)	0.27 (0.0)	0.32 (0.0)	0.59 (0.0)
GRU-SA/CM-LEWL	0.0 (0.0)	0.05 (0.01)	0.0 (0.0)	0.4 (0.03)	0.25 (0.04)	0.14 (0.0)	0.28 (0.01)	0.56 (0.01)
GRU-SA/MC-TEMWL	0.01 (0.0)	0.08 (0.02)	0.1 (0.04)	0.4 (0.03)	0.57 (0.02)	0.19 (0.02)	0.32 (0.0)	0.56 (0.0)

Table C.6: Results with respect to $\mathcal{L}_T$ and PCE for all models in Table C.3 on each unmarked dataset. Standard error across all splits is reported in parenthesis, and best results are highlighted in bold.

	$\mathcal{L}_T$							
	Taxi	Twitter	Reddit Subs	Reddit Ask	PUBG	Yelp T.	Yelp A.	Yelp M.
GRU-EC-TEM + B	-136.78 (3.91)	-6.21 (0.46)	-4403.97 (24.0)	-917.48 (13.1)	-106.99 (0.21)	-2803.23 (85.58)	-5.76 (0.18)	-56.18 (0.76)
GRU-LNM-TEM	**-137.02 (3.87)**	-11.89 (0.62)	**-4496.27 (37.01)**	**-926.36 (12.67)**	**-194.39 (18.07)**	**-2957.96 (92.81)**	-6.8 (0.19)	**-59.61 (0.9)**
GRU-FNN-LTO	-132.51 (4.01)	-11.66 (0.53)	-4424.06 (24.21)	-926.25 (12.87)	-121.88 (0.18)	-2486.32 (80.2)	-5.72 (0.16)	-58.52 (0.78)
GRU-MLP/MC-LTO	-136.25 (3.95)	-10.4 (0.32)	-4403.41 (24.25)	-921.58 (12.77)	-110.44 (1.21)	-2808.32 (84.78)	-5.75 (0.26)	-58.21 (0.79)
GRU-RMTPP-LTO	-136.15 (3.98)	-9.96 (0.53)	-4407.58 (24.0)	-919.08 (12.91)	-107.04 (0.2)	-2906.91 (87.45)	-5.85 (0.14)	-58.31 (0.79)
SA-SA/CM-LE	-134.2 (4.13)	-10.54 (0.51)	-4320.9 (23.17)	-856.8 (12.31)	-104.81 (1.31)	-2739.73 (81.7)	-5.29 (0.15)	-55.49 (0.88)
GRU-SA/MC-LE	-124.53 (5.1)	**-12.11 (0.5)**	-4387.88 (19.64)	-922.39 (13.07)	-109.3 (0.9)	-2901.79 (92.0)	**-6.83 (0.23)**	-57.35 (0.84)
Hawkes	-134.09 (3.93)	-7.62 (0.51)	-4395.2 (23.74)	-919.51 (12.92)	-102.69 (0.2)	-2773.63 (87.1)	-4.62 (0.18)	-53.33 (0.8)
Poisson	-41.71 (1.02)	5.18 (0.11)	-3706.3 (17.61)	-676.53 (10.2)	-93.11 (0.15)	-673.76 (17.22)	-2.11 (0.1)	-25.2 (0.29)
NH	-48.21 (24.44)	-1.94 (0.22)	-4158.41 (58.74)	-714.92 (11.7)	-94.95 (0.14)	-2084.84 (291.26)	-3.23 (0.14)	-39.19 (0.6)
	PCE							
	Taxi	Twitter	Reddit Subs	Reddit Ask	PUBG	Yelp T.	Yelp A.	Yelp M.
GRU-EC-TEM + B	**0.01 (0.0)**	0.13 (0.0)	0.01 (0.0)	0.02 (0.0)	0.02 (0.0)	0.05 (0.0)	0.02 (0.0)	0.05 (0.0)
GRU-LNM-TEM	**0.01 (0.0)**	**0.01 (0.0)**	**0.01 (0.0)**	**0.0 (0.0)**	0.01 (0.0)	**0.01 (0.0)**	**0.01 (0.0)**	**0.01 (0.0)**
GRU-FNN-LTO	0.02 (0.0)	**0.01 (0.0)**	**0.0 (0.0)**	**0.0 (0.0)**	**0.0 (0.0)**	0.05 (0.0)	**0.01 (0.0)**	**0.01 (0.0)**
GRU-MLP/MC-LTO	**0.01 (0.0)**	0.03 (0.01)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.05 (0.0)	**0.01 (0.0)**	0.02 (0.0)
GRU-RMTPP-LTO	**0.01 (0.0)**	0.03 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.03 (0.0)	**0.01 (0.0)**	**0.01 (0.0)**
SA-SA/CM-LE	**0.01 (0.0)**	**0.01 (0.0)**	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.04 (0.0)	**0.01 (0.0)**	0.02 (0.0)
GRU-SA/MC-LE	0.1 (0.03)	0.02 (0.0)	0.01 (0.0)	0.01 (0.0)	0.01 (0.0)	0.03 (0.0)	**0.01 (0.0)**	0.02 (0.0)
Hawkes	**0.01 (0.0)**	**0.01 (0.0)**	**0.0 (0.0)**	0.01 (0.0)	0.01 (0.0)	0.06 (0.0)	0.02 (0.0)	0.05 (0.0)
Poisson	0.35 (0.0)	0.2 (0.0)	0.27 (0.0)	0.25 (0.0)	0.07 (0.0)	0.47 (0.0)	0.12 (0.0)	0.27 (0.0)
NH	0.3 (0.06)	0.17 (0.0)	0.1 (0.02)	0.17 (0.0)	0.06 (0.0)	0.28 (0.06)	0.04 (0.0)	0.13 (0.0)

Table C.7: Combinations included in the experimental study. "Unmarked" refers to whether the method is adapted for unmarked datasets. For EC, LNM and RMTPP, the setting where a baseline intensity term (B) is also considered.

Decoder	Encoder	Event encoding	Name	Unmarked
EC	CONS	\	CONS-EC	✓
	GRU	TO	GRU-EC-TO	✓
		LTO	GRU-EC-LTO	✓
		CONCAT	GRU-EC-CONCAT	✗
		LCONCAT	GRU-EC-LCONCAT	✗
		TEM	GRU-EC-TEM	✓
		TEMWL	GRU-EC-TEMWL	✗
		LE	GRU-EC-LE	✓
		LEWL	GRU-EC-LEWL	✗
	SA	TO	SA-EC-TO	✓
		LTO	SA-EC-LTO	✓
		CONCAT	SA-EC-CONCAT	✗
		LCONCAT	SA-EC-LCONCAT	✗
		TEM	SA-EC-TEM	✓
		TEMWL	SA-EC-TEMWL	✗
		LE	SA-EC-LE	✓
		LEWL	SA-EC-LEWL	✗
LNM	CONS	\	CONS-LNM	✓
	GRU	TO	GRU-LNM-TO	✓
		LTO	GRU-LNM-LTO	✓
		CONCAT	GRU-LNM-CONCAT	✗
		LCONCAT	GRU-LNM-LCONCAT	✗
		TEM	GRU-LNM-TEM	✓
		TEMWL	GRU-LNM-TEMWL	✗
		LE	GRU-LNM-LE	✓
		LEWL	GRU-LNM-LEWL	✗
	SA	TO	SA-LNM-TO	✓
		LTO	SA-LNM-LTO	✓
		CONCAT	SA-LNM-CONCAT	✗
		LCONCAT	SA-LNM-LCONCAT	✗
		TEM	SA-LNM-TEM	✓
		TEMWL	SA-LNM-TEMWL	✗
		LE	SA-LNM-LE	✓
		LEWL	SA-LNM-LEWL	✗
RMTPP	CONS	\	CONS-RMTPP	✓
	GRU	TO	GRU-RMTPP-TO	✓
		LTO	GRU-RMTPP-LTO	✓
		CONCAT	GRU-RMTPP-CONCAT	✗
		LCONCAT	GRU-RMTPP-LCONCAT	✗
		TEM	GRU-RMTPP-TEM	✓
		TEMWL	GRU-RMTPP-TEMWL	✗
		LE	GRU-RMTPP-LE	✓
		LEWL	GRU-RMTPP-LEWL	✗
	SA	TO	SA-RMTPP-TO	✓
		LTO	SA-RMTPP-LTO	✓
		CONCAT	SA-RMTPP-CONCAT	✗
		LCONCAT	SA-RMTPP-LCONCAT	✗
		TEM	SA-RMTPP-TEM	✓
		TEMWL	SA-RMTPP-TEMWL	✗
		LE	SA-RMTPP-LE	✓
		LEWL	SA-RMTPP-LEWL	✗
FNN	CONS	TO	CONS-FNN-TO	✓
		LTO	CONS-FNN-LTO	✓
		CONCAT	CONS-FNN-CONCAT	✗
		LCONCAT	CONS-FNN-LCONCAT	✗
		LE	CONS-FNN-LE	✓
		LEWL	CONS-FNN-LEWL	✗
	GRU	TO	GRU-FNN-TO	✓
		LTO	GRU-FNN-LTO	✓
		CONCAT	GRU-FNN-CONCAT	✗
		LCONCAT	GRU-FNN-LCONCAT	✗
		LE	GRU-FNN-LE	✓
		LEWL	GRU-FNN-LEWL	✗
	SA	TO	SA-FNN-TO	✓
		LTO	SA-FNN-LTO	✓
		CONCAT	SA-FNN-CONCAT	✗
		LCONCAT	SA-FNN-LCONCAT	✗
		LE	SA-FNN-LE	✓
		LEWL	SA-FNN-LEWL	✗
MLP/MC	CONS	TO	CONS-MLP/MC-TO	✓
		LTO	CONS-MLP/MC-LTO	✓
		CONCAT	CONS-MLP/MC-CONCAT	✗
		LCONCAT	CONS-MLP/MC-LCONCAT	✗
		TEM	CONS-MLP/MC-TEM	✓
		TEMWL	CONS-MLP/MC-TEMWL	✗
		LE	CONS-MLP/MC-LE	✓
		LEWL	CONS-MLP/MC-LEWL	✗
	GRU	TO	GRU-MLP/MC-TO	✓
		LTO	GRU-MLP/MC-LTO	✓
		CONCAT	GRU-MLP/MC-CONCAT	✗
		LCONCAT	GRU-MLP/MC-LCONCAT	✗
		TEM	GRU-MLP/MC-TEM	✓
		TEMWL	GRU-MLP/MC-TEMWL	✗
		LE	GRU-MLP/MC-LE	✓
		LEWL	GRU-MLP/MC-LEWL	✗
	SA	TO	SA-MLP/MC-TO	✓
		LTO	SA-MLP/MC-LTO	✓
		CONCAT	SA-MLP/MC-CONCAT	✗
		LCONCAT	SA-MLP/MC-LCONCAT	✗
		TEM	SA-MLP/MC-TEM	✓
		TEMWL	SA-MLP/MC-TEMWL	✗
		LE	SA-MLP/MC-LE	✓
		LEWL	SA-MLP/MC-LEWL	✗
SA/CM	CONS	TO	CONS-SA/CM-TO	✓
		LTO	CONS-SA/CM-LTO	✓
		CONCAT	CONS-SA/CM-CONCAT	✗
		LCONCAT	CONS-SA/CM-LCONCAT	✗
		LE	CONS-SA/CM-LE	✓
		LEWL	CONS-SA/CM-LEWL	✗
	GRU	TO	GRU-SA/CM-TO	✓
		LTO	GRU-SA/CM-LTO	✓
		CONCAT	GRU-SA/CM-CONCAT	✗
		LCONCAT	GRU-SA/CM-LCONCAT	✗
		LE	GRU-SA/CM-LE	✓
		LEWL	GRU-SA/CM-LEWL	✗
	SA	TO	SA-SA/CM-TO	✓
		LTO	SA-SA/CM-LTO	✓
		CONCAT	SA-SA/CM-CONCAT	✗
		LCONCAT	SA-SA/CM-LCONCAT	✗
		LE	SA-SA/CM-LE	✓
		LEWL	SA-SA/CM-LEWL	✗
SA/MC	CONS	TO	CONS-SA/MC-TO	✓
		LTO	CONS-SA/MC-LTO	✓
		CONCAT	CONS-SA/MC-CONCAT	✗
		LCONCAT	CONS-SA/MC-LCONCAT	✗
		TEM	CONS-SA/MC-TEM	✓
		TEMWL	CONS-SA/MC-TEMWL	✗
		LE	CONS-SA/MC-LE	✓
		LEWL	CONS-SA/MC-LEWL	✗
	GRU	TO	GRU-SA/MC-TO	✓
		LTO	GRU-SA/MC-LTO	✓
		CONCAT	GRU-SA/MC-CONCAT	✗
		LCONCAT	GRU-SA/MC-LCONCAT	✗
		TEM	GRU-SA/MC-TEM	✓
		TEMWL	GRU-SA/MC-TEMWL	✗
		LE	GRU-SA/MC-LE	✓
		LEWL	GRU-SA/MC-LEWL	✗
	SA	TO	SA-SA/MC-TO	✓
		LTO	SA-SA/MC-LTO	✓
		CONCAT	SA-SA/MC-CONCAT	✗
		LCONCAT	SA-SA/MC-LCONCAT	✗
		TEM	SA-SA/MC-TEM	✓
		TEMWL	SA-SA/MC-TEMWL	✗
		LE	SA-SA/MC-LE	✓
		LEWL	SA-SA/MC-LEWL	✗
Neural Hawkes	\	\	NH	✓
Hawkes	\	\	Hawkes	✓
Poisson	\	\	Poisson	✓

C.4. Results for Relevant Datasets Only

The discussion in Section 3.4 highlighted that some datasets (MIMIC2, Stack Overflow, Taxi, Reddit Subs, Reddit Ask Comments, Yelp Toronto and Yelp Mississauga) might potentially be inappropriate for benchmarking neural TPP models. For completeness, we report in Tables C.8, C.9, C.10, C.11, C.12 and C.13 the results of the aggregation procedure discussed in Section 3.3.5 without these borderline adequate datasets included. We found no significance differences with respect to the conclusions of Section 3.4.

Table C.8: Mean and median scores, and average ranks of the best combinations per decoder on the $\mathcal{L}_T$ (top rows) and $\mathcal{L}_M$ (bottom row) across all relevant marked datasets. Best results are highlighted in bold.

	Marked Datasets														
	$\mathcal{L}_T$			PCE			$\mathcal{L}_M$			ECE			F1-score		
	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank
GRU-EC-LCONCAT	0.07	-0.01	8.5	0.22	0.23	9.17	-0.97	-1.34	4.5	0.29	0.28	4.33	0.27	0.27	4.0
GRU-LNM-CONCAT	**-0.86**	**-0.84**	**1.5**	**0.02**	**0.01**	**1.5**	-2.97	-1.76	**2.33**	0.26	0.27	3.33	0.34	0.31	2.67
GRU-LN-LTO	-0.41	-0.63	4.17	0.06	0.04	5.0	-0.26	-0.33	5.83	0.4	0.4	7.17	0.21	0.14	7.5
GRU-FNN-LTO	-0.62	-0.7	3.17	0.02	0.02	1.83	-0.09	-0.14	6.83	0.39	0.42	6.33	0.21	0.14	6.33
GRU-MLP/MC-LCONCAT	-0.41	-0.38	5.67	0.11	0.11	7.0	-0.58	-0.47	5.5	0.29	0.32	3.5	0.25	0.22	4.33
GRU-RMTPP-LCONCAT	-0.59	-0.5	4.67	0.05	0.04	4.67	-1.94	**-1.99**	2.67	0.25	0.26	2.67	0.35	0.34	2.17
GRU-SA/CM-LE	-0.33	-0.39	6.33	0.06	0.04	4.0	0.11	0.11	8.17	0.45	0.45	8.33	0.17	0.09	9.0
GRU-SA/MC-LE	-0.5	-0.46	4.17	0.09	0.09	5.5	0.1	-0.04	8.17	0.39	0.43	7.33	0.19	0.1	7.33
Hawkes	0.5	-0.15	7.17	0.13	0.17	6.67	**-6.27**	-1.58	3.17	**0.15**	**0.13**	**2.0**	**0.44**	**0.42**	**2**
Poisson	1.72	1.31	10.83	0.31	0.35	10.5	1.39	1.26	10.83	0.49	0.49	10.67	0.14	0.06	10.33
NH	1.53	1.07	9.83	0.3	0.35	10.17	0.16	0.41	8.0	0.48	0.49	10.33	0.14	0.06	10.33
GRU-EC-TEMWL	0.22	0.16	8.17	0.23	0.24	9.0	-1.5	-1.1	5.0	0.33	0.33	5.83	0.26	0.27	5.5
GRU-LNM-CONCAT	**-0.86**	**-0.84**	**1.17**	**0.02**	**0.01**	1.83	-2.97	-1.76	**2.83**	0.26	0.27	4.33	0.34	0.31	3.17
GRU-LN-CONCAT	-0.37	-0.57	4.33	0.08	0.04	4.5	-3.0	-1.81	3.5	0.28	0.3	3.83	0.3	0.31	4.0
GRU-FNN-LCONCAT	-0.62	-0.71	2.5	0.02	0.02	**1.33**	-1.98	-1.51	5.0	0.31	0.35	4.67	0.27	0.23	5.5
GRU-MLP/MC-TEMWL	0.16	0.24	7.83	0.22	0.25	8.17	-0.77	-0.64	7.17	0.35	0.39	6.17	0.27	0.26	6.33
GRU-RMTPP-LCONCAT + B	-0.54	-0.49	3.33	0.05	0.04	4.0	-3.28	**-2.0**	3.17	0.2	0.16	2.67	0.37	0.33	2.17
GRU-SA/CM-LEWL	-0.25	-0.23	5.83	0.1	0.09	5.17	-0.21	-0.17	8.0	0.42	0.44	8.5	0.19	0.1	9.0
GRU-SA/MC-TEMWL	-0.26	-0.3	5.67	0.14	0.15	6.0	-0.39	-0.46	7.67	0.36	0.36	6.83	0.22	0.15	7.17

Table C.9: Mean and median scores, as well as average ranks per decoder and variation of event encoding, for relevant marked datasets. Refer to Section 3.3.5 for details on the aggregation procedure. Best results are highlighted in bold.

	Marked Datasets														
	$\mathcal{L}_T$			PCE			$\mathcal{L}_M$			ECE			F1-score		
	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank
EC-TO	0.83	0.59	7.17	0.26	0.28	6.0	0.15	0.19	6.17	0.45	0.46	7.17	0.17	0.08	7.0
EC-LTO	0.82	0.6	7.33	0.26	0.27	6.5	0.12	0.15	5.67	0.45	0.46	6.67	0.17	0.08	7.5
EC-CONCAT	0.22	0.22	**2.5**	**0.22**	**0.23**	2.5	-0.39	-0.75	3.0	0.37	0.39	**1.67**	0.25	0.23	2.17
EC-LCONCAT	0.41	0.29	4.0	0.24	0.24	4.83	-0.38	-0.66	3.67	0.36	**0.35**	2.67	0.24	0.22	3.17
EC-TEM	0.24	0.2	4.0	0.23	0.24	4.5	0.14	0.13	6.0	0.44	0.45	6.17	0.19	0.09	5.67
EC-TEMWL	0.32	0.3	4.17	0.23	0.25	5.0	**-0.69**	**-0.84**	**1.83**	**0.35**	0.35	2.0	**0.27**	**0.24**	**1.33**
EC-LE	**0.2**	**0.16**	3.0	0.22	0.24	**2.0**	0.11	0.1	5.17	0.43	0.45	5.5	0.19	0.09	5.5
EC-LEWL	0.26	0.25	3.83	0.23	0.25	4.67	-0.15	-0.45	4.5	0.39	0.41	4.17	0.24	0.22	3.67
LNM-TO	-0.55	-0.57	6.5	**0.02**	0.02	5.0	0.03	0.11	6.83	0.45	0.46	6.83	0.17	0.09	7.5
LNM-LTO	-0.56	-0.59	6.17	0.02	0.02	5.5	0.0	0.06	6.5	0.45	0.46	7.17	0.17	0.1	6.67
LNM-CONCAT	**-0.8**	-0.8	**1.67**	0.02	0.02	4.0	**-2.48**	-1.47	**1.5**	**0.28**	**0.26**	2.0	**0.35**	**0.35**	**1.5**
LNM-LCONCAT	-0.79	**-0.83**	2.0	0.02	**0.01**	**2.33**	-1.99	-1.18	3.0	0.31	0.33	2.83	0.29	0.27	3.33
LNM-TEM	-0.74	-0.78	4.5	0.02	0.01	4.0	-0.0	0.05	6.5	0.44	0.45	6.0	0.19	0.1	5.83
LNM-TEMWL	-0.7	-0.72	4.83	0.03	0.02	4.0	-2.17	**-1.49**	2.17	0.3	0.32	**1.83**	0.32	0.29	2.0
LNM-LE	-0.71	-0.77	5.67	0.03	0.02	6.83	-0.03	-0.02	6.17	0.43	0.44	6.0	0.19	0.1	6.0
LNM-LEWL	-0.71	-0.75	4.67	0.02	0.02	4.33	-2.13	-1.18	3.33	0.32	0.31	3.33	0.29	0.26	3.17
FNN-TO	1.15	0.68	4.5	0.26	0.31	4.67	0.34	0.37	4.5	0.48	0.48	5.5	0.14	0.07	5.17
FNN-LTO	-0.43	-0.54	2.0	0.03	**0.02**	1.67	0.06	0.05	3.5	0.42	0.42	1.83	0.2	0.12	2.17
FNN-CONCAT	1.1	0.54	4.33	0.27	0.31	4.33	0.19	0.13	3.33	0.46	0.46	4.0	0.15	0.09	3.5
FNN-LCONCAT	**-0.59**	**-0.66**	**1.0**	**0.02**	0.02	**1.33**	**-1.23**	**-1.17**	**1.67**	**0.33**	**0.35**	**1.17**	**0.25**	**0.22**	**1.0**
FNN-LE	1.14	0.72	4.33	0.27	0.32	4.17	0.3	0.34	4.0	0.48	0.48	4.0	0.15	0.06	4.67
FNN-LEWL	1.19	0.65	4.83	0.27	0.32	4.83	0.24	0.34	4.0	0.47	0.47	4.5	0.15	0.08	4.5
MLP/MC-TO	0.23	0.22	6.67	0.19	0.25	5.83	0.34	0.33	6.33	0.43	0.43	6.67	0.18	0.09	7.33
MLP/MC-LTO	-0.19	-0.17	3.33	**0.11**	**0.1**	**1.67**	0.5	0.25	7.17	0.41	0.45	6.33	0.19	0.1	5.83
MLP/MC-CONCAT	0.03	0.06	4.83	0.2	0.22	5.83	-0.03	-0.18	3.17	0.37	0.39	3.0	0.24	0.22	3.33
MLP/MC-LCONCAT	**-0.33**	**-0.28**	**1.83**	0.11	0.1	1.67	-0.17	-0.42	3.17	**0.33**	**0.34**	2.67	0.24	0.21	2.83
MLP/MC-TEM	0.04	0.03	5.67	0.19	0.23	6.0	0.26	0.18	5.17	0.41	0.41	5.83	0.19	0.1	6.67
MLP/MC-TEMWL	0.24	0.32	7.33	0.22	0.25	7.83	**-0.6**	**-0.43**	**2.17**	0.35	0.38	**2.5**	**0.26**	**0.26**	**1.67**
MLP/MC-LE	-0.11	-0.13	3.17	0.16	0.19	3.5	0.38	0.22	5.67	0.4	0.42	6.33	0.19	0.1	5.83
MLP/MC-LEWL	-0.09	-0.13	3.17	0.17	0.2	3.67	-0.39	-0.33	3.17	0.34	0.35	2.67	0.24	0.2	2.5
RMTPP-TO	0.32	0.23	7.33	0.2	0.24	6.83	0.07	0.09	6.17	0.44	0.45	7.17	0.17	0.09	7.33
RMTPP-LTO	-0.28	-0.42	3.83	0.06	0.07	2.17	0.05	0.03	6.5	0.44	0.44	6.67	0.18	0.11	6.33
RMTPP-CONCAT	0.0	0.06	4.17	0.19	0.22	4.33	-1.66	-1.29	3.0	0.29	0.34	2.33	0.31	0.3	2.5
RMTPP-LCONCAT	**-0.52**	**-0.47**	**1.5**	**0.05**	**0.05**	**1.0**	-1.85	**-1.57**	3.0	**0.26**	**0.27**	**1.83**	0.34	0.32	**1.67**
RMTPP-TEM	0.03	0.04	5.33	0.19	0.22	5.83	0.06	0.05	5.67	0.43	0.44	6.5	0.19	0.11	6.33
RMTPP-TEMWL	0.08	0.12	4.83	0.19	0.22	5.83	**-2.28**	-1.42	**1.67**	0.27	0.29	2.17	**0.35**	**0.37**	1.83
RMTPP-LE	0.01	0.02	4.17	0.19	0.22	4.5	0.07	0.05	6.0	0.43	0.44	5.67	0.19	0.11	6.0
RMTPP-LEWL	0.05	0.12	4.83	0.19	0.23	5.5	-1.24	-1.03	4.0	0.32	0.38	3.67	0.28	0.28	4.0
SA/CM-TO	-0.06	-0.09	4.67	0.08	0.06	4.17	0.75	0.22	4.5	0.46	0.46	5.0	0.14	0.1	4.33
SA/CM-LTO	-0.21	-0.18	2.5	**0.05**	0.05	**2.33**	0.14	0.07	3.5	0.44	**0.44**	**2.67**	**0.19**	0.1	**2.5**
SA/CM-CONCAT	-0.12	-0.29	3.5	0.08	0.07	3.17	0.78	0.2	4.5	0.46	0.46	3.5	0.14	0.1	3.0
SA/CM-LCONCAT	-0.02	0.05	4.33	0.08	0.06	3.5	0.16	0.08	**2.33**	0.45	0.45	3.0	0.19	**0.11**	3.0
SA/CM-LE	**-0.3**	**-0.36**	**2.33**	0.06	**0.04**	2.83	0.1	0.1	3.17	0.45	0.45	4.17	0.18	0.08	4.5
SA/CM-LEWL	-0.19	-0.19	3.67	0.1	0.08	5.0	**-0.12**	**-0.11**	3.0	**0.42**	0.44	2.67	0.19	0.09	3.67
SA/MC-TO	1.4	1.0	7.5	0.29	0.34	6.67	0.22	0.31	5.5	0.48	0.49	7.0	0.14	0.06	7.33
SA/MC-LTO	1.24	0.82	5.67	0.28	0.32	6.0	0.25	0.4	6.0	0.47	0.48	6.33	0.15	0.06	6.17
SA/MC-CONCAT	0.63	0.85	6.67	0.24	0.31	6.33	0.03	0.11	5.0	0.44	0.49	5.83	0.18	0.06	5.83
SA/MC-LCONCAT	0.52	0.74	5.17	0.24	0.3	6.5	0.1	0.15	5.17	0.42	0.48	5.0	0.18	0.06	5.67
SA/MC-TEM	-0.35	-0.3	3.33	0.13	0.14	3.33	0.18	0.05	5.0	0.41	0.44	4.33	0.18	0.08	4.17
SA/MC-TEMWL	-0.25	-0.26	4.17	0.14	0.14	4.17	**-0.32**	**-0.27**	**2.17**	**0.37**	**0.38**	2.0	**0.22**	**0.14**	1.67
SA/MC-LE	**-0.48**	-0.42	**1.5**	**0.09**	**0.09**	**1.33**	0.12	-0.03	4.67	0.4	0.43	3.67	0.19	0.09	3.5
SA/MC-LEWL	-0.45	**-0.45**	2.0	0.1	0.1	1.67	-0.1	-0.13	2.5	0.37	0.42	**1.83**	0.2	0.1	**1.67**

Table C.10: Mean and median scores, as well as average ranks per decoder and variation of event encoding, for relevant marked datasets. Refer to Section 3.3.5 for details on the aggregation procedure. Best results are highlighted in bold.

	Marked Datasets														
	$\mathcal{L}_T$			PCE			$\mathcal{L}_M$			ECE			F1-score		
	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank	Mean	Median	Rank
CONS-EC	1.66	1.26	3.0	0.3	0.35	2.83	0.7	0.46	2.67	0.48	0.49	3.0	0.14	0.06	3.0
SA-EC	0.68	0.54	2.0	0.25	0.27	2.17	-0.02	-0.16	2.0	0.43	0.42	2.0	0.2	0.15	1.83
GRU-EC	**0.15**	**0.15**	**1.0**	**0.22**	**0.23**	**1.0**	**-0.25**	**-0.4**	**1.33**	**0.39**	**0.38**	**1.0**	**0.22**	**0.17**	**1.17**
CONS-LNM	-0.34	-0.5	2.83	**0.02**	0.02	2.33	0.65	0.39	3.0	0.48	0.49	3.0	0.14	0.06	3.0
SA-LNM	-0.6	-0.66	1.83	0.02	0.02	2.17	-0.8	-0.44	1.83	0.39	0.39	1.83	0.24	0.17	1.83
GRU-LNM	**-0.79**	**-0.8**	**1.33**	0.03	**0.01**	**1.5**	**-1.39**	**-0.89**	**1.17**	**0.35**	**0.37**	**1.17**	**0.26**	**0.2**	**1.17**
CONS-FNN	0.94	0.61	3.0	0.2	0.23	2.67	0.76	0.41	3.0	0.47	0.48	2.67	0.16	0.07	3.0
SA-FNN	0.68	0.38	1.83	0.19	0.22	2.0	0.15	0.16	2.0	0.45	**0.45**	1.83	0.17	**0.11**	1.83
GRU-FNN	**0.51**	**0.12**	**1.17**	**0.18**	**0.21**	**1.33**	**-0.18**	**-0.08**	**1.0**	**0.43**	0.45	**1.5**	**0.18**	0.1	**1.17**
CONS-MLP/MC	0.46	0.43	2.67	0.19	0.22	2.0	0.88	0.58	2.67	0.44	0.46	3.0	0.17	0.07	3.0
SA-MLP/MC	0.09	0.15	2.0	**0.17**	0.21	2.33	0.13	-0.04	2.0	0.39	0.4	2.0	0.21	0.15	1.83
GRU-MLP/MC	**-0.14**	**-0.12**	**1.33**	0.17	**0.19**	**1.67**	**-0.06**	**-0.23**	**1.33**	**0.36**	**0.37**	**1.0**	**0.22**	**0.17**	**1.17**
CONS-RMTPP	0.81	0.56	3.0	0.18	0.21	2.33	1.1	0.59	2.67	0.47	0.48	3.0	0.13	0.05	3.0
SA-RMTPP	0.1	0.1	1.83	0.17	0.19	2.17	-0.49	-0.41	2.17	0.38	0.39	1.83	0.24	0.18	1.83
GRU-RMTPP	**-0.18**	**-0.17**	**1.17**	**0.15**	**0.17**	**1.5**	**-1.2**	**-0.83**	**1.17**	**0.34**	**0.37**	**1.17**	**0.26**	**0.22**	**1.17**
CONS-SA/CM	-0.02	-0.01	2.33	0.09	0.1	2.17	0.75	0.47	2.67	**0.45**	**0.45**	2.0	**0.17**	0.09	**1.83**
SA-SA/CM	-0.14	**-0.16**	**1.67**	**0.07**	**0.06**	2.17	**0.29**	**0.1**	1.83	0.45	0.45	**1.83**	0.17	**0.1**	1.67
GRU-SA/CM	**-0.16**	-0.07	2.0	0.07	0.07	**1.67**	0.31	0.1	**1.5**	0.45	0.45	2.17	0.17	0.1	2.5
CONS-SA/MC	0.8	0.52	3.0	0.22	0.25	2.83	0.76	0.44	3.0	0.46	0.47	3.0	0.16	0.07	3.0
SA-SA/MC	0.29	0.23	1.67	**0.19**	**0.21**	1.83	0.1	0.13	1.83	**0.42**	0.46	2.0	**0.18**	**0.08**	**1.5**
GRU-SA/MC	**0.28**	**0.22**	**1.33**	0.19	0.21	**1.33**	**0.02**	**0.07**	**1.17**	0.42	**0.45**	**1.0**	0.18	0.08	**1.5**

Table C.11: Mean, median, worst scores, and average ranks of the best combinations per decoder on the $\mathcal{L}_T$ across all relevant unmarked datasets. Best results are highlighted in bold.

	Unmarked Datasets							
	$\mathcal{L}_T$				PCE			
	Mean	Median	Worst	Rank	Mean	Median	Worst	Rank
GRU-EC-TEM + B	-0.21	-0.28	0.16	7.0	0.06	0.02	0.13	8.33
GRU-LNM-TEM	**-3.7**	**-1.03**	**-0.83**	**1.67**	**0.01**	**0.01**	**0.01**	**1.67**
GRU-LN-LE	-0.25	-0.46	0.47	6.0	0.02	0.02	0.04	6.67
GRU-FNN-LTO	-1.03	-0.79	-0.49	4.0	**0.01**	**0.01**	**0.01**	2.33
GRU-MLP/MC-LTO	-0.57	-0.57	-0.5	4.67	0.02	**0.01**	0.03	5.67
GRU-RMTPP-TEM + B	-0.51	-0.54	-0.36	5.67	0.02	**0.01**	0.05	5.0
GRU-SA/CM-TO	-0.71	-0.56	0.73	5.67	0.02	**0.01**	0.03	6.0
GRU-SA/MC-LE	-0.81	-0.87	-0.51	2.33	**0.01**	**0.01**	0.02	5.0
Hawkes	0.05	0.06	0.16	8.0	**0.01**	**0.01**	0.02	4.33
Poisson	1.54	1.32	2.14	11.0	0.13	0.12	0.2	11.0
NH	0.87	0.9	0.96	10.0	0.09	0.06	0.17	10.0

Table C.12: Mean, median, and worst scores, as well as average ranks per decoder and variation of event encoding, for relevant unmarked datasets. Refer to Section 3.3.5 for details on the aggregation procedure. Best scores are highlighted in bold.

	Unmarked Datasets							
	$\mathcal{L}_T$				PCE			
	Mean	Median	Worst	Rank	Mean	Median	Worst	Rank
EC-TO	0.36	0.35	0.6	3.33	0.08	0.04	0.16	2.67
EC-LTO	0.39	0.34	0.64	3.67	0.08	0.04	0.16	4.0
EC-TEM	**-0.03**	**-0.13**	**0.31**	**1.0**	**0.06**	**0.03**	**0.14**	**1.0**
EC-LE	0.17	0.04	0.46	2.0	0.07	0.03	0.15	2.33
LNM-TO	-2.19	-0.73	-0.53	3.0	**0.01**	**0.01**	**0.01**	2.67
LNM-LTO	-2.18	-0.72	-0.49	3.33	0.01	0.01	0.01	2.33
LNM-TEM	**-2.64**	**-0.82**	**-0.76**	**1.33**	0.01	0.01	0.01	**1.67**
LNM-LE	-1.55	-0.82	-0.56	2.33	0.01	0.01	0.01	3.33
FNN-TO	0.54	0.51	0.62	2.33	0.08	0.04	0.15	2.33
FNN-LTO	**-0.78**	**-0.65**	**-0.25**	**1.0**	**0.01**	**0.01**	**0.01**	**1.0**
FNN-LE	0.51	0.52	0.54	2.67	0.08	0.05	0.15	2.67
MLP/MC-TO	-0.03	0.01	0.1	3.33	0.04	0.03	0.08	3.33
MLP/MC-LTO	-0.21	-0.17	0.06	2.33	**0.02**	0.03	**0.03**	**1.33**
MLP/MC-TEM	-0.12	-0.14	-0.04	2.33	0.04	0.03	0.09	3.0
MLP/MC-LE	**-0.31**	**-0.39**	**-0.12**	**2.0**	0.03	**0.02**	0.06	2.33
RMTPP-TO	-0.11	-0.16	0.11	3.33	0.04	0.02	0.08	3.33
RMTPP-LTO	-0.12	-0.15	0.2	3.0	**0.02**	0.02	**0.03**	2.33
RMTPP-TEM	**-0.33**	**-0.29**	**-0.25**	**1.33**	0.03	**0.01**	0.07	2.33
RMTPP-LE	-0.25	-0.24	-0.14	2.33	0.03	0.01	0.08	**2.0**
SA/CM-TO	**-0.34**	-0.5	0.72	2.0	0.02	0.02	0.03	2.33
SA/CM-LTO	0.67	**-0.55**	3.38	2.33	0.04	0.02	0.1	2.67
SA/CM-LE	-0.25	-0.3	**0.19**	**1.67**	**0.01**	**0.01**	**0.01**	**1.0**
SA/MC-TO	0.76	0.72	0.96	4.0	0.08	0.07	0.15	4.0
SA/MC-LTO	0.51	0.42	0.89	3.0	0.07	0.06	0.14	3.0
SA/MC-TEM	-0.57	-0.66	-0.39	2.0	**0.01**	**0.01**	**0.02**	**1.0**
SA/MC-LE	**-0.69**	**-0.8**	**-0.43**	**1.0**	0.01	0.01	0.02	2.0

Table C.13: Mean, median, and worst scores, as well as average ranks per decoder and variation of history encoder, for relevant unmarked datasets. Refer to Section 3.3.5 for details on the aggregation procedure. Best scores are highlighted in bold.

	Unmarked Datasets							
	$\mathcal{L}_T$				PCE			
	Mean	Median	Worst	Rank	Mean	Median	Worst	Rank
CONS-EC	1.29	1.14	2.14	3.0	0.1	0.07	0.2	3.0
SA-EC	0.61	0.55	0.75	2.0	0.09	0.05	0.17	2.0
GRU-EC	**-0.17**	**-0.27**	**0.25**	**1.0**	**0.06**	**0.02**	**0.14**	**1.0**
CONS-LNM	-1.83	-0.25	0.12	2.33	**0.01**	**0.01**	0.02	**1.67**
SA-LNM	-1.24	-0.64	-0.24	2.67	0.01	0.01	**0.01**	2.33
GRU-LNM	**-3.04**	**-0.93**	**-0.9**	**1.0**	0.01	0.01	0.01	2.0
CONS-FNN	0.61	0.37	1.26	2.67	0.06	**0.03**	0.13	2.0
SA-FNN	0.15	0.16	0.34	2.0	**0.05**	0.03	**0.1**	**1.33**
GRU-FNN	**0.03**	**0.14**	**0.17**	**1.33**	0.06	0.03	0.11	2.67
CONS-MLP/MC	0.51	0.58	0.74	2.67	0.04	0.03	0.07	2.67
SA-MLP/MC	0.1	0.23	0.33	2.33	0.04	0.03	**0.06**	2.0
GRU-MLP/MC	**-0.44**	**-0.41**	**-0.38**	**1.0**	**0.03**	**0.02**	0.06	**1.33**
CONS-RMTPP	0.45	0.6	0.79	2.67	**0.03**	0.02	**0.05**	**1.67**
SA-RMTPP	0.03	-0.04	0.29	2.33	0.03	0.02	0.07	2.67
GRU-RMTPP	**-0.43**	**-0.37**	**-0.33**	**1.0**	0.03	**0.01**	0.06	1.67
CONS-SA/CM	0.7	0.12	2.02	3.0	0.03	0.02	**0.04**	2.67
SA-SA/CM	**0.01**	-0.43	**0.99**	1.67	**0.02**	0.02	0.04	**1.33**
GRU-SA/CM	0.04	**-0.61**	1.54	**1.33**	0.03	**0.01**	0.06	2.0
CONS-SA/MC	0.54	0.73	0.83	3.0	**0.05**	0.04	0.09	2.67
SA-SA/MC	0.05	-0.06	0.27	2.0	0.05	**0.03**	**0.08**	**1.33**
GRU-SA/MC	**-0.05**	**-0.16**	**0.25**	**1.0**	0.05	0.04	0.08	2.0

C.5. Additional Visualizations

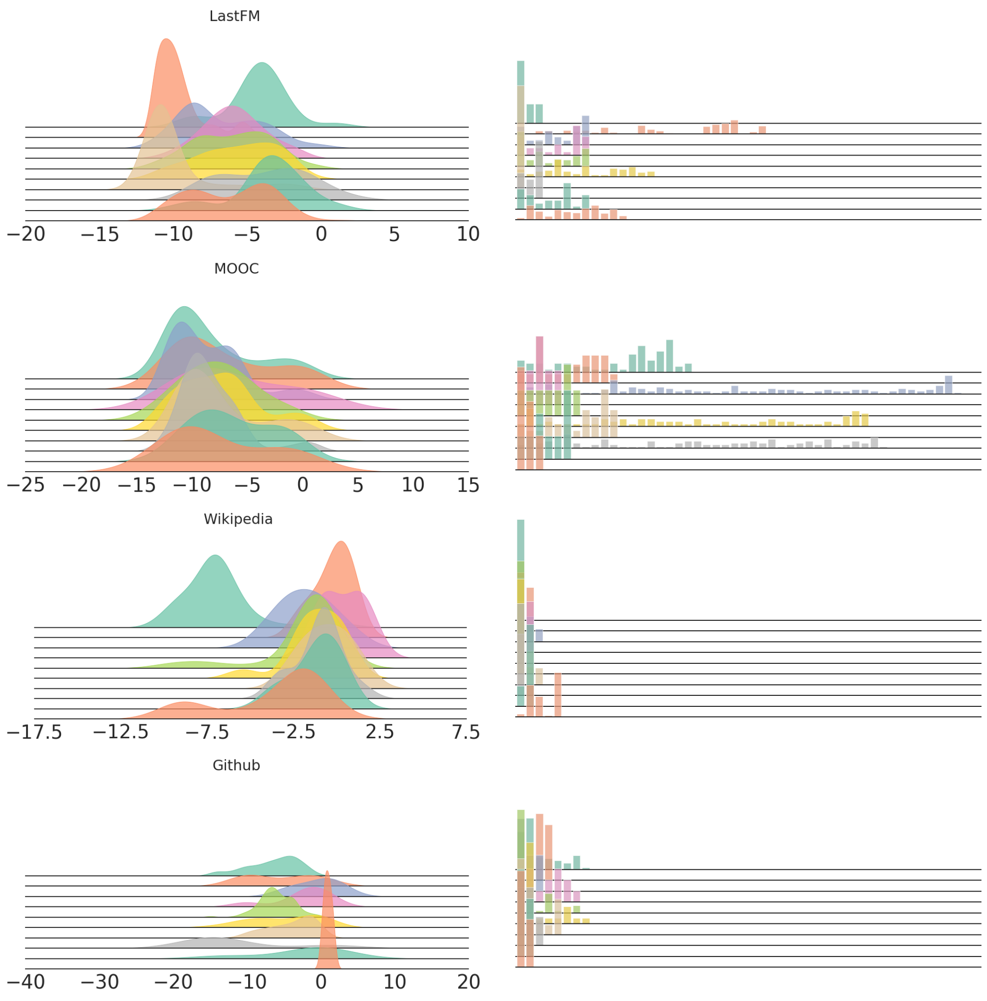


Figure C.3: Distributions of log τ (left) and marks (right) for 10 randomly sampled sequences in LastFM, MOOC, Wikipedia, and Github, after preprocessing.

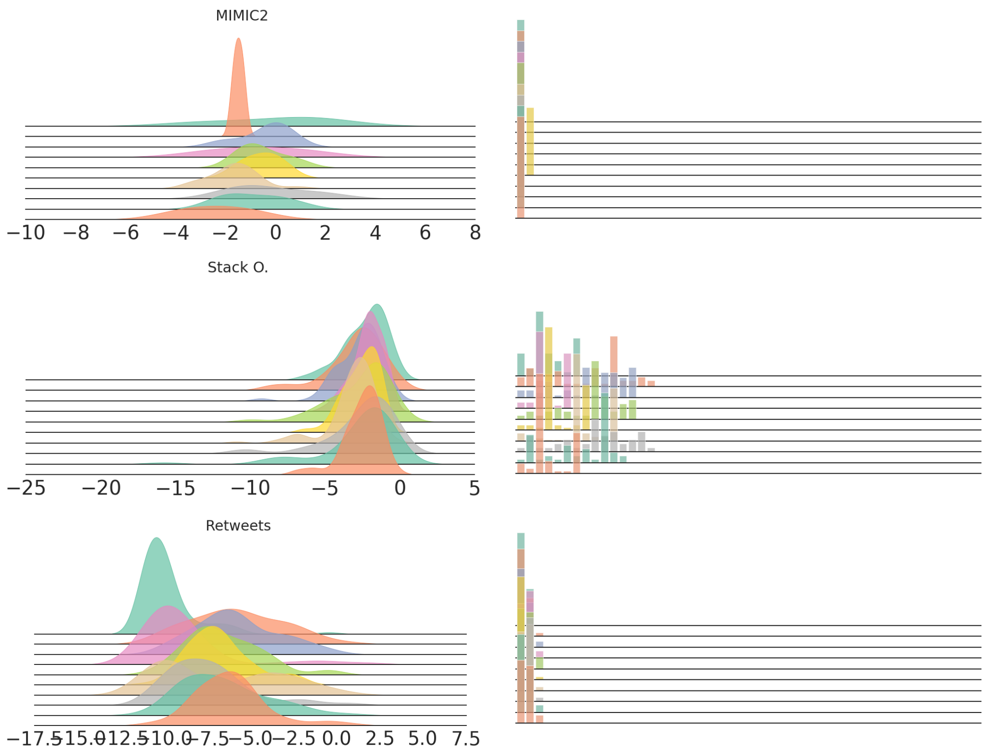


Figure C.4: Distributions of log τ (left) and marks (right) for 10 randomly sampled sequences in MIMIC2, Stack Overflow, and Retweets, after pre-processing.

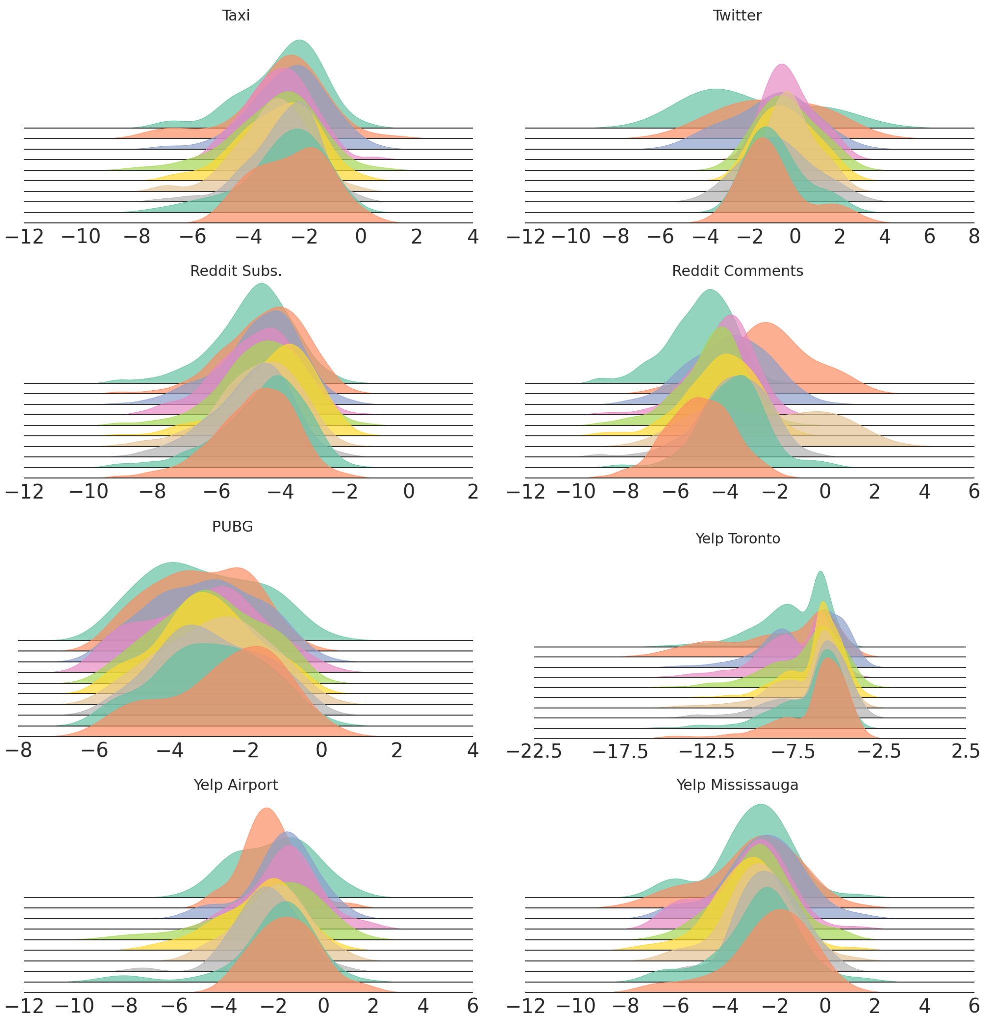


Figure C.5: Distribution of $\log \tau$ for 10 randomly sampled sequences in unmarked datasets, after pre-processing.

APPENDIX D

Supplementary Material for Chapter 4

D.1. Additional Forms of the Negative Log-Likelihood

Consider a dataset $\mathcal{S}_{\text{train}} = \{\mathcal{S}_1, ..., \mathcal{S}_L\}$, where each sequence $\mathcal{S}_l$ comprises n_l events with arrival times observed within the interval $[0, T]$ and $l = 1, ..., L$. The average sequence NLL for these sequences can be expressed as a function of the decompositions of $\lambda_k^*(t; \boldsymbol{\theta})$ and $\Lambda_k^*(t; \boldsymbol{\theta})$ as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log\,\lambda^*(t_{l,i}; \boldsymbol{\theta}) + \int_{t_n}^{T}\lambda^*(s; \boldsymbol{\theta})ds\right]}_{\mathcal{L}_T(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})} \\ & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_l}\log\,p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta})}_{\mathcal{L}_M(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})},\end{aligned} \tag{D.1}$$

and

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}; \mathcal{S}_{\text{train}}) = & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\left[\sum_{i=1}^{n_l}\log\,\left[\frac{d}{dt}\Lambda^*(t_{l,i}; \boldsymbol{\theta})\right] + \Lambda(T; \boldsymbol{\theta}))\right]}_{\mathcal{L}_T(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})} \\ & \underbrace{-\frac{1}{L}\sum_{l=1}^{L}\sum_{i=1}^{n_l}\log\,p^*(k_{l,i}|\tau_{l,i}; \boldsymbol{\theta})}_{\mathcal{L}_M(\boldsymbol{\theta}; \mathcal{S}_{\text{train}})},\end{aligned} \tag{D.2}$$

where $\lambda^*(t; \boldsymbol{\theta}) = \sum_{k=1}^{K}\lambda_k^*(t; \boldsymbol{\theta})$ and $\Lambda^*(t; \boldsymbol{\theta}) = \sum_{k=1}^{K}\Lambda_k^*(t; \boldsymbol{\theta})$.

D.2. Training Details

Encoding past events. To obtain the encoding $\boldsymbol{l}_i \in \mathbb{R}^{d_l}$ of an event $\boldsymbol{e}_i = (t_i, k_i)$ in $\mathcal{H}_t$, we follow the work of (Enguehard et al., 2020) by first mapping t_i to a vector of sinusoidal functions:

$$\boldsymbol{l}_i^t = \bigoplus_{j=0}^{d_t/2-1}\sin\,(\alpha_j t_i) \oplus \cos\,(\alpha_j t_i) \in \mathbb{R}^{d_t}, \tag{D.3}$$

where $\alpha_j \propto 1000^{\frac{-2j}{d_t}}$ and $\oplus$ is the concatenation operator. Then, a mark embedding $\boldsymbol{l}_i^k \in \mathbb{R}^{d_k}$ for k_i is generated as $\boldsymbol{l}_i^k = \mathbf{E}^k\boldsymbol{k}_i$, where $\mathbf{E}^k \in \mathbb{R}^{d_k \times K}$ is a learnable embedding matrix, and $\boldsymbol{k}_i \in \{0,1\}^K$ is the one-hot encoding of k_i. Finally, we obtain $\boldsymbol{l}_i$ through concatenation, i.e. $\boldsymbol{l}_i = [\boldsymbol{l}_i^t || \boldsymbol{l}_i^k]$.

Hyperparameters. To ensure that changes in performance are solely attributed to the features enabled by our framework, we control the number of parameters such the a baseline's capacity remains equivalent across the base, base+, base++, base-D, and base-DD setups. Notably, since STHP inherently models the decomposition of the marked intensities, the base and base+ configurations are equivalent. Also, LNM, RMTPP and STHP are equivalent between the base+ and the base-D settings, and between the base++ and base-DD settings. Hence, we only consider these models in the base+ and base++ settings. Furthermore, Table D.1 provides the total number of trainable parameters for each setup when trained on the LastFM dataset, as well as their distribution across the encoder and decoder heads. For all baselines and setups, we use a single encoder layer, and the dimension d_e of the event encodings is set to 8. Additionally, we chose a value of $M = 32$ for the number of mixture components. It is worth noting that (Shchur et al., 2020a) found LogNormMix to be robust to the choice of M. Finally, we set the number of GCIF projections to $C = 32$.

Table D.1: Number of parameters for each baseline when trained on the LastFM dataset. The distribution of parameters between the encoder and decoder heads is reported in parenthesis.

	THP	STHP	SAHP	FNN	LNM	RMTPP
Base	14720 (0.66/0.34)	\	15588 (0.68/0.32)	15939 (0.65/0.35)	13930 (0.67/0.33)	13619 (0.69/0.31)
Base+	14786 (0.64/0.36)	18586 (0.5/0.5)	15210 (0.67/0.33)	16083 (0.65/0.35)	13946 (0.67/0.33)	13669 (0.69/0.31)
Base++	14602 (0.63/0.37)	18402 (0.5/0.5)	15514 (0.67/0.33)	14961 (0.67/0.33)	13340 (0.66/0.34)	13063 (0.67/0.33)
Base-D	15512 (0.66/0.34)	\	15412 (0.67/0.33)	16083 (0.65/0.35)	\	\
Base-DD	15252 (0.66/0.34)	\	15464 (0.68/0.32)	10446 (0.65/0.35)	\	\

D.3. Additional Results

D.3.1 Evaluation Metrics

Tables D.2, D.3 and D.4 give the PCE, ECE, MRR, F1-score and Accuracy@{1,5} for the base, base+ and base++ setups across all datasets. The metrics are averaged over 3 splits, and the standard error is given in parenthesis. We note that these results are consistent with our conclusions in the main text. Specifically, we observe general improvement with respect to mark related metrics (i.e. ECE, MRR, accuracy@{1,5}) when moving from the base models to the base+ or base++ setups. Moreover, the PCE metric does not always improve between the base+ and base++ setups, suggesting that the remaining conflicting gradients at the encoder head in the base+ setup are mostly detrimental to the mark prediction task. Finally, mirroring our discussion in the main text, the Hawkes decoder remains the most competitive baseline on the Accuracy@{1,5}, F1-score and MRR metrics on LastFM.

Additionally, we report the results with respect to the MAE metric in Table D.5. We notice that lower MAE values do not systematically match the lower values of $\mathcal{L}_T$ or PCE in Tables 4.1 and D.2, indicating that the MAE may not be entirely appropriate to evaluate the time prediction task. As discussed in Shchur et al. (2021b), neural MTPP models are probabilistic models that enable the generation of complete distributions over future events. In this context, point prediction metrics, like MAE, are deemed less suitable for evaluating MTPP models because they consider single point predictions into account. In contrast, the NLL and calibration scores directly evaluate the entire predictive distributions, and should be therefore favored compared to point prediction metrics.

Table D.2: PCE and ECE results of the different setups across all datasets. The values are computed over 5 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

			PCE		
	LastFM	MOOC	Github	Reddit	Stack O.
THP	0.28 (0.0)	**0.37 (0.0)**	0.2 (0.01)	0.12 (0.0)	**0.01 (0.0)**
THP+	**0.27 (0.01)**	**0.37 (0.0)**	**0.19 (0.01)**	0.07 (0.0)	**0.01 (0.0)**
THP++	0.28 (0.01)	**0.37 (0.0)**	**0.19 (0.02)**	**0.05 (0.0)**	**0.01 (0.0)**
STHP+	**0.29 (0.01)**	0.37 (0.0)	0.29 (0.02)	**0.1 (0.0)**	**0.01 (0.0)**
STHP++	**0.29 (0.01)**	**0.36 (0.0)**	**0.25 (0.02)**	**0.1 (0.0)**	**0.01 (0.0)**
SAHP	0.06 (0.01)	0.12 (0.0)	0.07 (0.0)	0.1 (0.01)	**0.01 (0.0)**
SAHP+	0.05 (0.01)	**0.03 (0.0)**	**0.04 (0.01)**	**0.01 (0.0)**	**0.0 (0.0)**
SAHP++	**0.04 (0.01)**	**0.03 (0.0)**	**0.04 (0.01)**	0.01 (0.0)	0.04 (0.0)
LNM	**0.03 (0.01)**	**0.01 (0.0)**	**0.02 (0.0)**	**0.01 (0.0)**	**0.0 (0.0)**
LNM+	0.05 (0.01)	**0.01 (0.0)**	0.03 (0.0)	**0.01 (0.0)**	**0.0 (0.0)**
LNM++	**0.03 (0.0)**	**0.01 (0.0)**	0.03 (0.0)	**0.01 (0.0)**	**0.0 (0.0)**
FNN	0.04 (0.0)	0.09 (0.0)	0.04 (0.01)	0.07 (0.0)	0.05 (0.0)
FNN+	0.03 (0.0)	**0.01 (0.0)**	0.03 (0.01)	**0.01 (0.0)**	**0.01 (0.0)**
FNN++	**0.02 (0.0)**	**0.01 (0.0)**	**0.02 (0.0)**	**0.01 (0.0)**	**0.01 (0.0)**
RMTPP	**0.25 (0.01)**	0.29 (0.0)	0.18 (0.01)	**0.03 (0.0)**	**0.01 (0.0)**
RMTPP+	0.26 (0.01)	**0.27 (0.01)**	0.18 (0.01)	**0.03 (0.0)**	**0.01 (0.0)**
RMTPP++	0.26 (0.0)	0.29 (0.0)	**0.17 (0.01)**	0.04 (0.0)	**0.01 (0.0)**
Hawkes	0.24 (0.0)	0.25 (0.0)	0.15 (0.01)	0.09 (0.0)	0.01 (0.0)

			ECE		
	LastFM	MOOC	Github	Reddit	Stack O.
THP	0.23 (0.03)	0.07 (0.0)	0.09 (0.02)	**0.02 (0.0)**	0.04 (0.02)
THP+	0.05 (0.01)	**0.02 (0.0)**	**0.07 (0.01)**	0.03 (0.01)	**0.01 (0.0)**
THP++	**0.03 (0.0)**	**0.02 (0.0)**	**0.07 (0.02)**	**0.02 (0.0)**	**0.01 (0.0)**
STHP+	0.05 (0.0)	**0.03 (0.0)**	0.06 (0.02)	**0.03 (0.0)**	**0.02 (0.0)**
STHP++	**0.04 (0.0)**	**0.04 (0.0)**	0.04 (0.01)	0.05 (0.01)	0.03 (0.01)
SAHP	0.11 (0.01)	0.13 (0.0)	0.08 (0.02)	0.08 (0.01)	0.03 (0.0)
SAHP+	0.06 (0.01)	**0.02 (0.0)**	**0.07 (0.02)**	0.03 (0.01)	**0.01 (0.0)**
SAHP++	**0.03 (0.0)**	**0.02 (0.01)**	**0.07 (0.01)**	**0.02 (0.0)**	**0.01 (0.0)**
LNM	0.09 (0.02)	0.07 (0.01)	**0.05 (0.01)**	0.03 (0.01)	0.03 (0.01)
LNM+	0.05 (0.01)	0.04 (0.01)	0.06 (0.01)	**0.02 (0.0)**	**0.01 (0.0)**
LNM++	**0.02 (0.0)**	**0.02 (0.0)**	**0.05 (0.01)**	**0.02 (0.0)**	**0.01 (0.0)**
FNN	0.08 (0.01)	0.05 (0.0)	**0.05 (0.0)**	0.04 (0.01)	0.02 (0.0)
FNN+	0.04 (0.01)	**0.02 (0.0)**	0.06 (0.0)	**0.02 (0.0)**	**0.01 (0.0)**
FNN++	**0.03 (0.0)**	**0.02 (0.0)**	0.07 (0.02)	**0.02 (0.0)**	**0.01 (0.0)**
RMTPP	0.05 (0.01)	0.06 (0.01)	0.07 (0.0)	0.03 (0.0)	0.03 (0.02)
RMTPP+	0.05 (0.01)	0.03 (0.01)	**0.06 (0.0)**	0.03 (0.01)	**0.01 (0.0)**
RMTPP++	**0.04 (0.01)**	**0.02 (0.0)**	**0.06 (0.01)**	**0.02 (0.0)**	**0.01 (0.0)**
Hawkes	0.03 (0.0)	0.19 (0.01)	0.04 (0.0)	0.05 (0.0)	0.03 (0.01)

Table D.3: Accuracy@1 and Accuracy@5 results of the different setups across all datasets. The values are computed over 5 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

			Accuracy		
	LastFM	MOOC	Github	Reddit	Stack O.
THP	0.18 (0.01)	0.4 (0.0)	0.59 (0.02)	**0.83 (0.0)**	0.48 (0.0)
THP+	0.21 (0.01)	0.52 (0.0)	0.64 (0.01)	0.82 (0.0)	**0.49 (0.0)**
THP++	**0.25 (0.01)**	**0.55 (0.0)**	**0.67 (0.01)**	0.82 (0.0)	**0.49 (0.0)**
STHP+	0.2 (0.01)	**0.46 (0.0)**	0.63 (0.01)	**0.81 (0.0)**	**0.48 (0.0)**
STHP++	**0.21 (0.01)**	**0.46 (0.0)**	**0.65 (0.01)**	**0.81 (0.0)**	**0.48 (0.0)**
SAHP	0.05 (0.0)	0.36 (0.01)	0.6 (0.01)	0.69 (0.02)	0.48 (0.0)
SAHP+	0.14 (0.01)	0.54 (0.0)	0.65 (0.01)	**0.82 (0.0)**	**0.49 (0.0)**
SAHP++	**0.24 (0.01)**	**0.56 (0.0)**	**0.67 (0.01)**	**0.82 (0.0)**	**0.49 (0.0)**
LNM	0.21 (0.01)	0.43 (0.0)	0.64 (0.01)	0.81 (0.0)	0.47 (0.0)
LNM+	0.24 (0.01)	0.52 (0.0)	**0.67 (0.01)**	**0.82 (0.0)**	**0.49 (0.0)**
LNM++	**0.25 (0.01)**	**0.54 (0.0)**	**0.67 (0.01)**	**0.82 (0.0)**	0.48 (0.0)
FNN	0.13 (0.01)	0.51 (0.0)	**0.67 (0.01)**	0.8 (0.01)	0.48 (0.0)
FNN+	0.23 (0.01)	**0.55 (0.0)**	**0.67 (0.01)**	**0.82 (0.0)**	**0.49 (0.0)**
FNN++	**0.25 (0.01)**	**0.55 (0.0)**	**0.67 (0.01)**	**0.82 (0.0)**	**0.49 (0.0)**
RMTPP	0.23 (0.01)	0.42 (0.0)	0.61 (0.02)	**0.82 (0.0)**	0.47 (0.0)
RMTPP+	0.23 (0.01)	0.53 (0.0)	0.64 (0.01)	**0.82 (0.0)**	**0.49 (0.0)**
RMTPP++	**0.24 (0.01)**	**0.55 (0.0)**	**0.67 (0.01)**	**0.82 (0.0)**	**0.49 (0.0)**
Hawkes	0.31 (0.0)	0.3 (0.0)	0.64 (0.01)	0.82 (0.0)	0.47 (0.0)

			Accuracy@5		
	LastFM	MOOC	Github	Reddit	Stack O.
THP	0.46 (0.01)	0.87 (0.0)	0.95 (0.01)	**0.93 (0.0)**	**0.93 (0.0)**
THP+	0.47 (0.01)	0.89 (0.0)	0.96 (0.0)	0.92 (0.0)	**0.93 (0.0)**
THP++	**0.52 (0.01)**	**0.9 (0.0)**	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
STHP+	**0.46 (0.01)**	0.88 (0.0)	0.96 (0.0)	**0.92 (0.0)**	**0.93 (0.0)**
STHP++	**0.46 (0.01)**	**0.89 (0.0)**	**0.97 (0.0)**	0.91 (0.0)	0.92 (0.0)
SAHP	0.22 (0.01)	0.67 (0.01)	0.95 (0.0)	0.84 (0.02)	0.91 (0.0)
SAHP+	0.39 (0.01)	0.89 (0.0)	**0.97 (0.0)**	0.92 (0.0)	**0.93 (0.0)**
SAHP++	**0.52 (0.01)**	**0.9 (0.0)**	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
LNM	0.47 (0.02)	0.88 (0.0)	0.96 (0.0)	0.92 (0.0)	0.92 (0.0)
LNM+	0.51 (0.01)	0.89 (0.0)	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
LNM++	**0.54 (0.01)**	**0.9 (0.0)**	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
FNN	0.4 (0.01)	0.88 (0.0)	**0.97 (0.0)**	0.92 (0.0)	0.91 (0.0)
FNN+	0.5 (0.01)	**0.9 (0.0)**	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
FNN++	**0.53 (0.01)**	**0.9 (0.0)**	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
RMTPP	0.48 (0.01)	0.88 (0.0)	0.96 (0.0)	**0.93 (0.0)**	0.92 (0.0)
RMTPP+	0.49 (0.01)	0.89 (0.0)	0.96 (0.0)	**0.93 (0.0)**	**0.93 (0.0)**
RMTPP++	**0.52 (0.01)**	**0.9 (0.0)**	**0.97 (0.0)**	**0.93 (0.0)**	**0.93 (0.0)**
Hawkes	0.71 (0.01)	0.76 (0.0)	0.96 (0.0)	0.94 (0.0)	0.9 (0.0)

Table D.4: F1-score and MRR results of the different setups across all datasets. The values are computed over 5 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

			F1-score		
	LastFM	MOOC	Github	Reddit	Stack O.
THP	0.18 (0.0)	0.37 (0.0)	0.48 (0.05)	**0.81 (0.0)**	0.34 (0.0)
THP+	0.22 (0.01)	0.51 (0.0)	0.57 (0.01)	**0.81 (0.0)**	**0.35 (0.0)**
THP++	**0.25 (0.01)**	**0.54 (0.0)**	**0.61 (0.01)**	**0.81 (0.0)**	**0.35 (0.0)**
STHP+	**0.21 (0.01)**	**0.44 (0.0)**	0.58 (0.01)	**0.8 (0.0)**	**0.35 (0.0)**
STHP++	**0.21 (0.01)**	**0.44 (0.0)**	**0.59 (0.01)**	**0.8 (0.0)**	0.34 (0.0)
SAHP	0.02 (0.0)	0.29 (0.01)	0.54 (0.02)	0.67 (0.02)	0.34 (0.0)
SAHP+	0.11 (0.02)	0.53 (0.0)	0.57 (0.01)	**0.81 (0.0)**	**0.35 (0.0)**
SAHP++	**0.24 (0.0)**	**0.54 (0.0)**	**0.61 (0.01)**	**0.81 (0.0)**	**0.35 (0.0)**
LNM	0.2 (0.02)	0.41 (0.0)	0.57 (0.01)	0.8 (0.0)	0.33 (0.0)
LNM+	0.24 (0.0)	0.51 (0.0)	**0.61 (0.01)**	**0.81 (0.0)**	**0.35 (0.0)**
LNM++	**0.25 (0.01)**	**0.53 (0.0)**	**0.61 (0.01)**	**0.81 (0.0)**	**0.35 (0.0)**
FNN	0.12 (0.02)	0.49 (0.0)	0.6 (0.01)	0.79 (0.01)	0.33 (0.0)
FNN+	0.23 (0.01)	**0.54 (0.0)**	0.61 (0.01)	0.81 (0.0)	**0.35 (0.0)**
FNN++	**0.24 (0.01)**	**0.54 (0.0)**	**0.62 (0.01)**	**0.82 (0.0)**	**0.35 (0.0)**
RMTPP	0.23 (0.02)	0.4 (0.0)	0.51 (0.03)	**0.81 (0.0)**	0.33 (0.0)
RMTPP+	0.23 (0.01)	0.52 (0.0)	0.57 (0.0)	**0.81 (0.0)**	**0.35 (0.0)**
RMTPP++	**0.24 (0.01)**	**0.54 (0.0)**	**0.62 (0.01)**	**0.81 (0.0)**	**0.35 (0.0)**
Hawkes	0.3 (0.0)	0.25 (0.0)	0.55 (0.01)	0.81 (0.0)	0.32 (0.0)

			MRR		
	LastFM	MOOC	Github	Reddit	Stack O.
THP	0.32 (0.01)	0.6 (0.0)	0.74 (0.01)	**0.87 (0.0)**	**0.67 (0.0)**
THP+	0.34 (0.01)	0.68 (0.0)	0.77 (0.01)	**0.87 (0.0)**	**0.67 (0.0)**
THP++	**0.38 (0.01)**	**0.7 (0.0)**	**0.79 (0.01)**	**0.87 (0.0)**	**0.67 (0.0)**
STHP+	0.33 (0.01)	**0.64 (0.0)**	0.77 (0.01)	**0.86 (0.0)**	**0.67 (0.0)**
STHP++	**0.34 (0.01)**	**0.64 (0.0)**	**0.78 (0.0)**	**0.86 (0.0)**	**0.67 (0.0)**
SAHP	0.16 (0.0)	0.5 (0.01)	0.74 (0.01)	0.76 (0.02)	**0.67 (0.0)**
SAHP+	0.27 (0.01)	0.69 (0.0)	0.78 (0.01)	0.86 (0.0)	**0.67 (0.0)**
SAHP++	**0.38 (0.01)**	**0.7 (0.0)**	**0.79 (0.01)**	**0.87 (0.0)**	**0.67 (0.0)**
LNM	0.35 (0.01)	0.62 (0.0)	0.77 (0.0)	0.86 (0.0)	0.66 (0.0)
LNM+	0.37 (0.01)	0.68 (0.0)	**0.79 (0.0)**	**0.87 (0.0)**	**0.67 (0.0)**
LNM++	**0.4 (0.01)**	**0.69 (0.0)**	**0.79 (0.01)**	**0.87 (0.0)**	**0.67 (0.0)**
FNN	0.27 (0.01)	0.67 (0.0)	0.79 (0.01)	0.85 (0.0)	0.66 (0.0)
FNN+	0.36 (0.01)	**0.7 (0.0)**	0.79 (0.0)	**0.87 (0.0)**	**0.67 (0.0)**
FNN++	**0.38 (0.01)**	**0.7 (0.0)**	**0.8 (0.0)**	**0.87 (0.0)**	**0.67 (0.0)**
RMTPP	0.36 (0.01)	0.61 (0.0)	0.75 (0.01)	**0.87 (0.0)**	0.66 (0.0)
RMTPP+	0.36 (0.01)	0.69 (0.0)	0.77 (0.01)	**0.87 (0.0)**	**0.67 (0.0)**
RMTPP++	**0.38 (0.01)**	**0.7 (0.0)**	**0.79 (0.01)**	**0.87 (0.0)**	**0.67 (0.0)**
Hawkes	0.49 (0.0)	0.5 (0.0)	0.77 (0.01)	0.87 (0.0)	0.65 (0.0)

Table D.5: MAE results of the different setups across all datasets. The values are computed over 5 splits, and the standard error is reported in parenthesis. Best results are highlighted in bold.

	MAE				
	LastFM	MOOC	Github	Reddit	Stack O.
THP	**0.01624 (0.00056)**	**0.13332 (0.00096)**	0.0935 (0.01328)	**0.15257 (0.00322)**	**0.09885 (0.00097)**
THP+	0.01925 (0.00165)	0.13638 (0.0016)	0.09769 (0.014)	0.15811 (0.00079)	0.09972 (0.00077)
THP++	0.01744 (0.00103)	0.13365 (0.00134)	**0.09327 (0.01162)**	0.17627 (0.00359)	0.09926 (0.00069)
STHP+	**0.03717 (0.00978)**	0.17422 (0.01899)	0.11691 (0.00889)	**0.15864 (0.00409)**	**0.09883 (0.0007)**
STHP++	0.08356 (0.01887)	**0.17057 (0.02083)**	**0.1029 (0.01195)**	0.22542 (0.02801)	0.09931 (0.00056)
SAHP	0.0137 (0.00055)	0.12319 (0.00174)	0.08409 (0.01092)	0.18839 (0.00389)	0.0992 (0.00066)
SAHP+	0.01364 (0.00056)	0.12318 (0.00191)	0.07491 (0.0117)	0.13427 (0.00216)	**0.09853 (0.00071)**
SAHP++	**0.01358 (0.0006)**	**0.12138 (0.00197)**	**0.07127 (0.0102)**	**0.13168 (0.00206)**	0.10285 (0.00057)
FNN	0.03563 (0.0138)	0.16602 (0.0487)	0.41847 (0.19173)	0.23207 (0.05185)	0.266 (0.09399)
FNN+	0.01055 (0.00043)	0.07732 (0.00034)	**0.07093 (0.00902)**	0.13157 (0.00208)	**0.09299 (0.00072)**
FNN++	**0.01054 (0.00042)**	**0.07711 (0.00039)**	0.07102 (0.0093)	**0.13148 (0.00231)**	0.09305 (0.00064)
LNM	0.0111 (0.00049)	0.13516 (0.00078)	0.08706 (0.01153)	**0.13494 (0.00202)**	0.09494 (0.00075)
LNM+	0.01131 (0.00054)	0.13548 (0.0011)	0.08715 (0.0114)	0.135 (0.00213)	0.09517 (0.00069)
LNM++	**0.01098 (0.00049)**	**0.13083 (0.00096)**	**0.08694 (0.01135)**	0.1361 (0.00209)	**0.09476 (0.00071)**
RMTPP	**0.01619 (0.00097)**	0.1208 (0.00084)	0.12939 (0.02772)	**0.13577 (0.00223)**	**0.09522 (0.00069)**
RMTPP+	0.01677 (0.00072)	0.12059 (0.00083)	0.11416 (0.02624)	0.1362 (0.00227)	0.09563 (0.00085)
RMTPP++	0.01705 (0.00058)	**0.12011 (0.00079)**	**0.09183 (0.0106)**	0.13772 (0.00192)	0.09656 (0.0008)
Hawkes	0.0116 (0.00083)	0.13249 (0.00122)	0.08946 (0.01947)	0.14139 (0.00338)	0.09672 (0.00107)

D.3.2 Reliability Diagrams

Figures D.1 to D.3 show the reliability diagrams of the time and mark predictive distribution for all models in the base and base++ setups on all datasets. In most cases, we observe improved mark calibration for the base++ setup compared to the base models, in accordance to the ECE results of Table D.2. Additionally, improvements with respect to the calibration of the time predictive distribution is in general less prevalent, corroborating our discussion in the main text. This observation is also in coherent with the PCE results of Table D.2. Nonetheless, we observe substantial time calibration improvements for SAHP and FNN when trained in the base++ setup on MOOC and Reddit.

Figure D.4 shows the reliability diagrams for the time and mark predictive distributions of the Hawkes model for all datasets. We observe that the Hawkes model either shows competitive or worse mark calibration compared to neural MTPP models trained in the base++ setup. Note that this observation contrasts with our findings of Section 3.3, in which we found the Hawkes model to have the overall best mark calibration. Specifically, the ECE values of Table C.5 indicate that the Hawkes model outperforms neural MTPP baselines on LastFM and Github. Nonetheless, the mark calibration of the Hawkes model is now matched on these datasets by training neural MTPP baselines in the base++ setup, as demonstrated by the ECE values of Table D.2 and the reliability diagrams of Figures D.1 and D.2. Finally, comparing Figures D.1 to D.3 and Figure D.4, we observe that the time predictive distributions of neural MTPP models trained in the base++ setup are in general better calibrated than the ones returned by the Hawkes model.

(a) LastFM

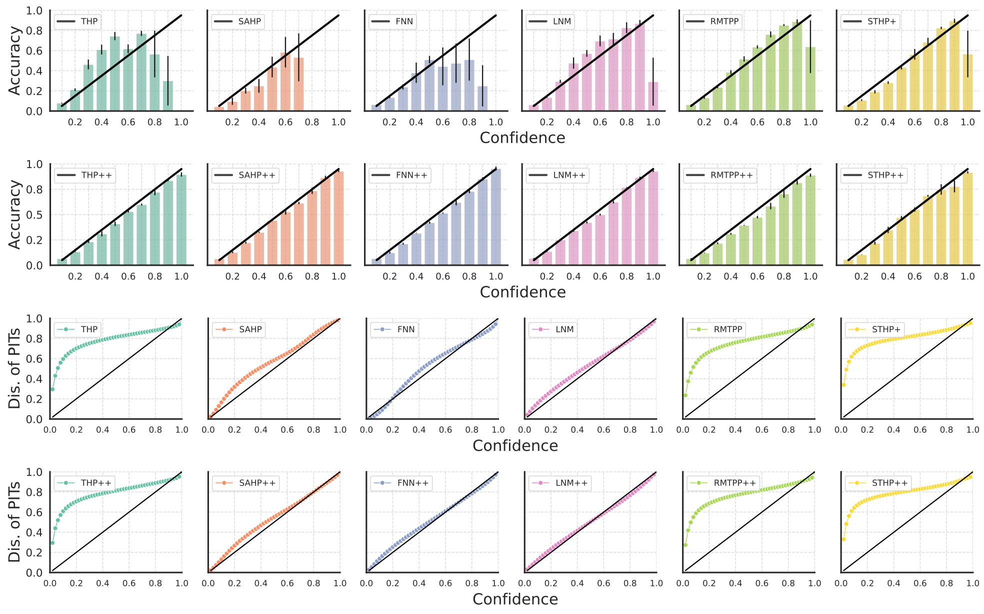
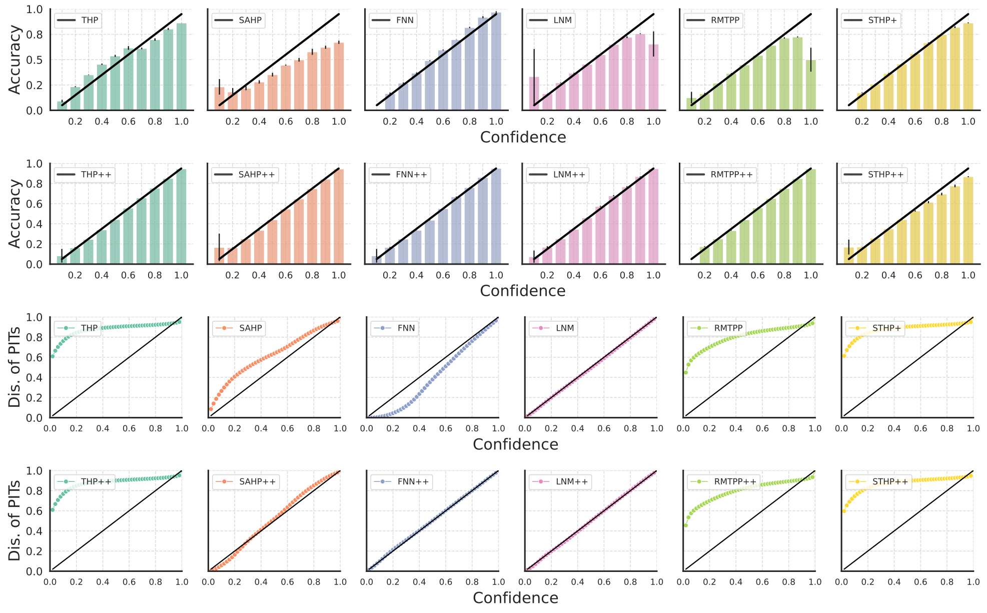


Figure D.1: Reliability diagrams of the mark (top) and time (bottom) predictive distributions on LastFM. Accuracy and distribution of PITs (Dis. of PITs) aligning with the black diagonal corresponds to perfect calibration. The results are averaged over 5 splits, and the error bars correspond to the standard error.

(a) MOOC



(b) Github

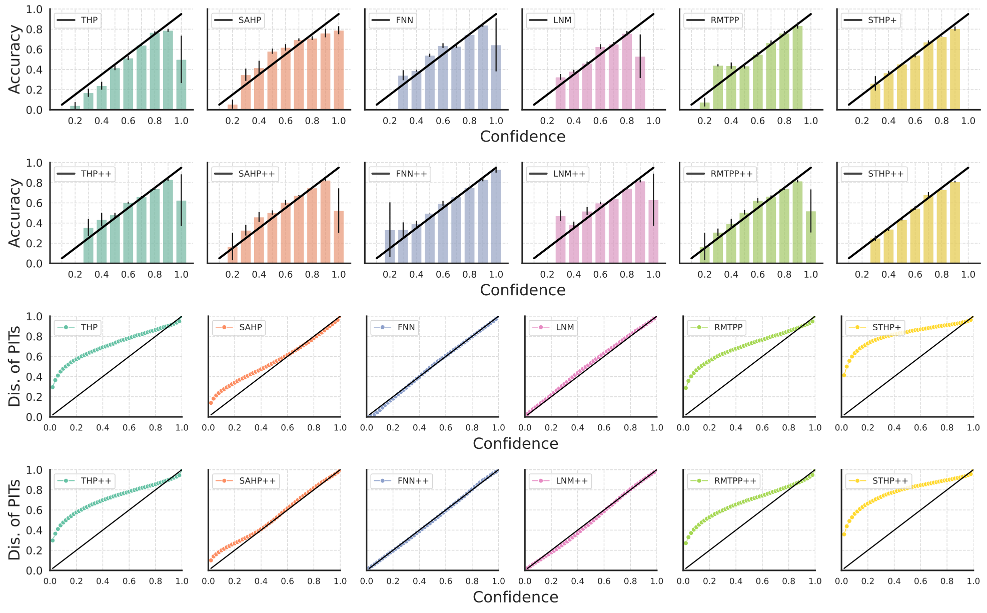
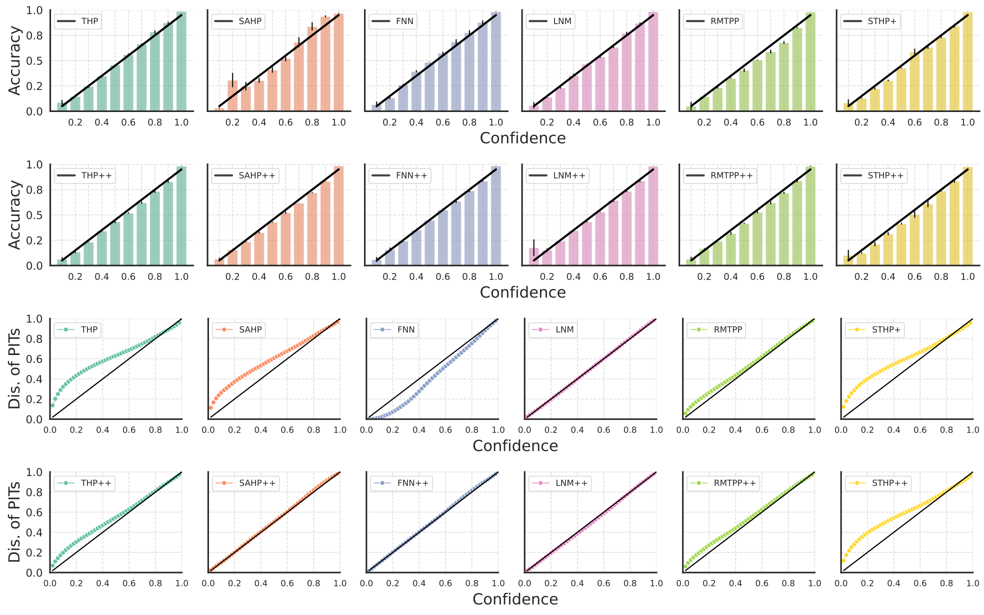


Figure D.2: Reliability diagrams of mark (top) and time (bottom) predictive distributions on MOOC and Github. Accuracy and distribution of PITs (Dis. of PITs) aligning with the black diagonal corresponds to perfect calibration. The results are averaged over 5 splits, and the error bars correspond to the standard error.

(a) Reddit



(b) Stack Overflow

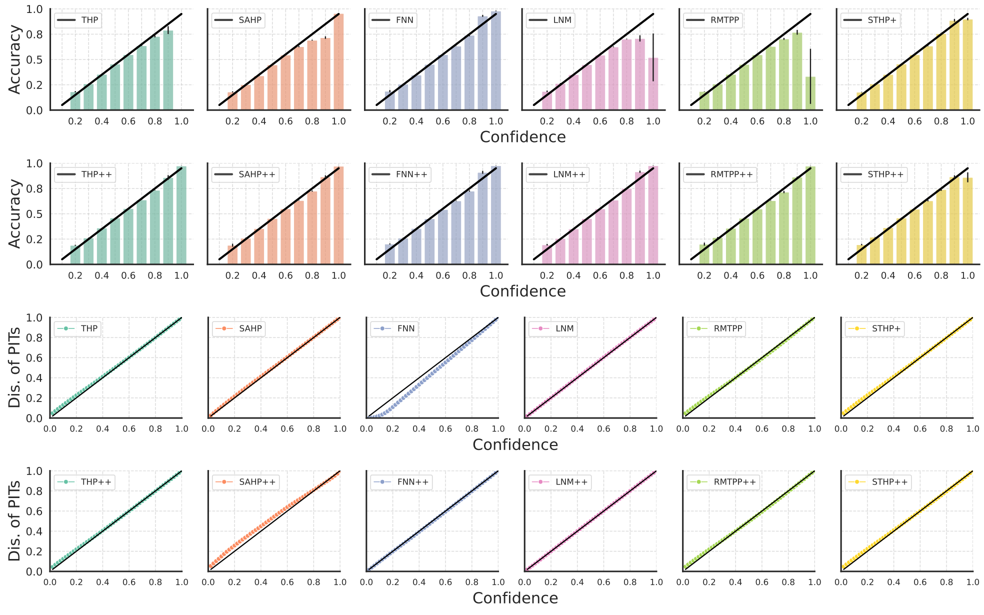


Figure D.3: Reliability diagrams of the mark (top) and time (bottom) predictive distributions on Reddit and Stack Overflow. Accuracy and distribution of PITs (Dis. of PITs) aligning with the black diagonal corresponds to perfect calibration. The results are averaged over 5 splits, and the error bars correspond to the standard error.

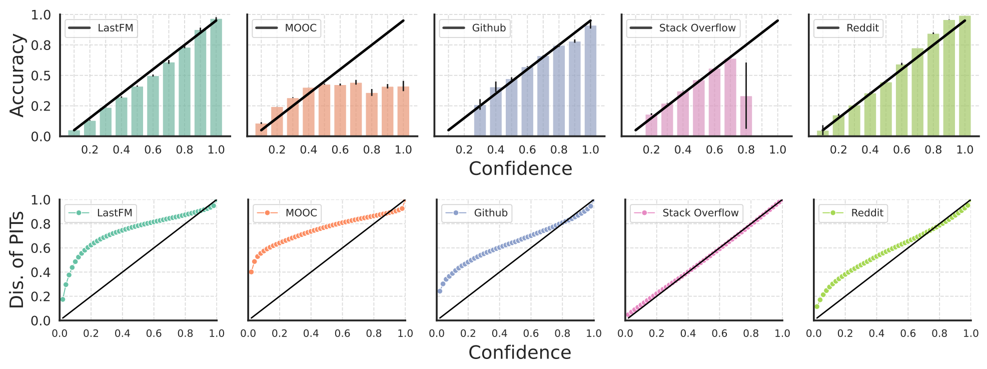


Figure D.4: Reliability diagrams of the mark (top) and time (bottom) predictive distributions for the Hawkes model on all datasets. Accuracy and distribution of PITs (Dis. of PITs) aligning with the black diagonal corresponds to perfect calibration. The results are averaged over 5 splits, and the error bars correspond to the standard error.

D.3.3 Distributions of Conflicting Gradients Between the Base and Base+ Settings

Figures D.5 to D.6 show the distributions of $\cos\phi_{TM}$ during training, as well as the proportion of conflicting gradients (CG), their average GMS, and their average TPI for all models in the base and base+ setups on all datasets. As discussed in the main text, severe conflicts ($\cos\phi_{TM} < -0.5$) often arise when training neural MTPP models in the base setup. For THP, SAHP, and FNN, the base+ setup reduces the occurrence of severe conflicts at the encoder heads, and prevents conflicting gradients to appear at the decoder heads. For LNM and RMTPP, we only show the distribution of $\cos\phi_{TM}$ at the encoder heads as the decoders are by design free of conflicting gradients in the base and base+ setups. For these models, the base+ setup does not change significantly the distribution of conflicting gradients during training, which aligns with expectations. Hence, we attribute performance gains on the mark prediction task to the alleviation of the conditional independence assumption of arrival times and marks through (4.26).

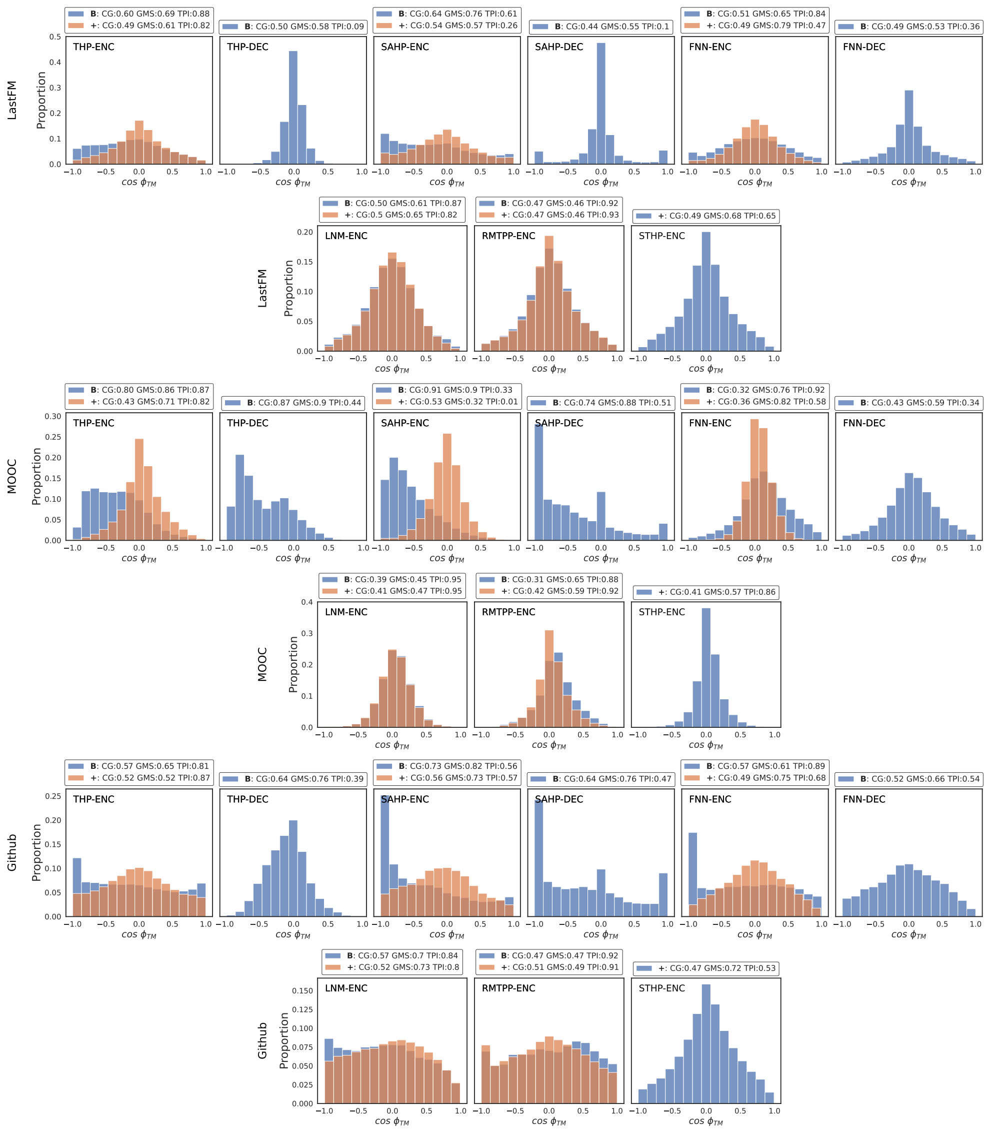


Figure D.5: Distribution of cos ϕ_{TM} during training at the encoder (ENC) and decoder (DEC) heads for all baselines in the base and base+ setup on LastFM, MOOC and Github. "B" and "+" refer to the base and base+ models, respectively, and the distribution is obtained by pooling the values of ϕ_{TM} over 5 training runs. As the decoders are disjoint in the base+ setting, note that cos ϕ_{TM} is not defined.

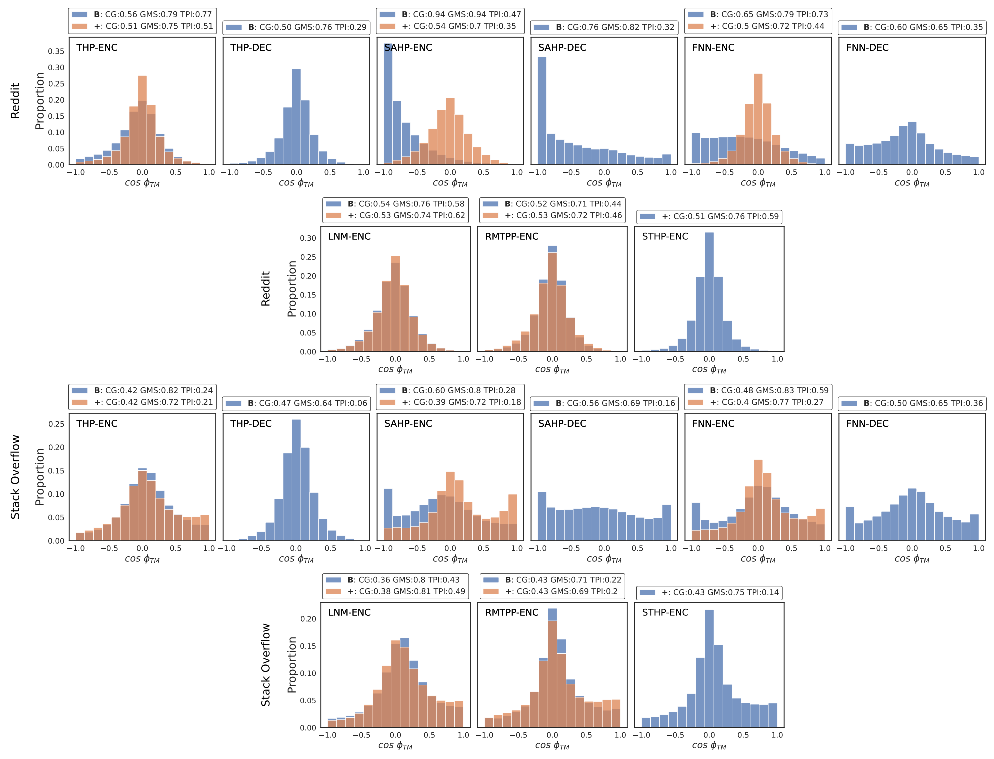


Figure D.6: Distribution of cos ϕ_{TM} during training at the encoder (ENC) and decoder (DEC) heads for all baselines in the base and base+ setup on Reddit and Stack Overflow. "B" and "+" refer to the base and base+ models, respectively, and the distribution is obtained by pooling the values of ϕ_{TM} over 5 training runs. As the decoders are disjoint in the base+ setting, note that cos ϕ_{TM} is not defined.

D.3.4 Distributions of Conflicting Gradients in the Base and Base-D Settings

Figure D.7 shows the distribution of cos ϕ_{TM} during training, as well as the proportion of conflicting gradients (CG), their average GMS, and their average TPI for THP, SAHP, and FNN in the base and base-D setups on all datasets. We observe that using two identical and task-specific instances of the same decoder in the base-D setup for the time and mark prediction tasks mitigates the occurrence of conflicts. The decrease in conflicting gradients during training is in turn associated to increased performance with respect to both tasks. We refer the reader to the discussion in Section 4.4.2 of the main text for further details.

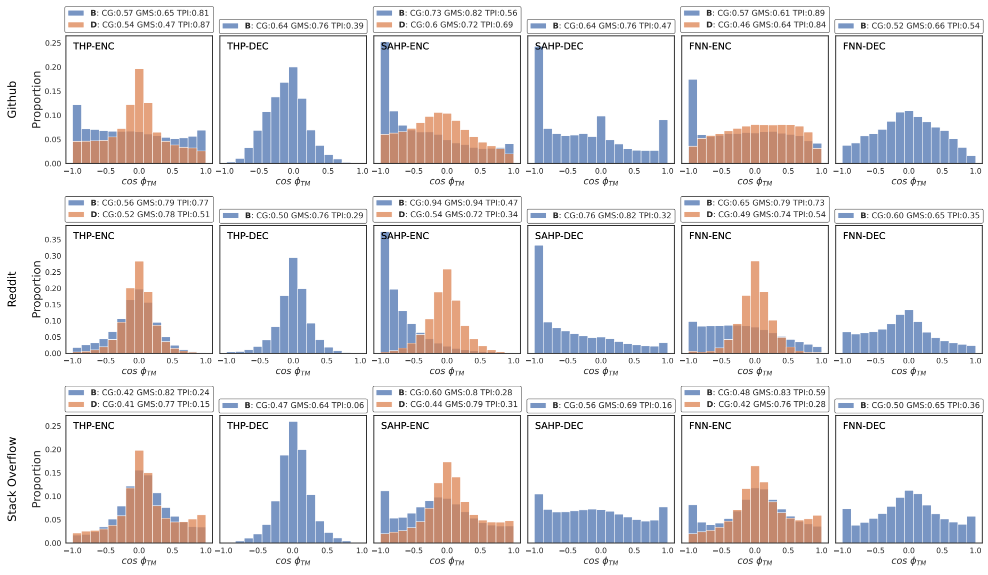


Figure D.7: Distribution of cos ϕ_{TM} during training at the encoder (ENC) and decoder (DEC) heads for all baselines in the base and base-D setup on Github, Reddit and Stack Overflow. "B" and "D" refer to the base and base-D models, respectively, and the distribution is obtained by pooling the values of ϕ_{TM} over 5 training runs. As the decoders are disjoint in the base-D setting, note that cos ϕ_{TM} is not defined.

APPENDIX E

Supplementary Material for Chapter 5

E.1. Results on Other Models

In Section 5.5.5, we provided results for the LNM+ model. In Sections E.1.1 to E.1.3, we present additional findings for the FNN, RMTPP, and SAHP models, respectively, on the datasets discussed in the main text. Ideally, the MC and WSC metrics should align with the nominal coverage $1 - \alpha = 0.8$, while the relative length (R. Length), geometric length (G. Length), and CCE metrics should be minimized. Across all these models, our conclusions align with those outlined in Section 5.5.5, and are applicable to all scenarios considered, i.e. prediction regions for the arrival times, marks, or both.

E.1.1 FNN

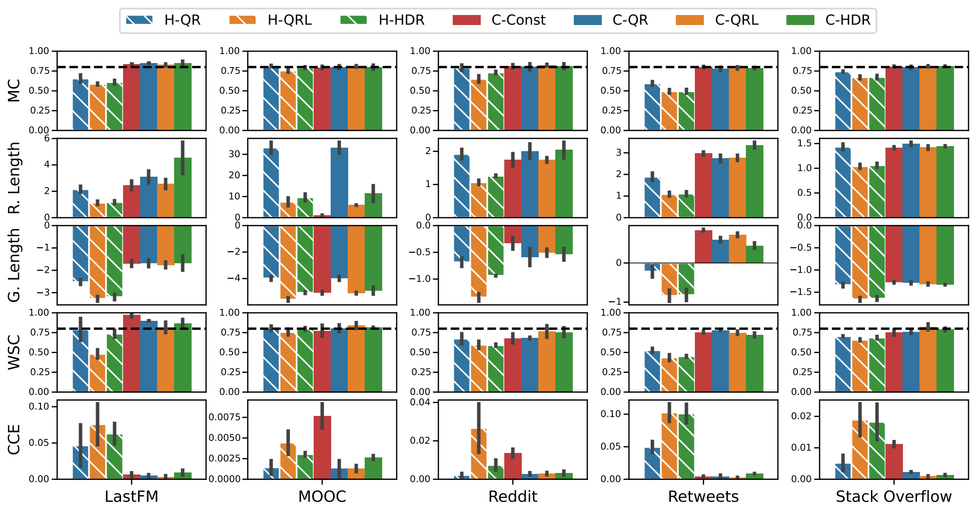


Figure E.1: Performance of different methods producing a region for the time on real world datasets using the FNN model.

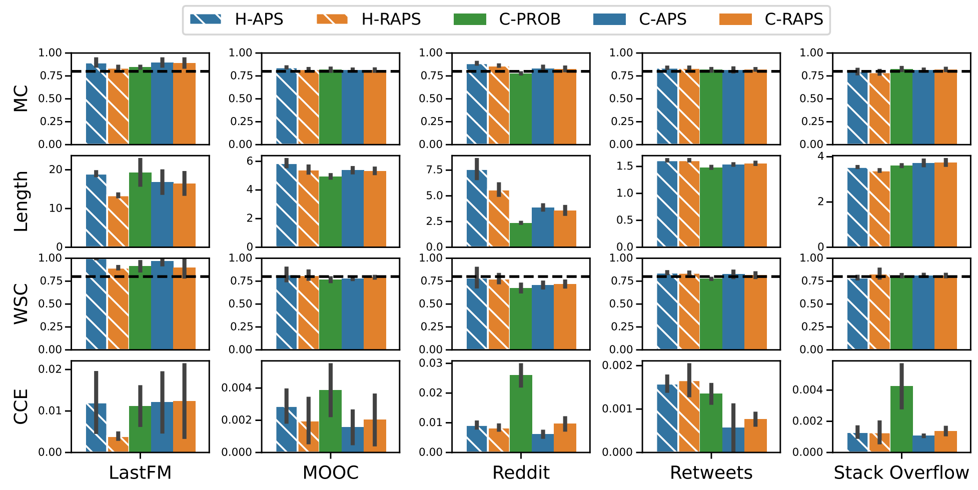


Figure E.2: Performance of different methods producing a region for the mark on real world datasets using the FNN model.

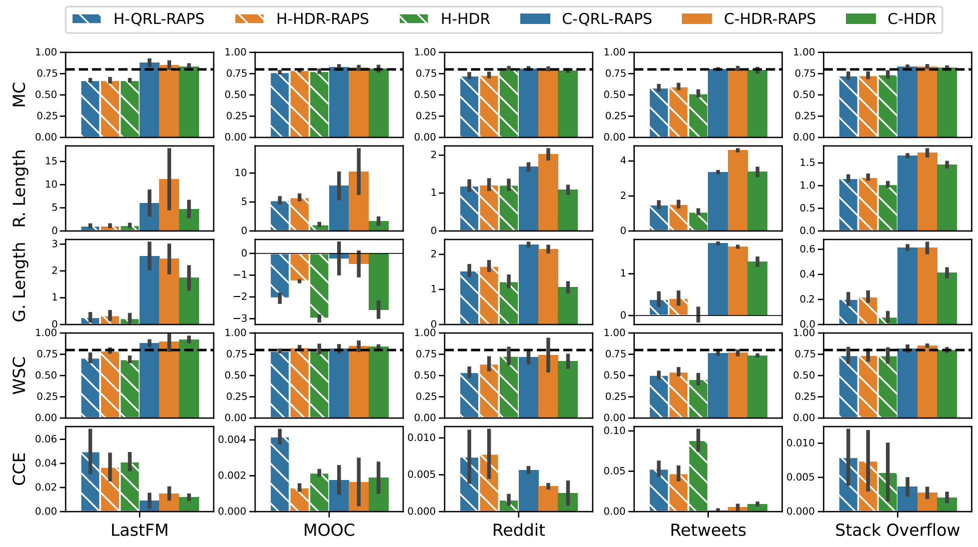


Figure E.3: Performance of different methods producing a joint region for the time and mark on real world datasets using the FNN model.

E.1.2 RMTPP

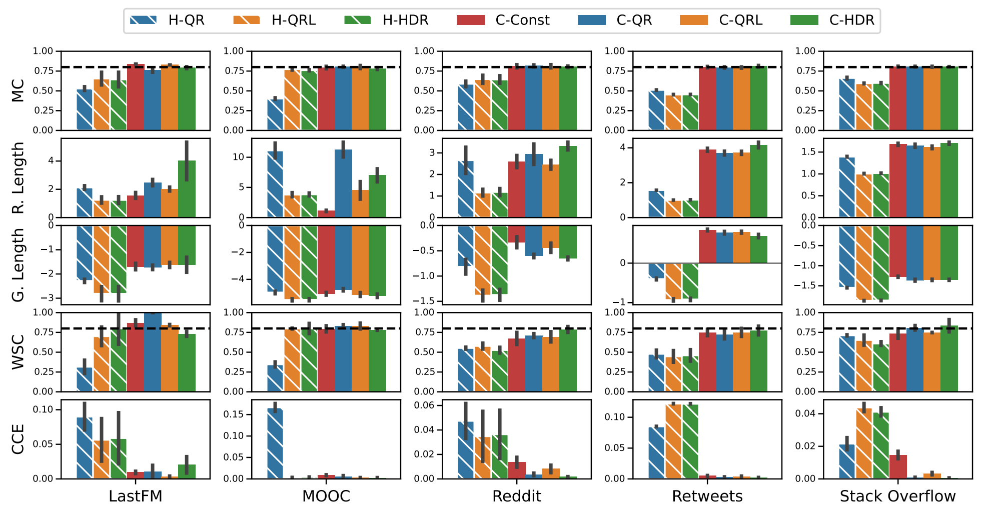


Figure E.4: Performance of different methods producing a region for the time on real world datasets using the RMTPP model.

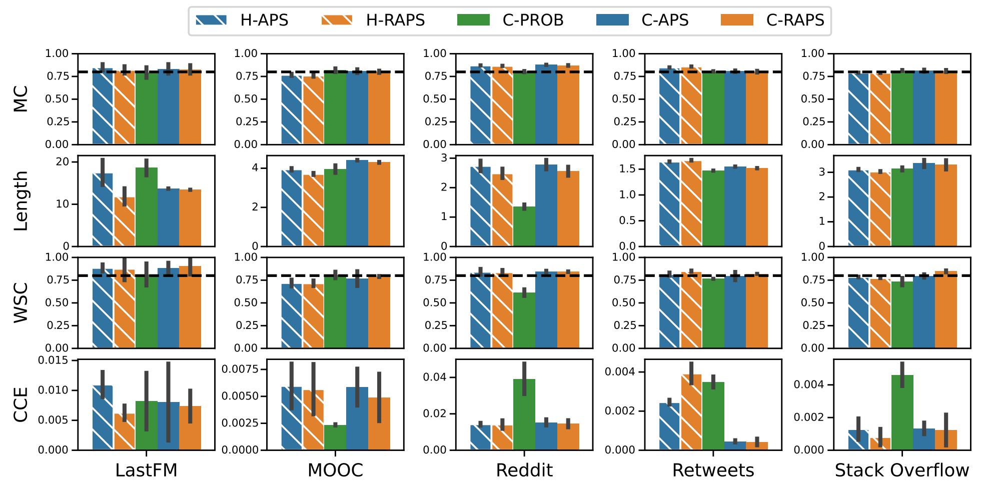


Figure E.5: Performance of different methods producing a region for the mark on real world datasets using the RMTPP model.

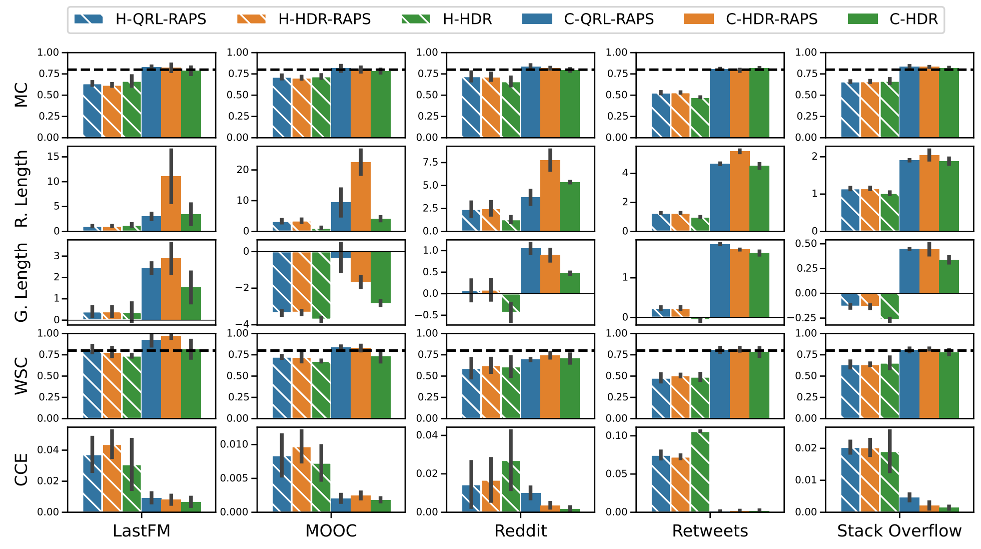


Figure E.6: Performance of different methods producing a joint region for the time and mark on real world datasets using the RMTPP model.

E.1.3 SAHP

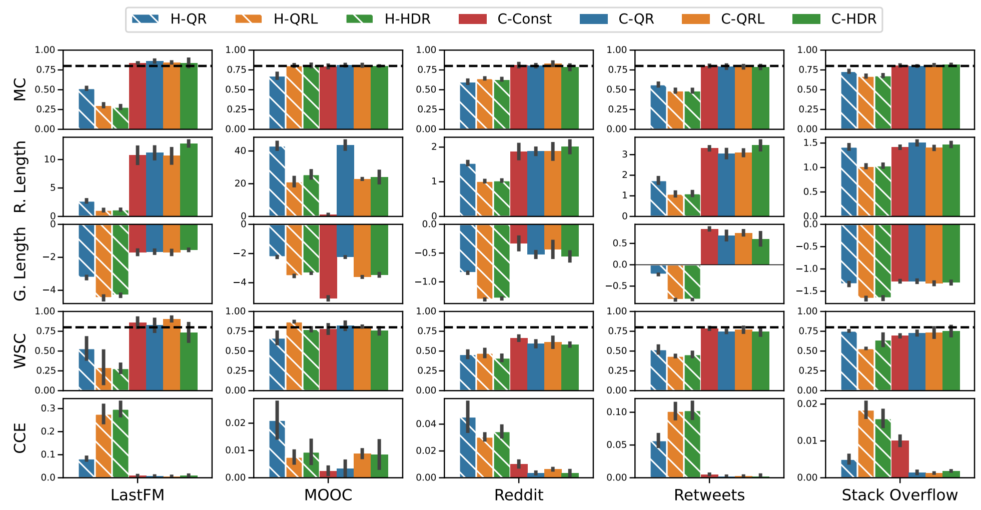


Figure E.7: Performance of different methods producing a region for the time on real world datasets using the SAHP model.

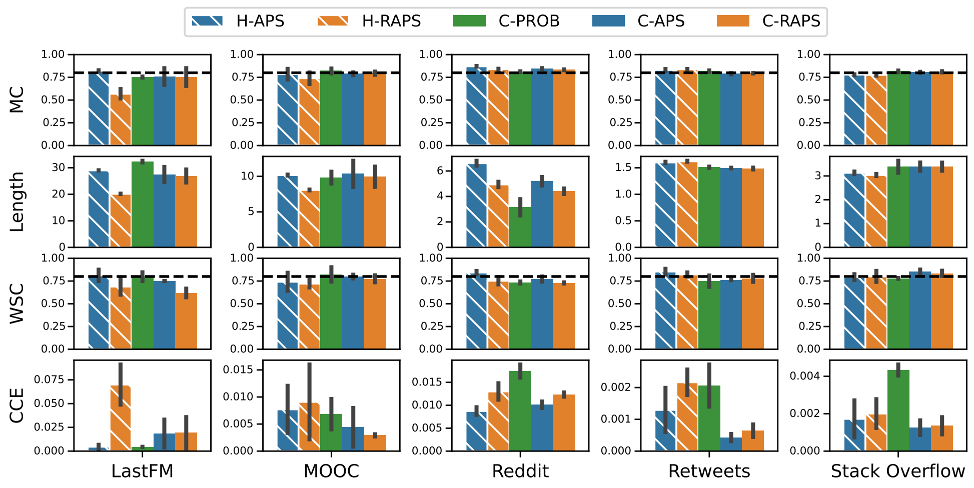


Figure E.8: Performance of different methods producing a region for the mark on real world datasets using the SAHP model.

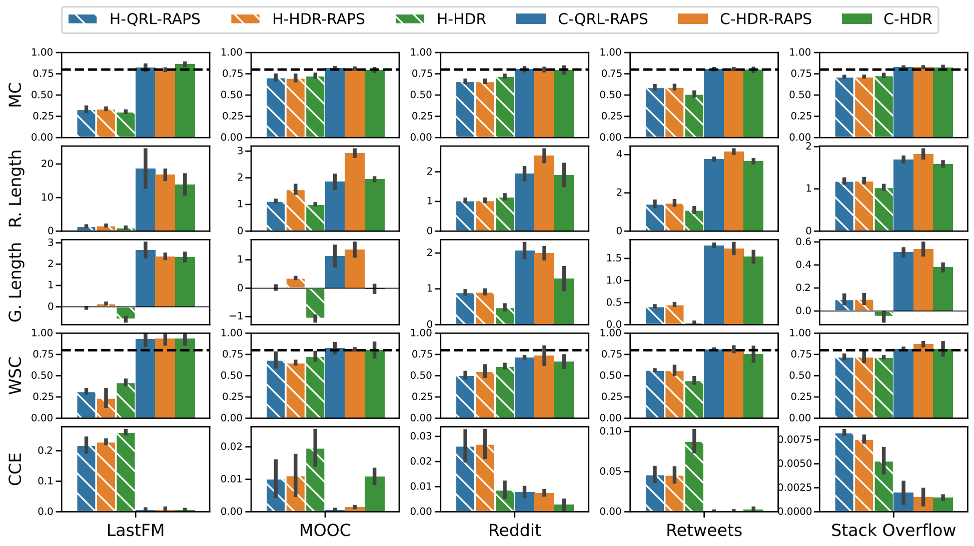


Figure E.9: Performance of different methods producing a joint region for the time and mark on real world datasets using the SAHP model.

E.2. Results on Other Real and Synthetic Datasets

In this section, we report the results on the Github, MIMIC2, Wikipedia, and Hawkes datasets for all models and all scenarios. Ideally, the MC and WSC metrics should align with the nominal coverage $1 - \alpha = 0.8$, while the relative length (R. Length), geometric length (G. Length), and CCE metrics should be minimized. We note that the findings on these datasets are also generally consistent with our conclusions from Section 5.5.5. Nonetheless, we usually observe a large variability in the results for Github, MIMIC2, and Wikipedia, explained by the few number of observations in the calibration and test sequences. We therefore invite the reader to exercise caution when interpreting the findings on these real-world datasets.

Finally, for the Hawkes dataset, we observe that heuristic methods tend to already attain the desired coverage level. This finding may be explained by a too simplistic underlying generative Hawkes process, which is already well fitted by the MTPP models. We plan to investigate more complex simulated point processes as part of our future work.

E.2.1 LNM+

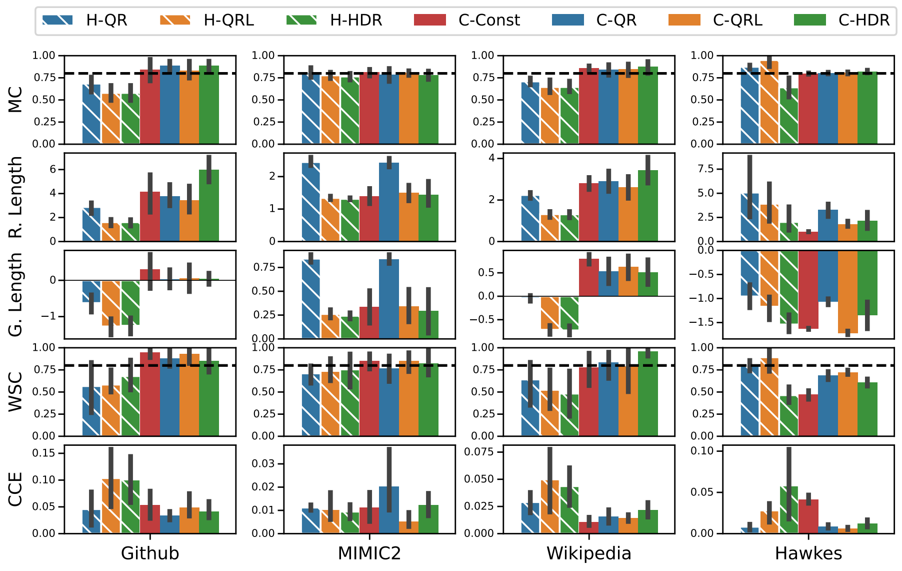


Figure E.10: Performance of different methods producing a region for the time on the datasets not discussed in the main text using the LNM+ model.

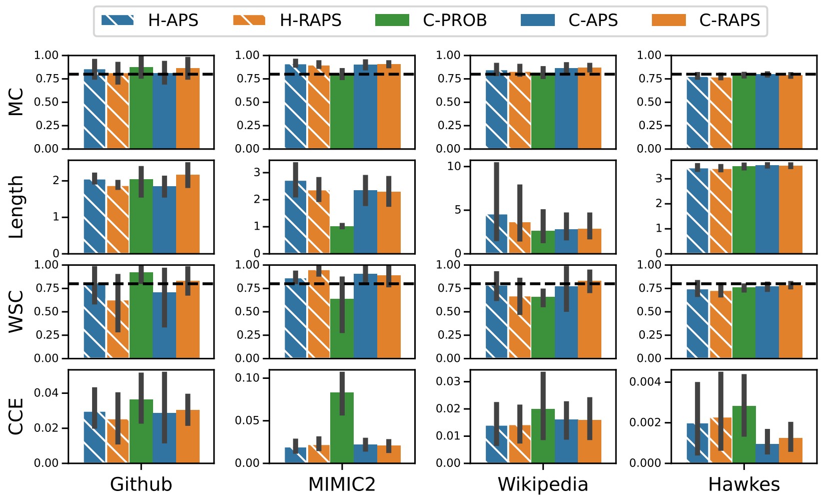


Figure E.11: Performance of different methods producing a region for the mark on the datasets not discussed in the main text using the LNM+ model.

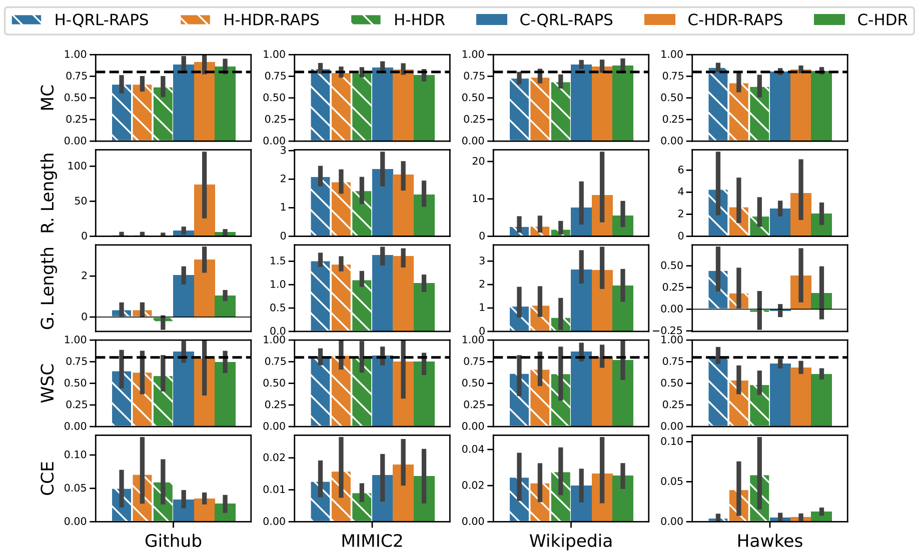


Figure E.12: Performance of different methods producing a joint region for the time and mark on the datasets not discussed in the main text using the LNM+ model.

E.2.2 FNN

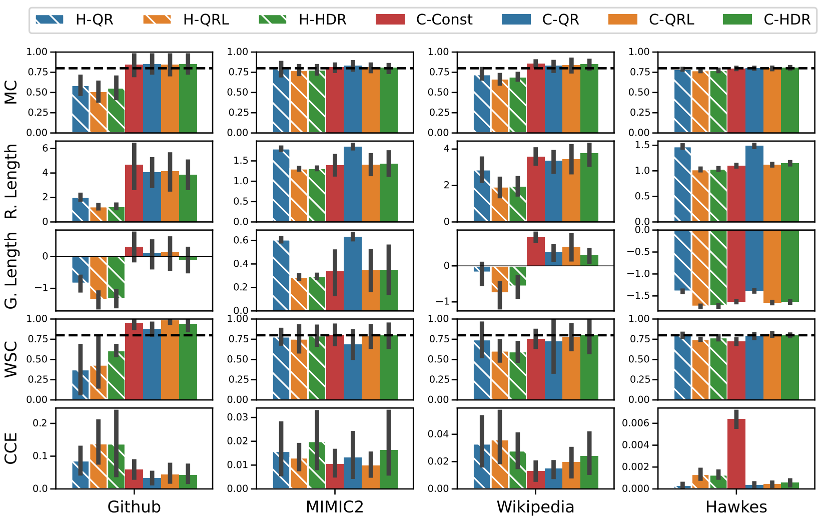


Figure E.13: Performance of different methods producing a region for the time on the datasets not discussed in the main text using the FNN model.

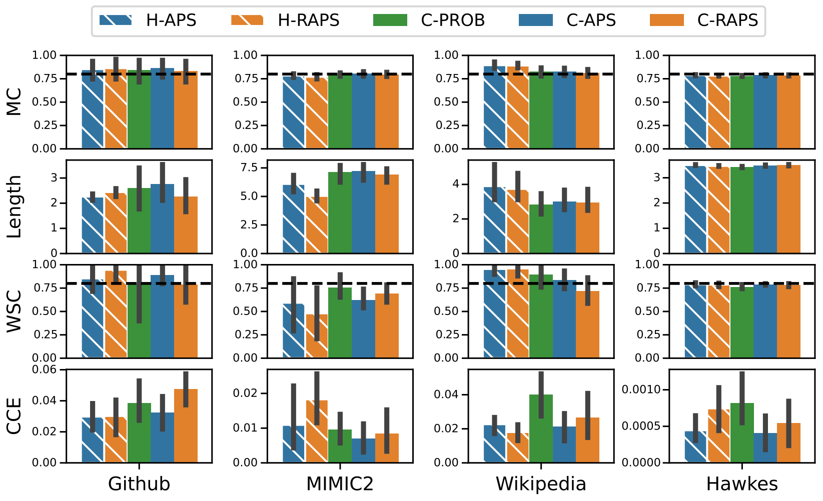


Figure E.14: Performance of different methods producing a region for the mark on the datasets not discussed in the main text using the FNN model.

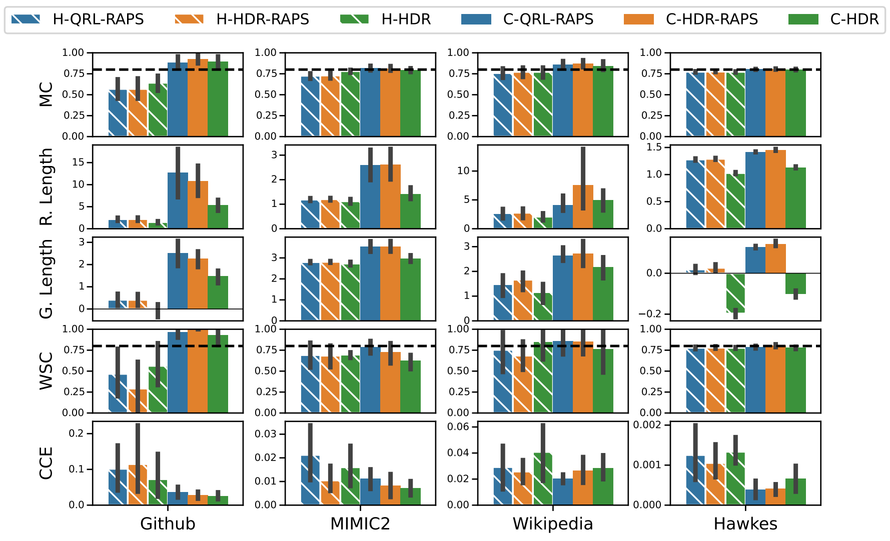


Figure E.15: Performance of different methods producing a joint region for the time and mark on the datasets not discussed in the main text using the FNN model.

E.2.3 RMTPP

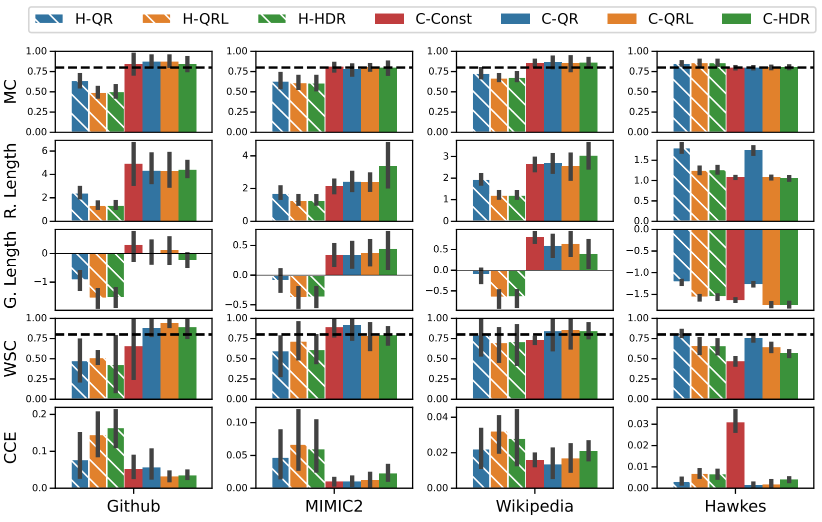


Figure E.16: Performance of different methods producing a region for the time on the datasets not discussed in the main text using the RMTPP model.

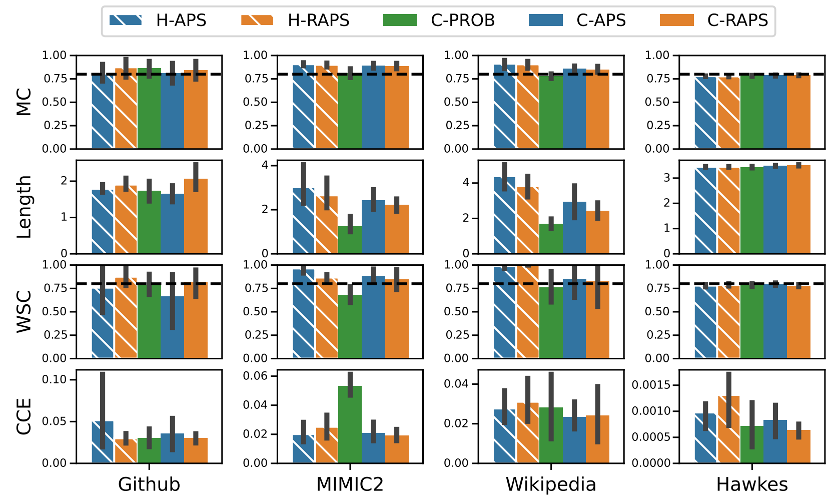


Figure E.17: Performance of different methods producing a region for the mark on the datasets not discussed in the main text using the RMTPP model.

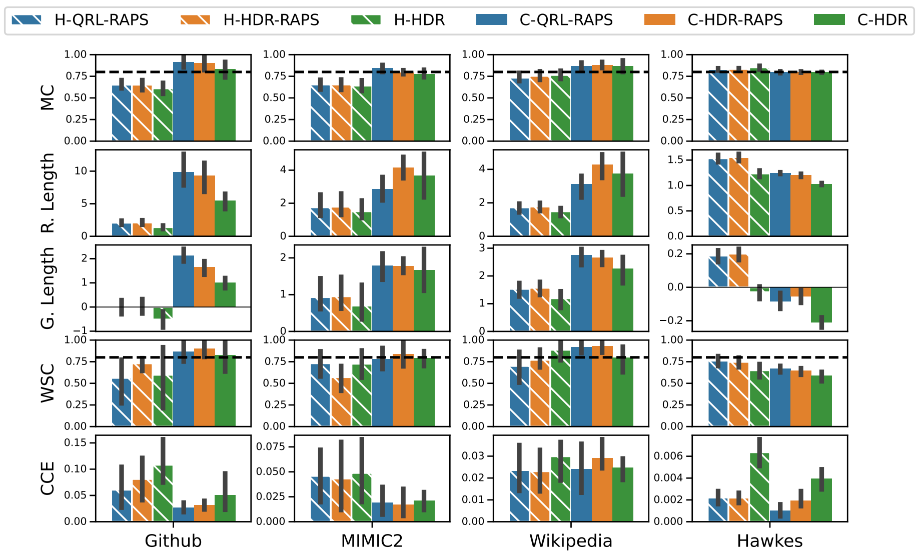


Figure E.18: Performance of different methods producing a joint region for the time and mark on the datasets not discussed in the main text using the RMTPP model.

E.2.4 SAHP

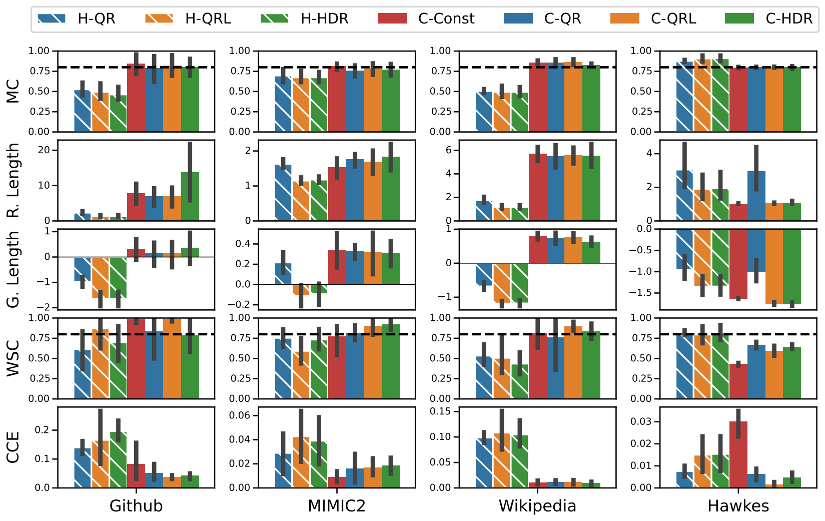


Figure E.19: Performance of different methods producing a region for the time on the datasets not discussed in the main text using the SAHP model.

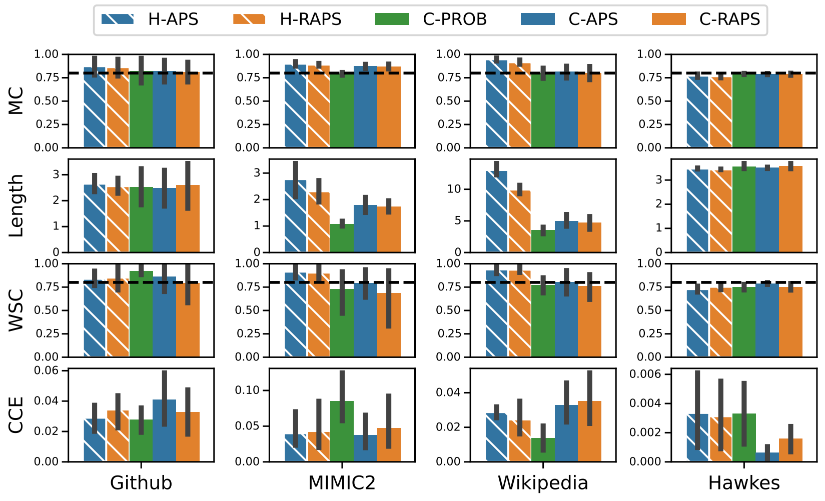


Figure E.20: Performance of different methods producing a region for the mark on the datasets not discussed in the main text using the SAHP model.

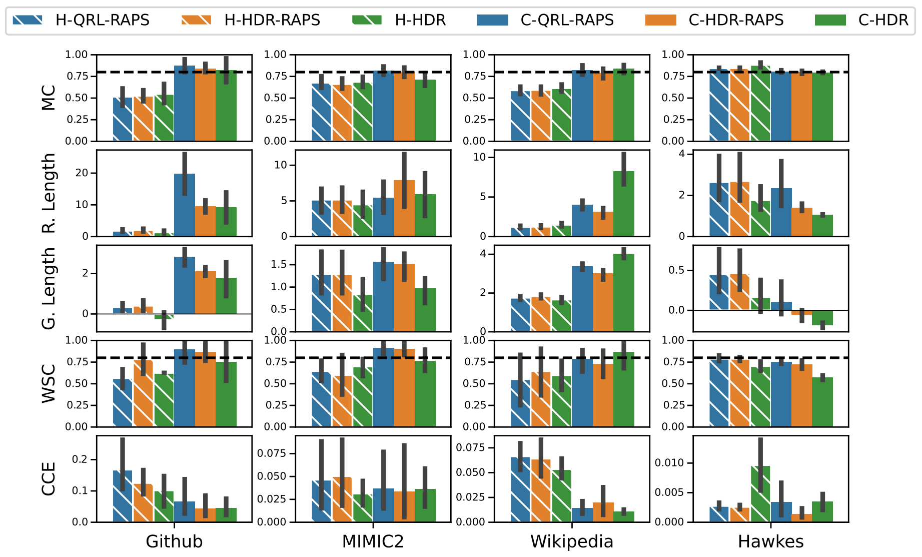


Figure E.21: Performance of different methods producing a joint region for the time and mark on the datasets not discussed in the main text using the SAHP model.

E.3. Coverage Per Level

Section 5.5.5.4 discussed the empirical marginal coverage obtained at different coverage levels for methods that generate a joint prediction region on the arrival time and mark. In this section, we present additional results for methods that generate a prediction region individually for either the time or mark.

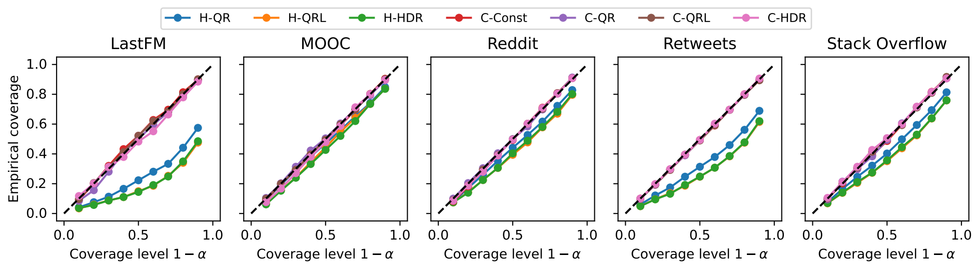


Figure E.22: Empirical marginal coverage for different coverage levels for methods that produce a prediction region for the time with the LNM+ model.

Fig. E.22 shows that conformal methods for the time attain the desired coverage at all levels while heuristic methods generally undercover. This is expected and mirrors the observations in Section 5.5.5.4.

Fig. E.23 shows that all methods, either heuristic or conformal, overcover for small coverage levels, while coverage is attained for high coverage levels. The reason is that all methods that generate a prediction set for the mark guarantee that prediction sets are not empty by always adding the class with the highest probability, as presented in Section 5.5.2. We do not observe overcoverage for high coverage levels because the class with highest probability will almost always be included. However, for low coverage levels, prediction sets that would normally be empty now include the mark with the highest probability, which leads to increased coverage.

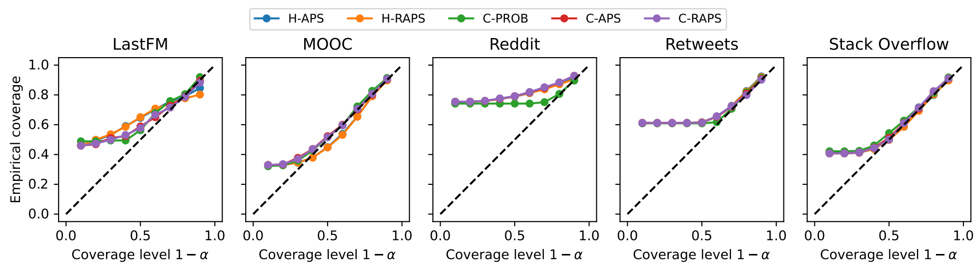


Figure E.23: Empirical marginal coverage for different coverage levels for methods that produce a prediction region for the mark with the LNM+ model.

E.4. An Additional Toy Example of Prediction Regions

In Fig. 5.5 in Section 5.5.5.1, we presented an example illustrating prediction regions for the time for seven methods with $\alpha = 0.5$. For completeness, we provide an additional toy example with $\alpha = 0.2$ and a calibration dataset of 6 data points in Fig. E.24. As in Fig. 5.5, the heuristic methods undercover, achieving a maximum coverage of 4/6, which is less than the desired coverage of 0.8. Notably, H-QRL and H-HDR produce exactly the same prediction regions because the densities are decreasing in this case. Conformal methods adjust the predictions regions to achieve coverage in at least five out of six cases. Similarly to Fig. 5.5, C-HDR generates larger regions on average than other conformal methods despite H-HDR always producing shorter or equivalent lengths compared to H-QR and H-QRL.

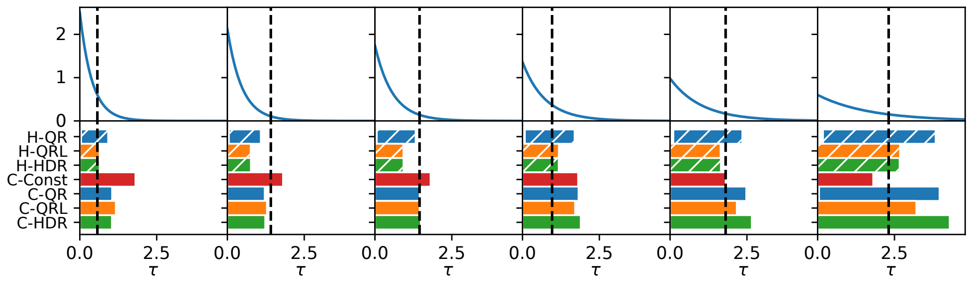


Figure E.24: Toy example with $\alpha = 0.2$ and a calibration dataset of 6 data points.

E.5. Computational Time

In Table E.1, we present the computation times for evaluating the scores and regions across each conformal method utilized in our experiments. Except for `C-Const`, which incurs a minimal computation time primarily due to data loading, the computational demands of the other methods are of a comparable magnitude.

For all methods excluding `C-Const`, computation time is primarily governed by the calculation of the CDF of the time and the joint PDF of the time and mark. Specifically, the most resource-intensive tasks involve computing the quantiles of the time or generating samples from the time distribution, as these operations require inverting the CDF using the bisection method, typically necessitating around 30 evaluations.

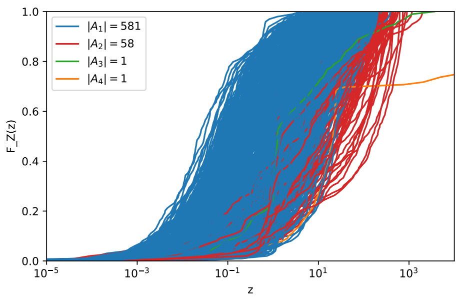
In the cases of `C-QR` and `C-QRL`, the computation time is dominated by computing the quantiles of the time. For `C-HDR` (time and joint), `C-PROB`, `C-APS`, and `C-RAPS`, the primary computational load comes from generating time samples. More specifically, for `C-HDR`, these samples are needed to compute HPD values. For `C-PROB`, `C-APS`, and `C-RAPS`, computing the marginal PMF of the mark relative to the time involves averaging the joint density over the time across these samples.

Table E.1: Time to compute the scores and regions for all considered conformal methods on real world datasets using the CLNM model, averaged over 5 runs, in seconds.

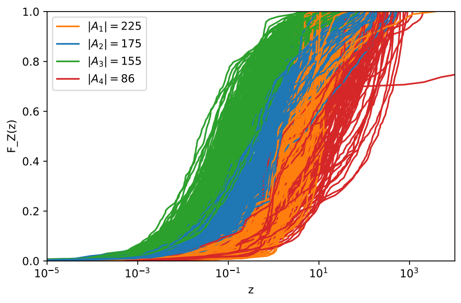
Dataset	**Compute type**	**Time** C-Const	C-QR	C-QRL	C-HDR	**Mark** C-PROB	C-APS	C-RAPS	**Joint** C-QRL -RAPS	C-HDR -RAPS	C-HDR
LastFM	Score	0.07	15.01	8.10	8.63	8.40	8.40	8.36	16.56	17.11	8.71
	Region	0.10	9.97	5.42	10.51	5.61	5.62	5.61	10.34	15.55	11.70
MOOC	Score	0.38	93.17	47.64	51.70	49.68	49.33	49.71	96.90	102.07	51.88
	Region	0.75	62.32	32.06	66.46	33.30	33.36	33.58	64.24	99.40	74.98
Reddit	Score	0.25	56.47	29.04	31.51	30.34	30.23	30.19	59.48	61.97	31.93
	Region	0.48	38.31	19.84	40.60	20.72	20.78	20.73	39.48	60.66	45.68
Retweets	Score	0.60	159.38	80.30	85.84	84.54	84.27	84.13	165.46	171.16	86.88
	Region	0.56	106.49	53.93	112.55	56.77	56.43	56.60	110.03	169.60	114.00
Stack Overflow	Score	0.45	105.68	53.33	57.20	55.86	55.92	55.76	113.25	112.95	57.55
	Region	0.58	71.12	35.91	75.17	37.90	38.14	37.88	79.21	111.91	79.11

E.6. Partitioning for Conditional Coverage

In this section, we elaborate on the choice of distance function to create the partitions for the metric CCE introduced in Section 5.5.4. On Fig. E.25, we show the CDF of Z for all instances in the calibration dataset of the Reddit dataset. As shown in Fig. E.25a, instances where the distributions of Z have the longest tails exhibit extreme distances from other distributions, resulting into their isolation into small clusters. Fig. E.25b shows that, by instead focusing on the random variable $\log Z$, we achieve more balanced cluster sizes, which is crucial to have an accurate estimation of coverage within each partition.



(a) Partitions obtained with the distance d_Z.



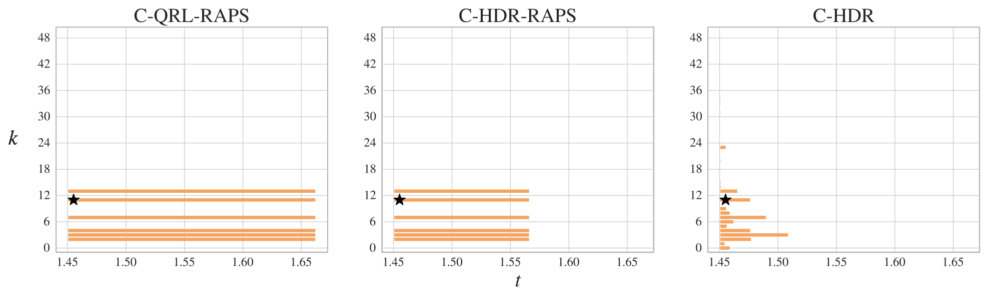
(b) Partitions obtained with the distance $d_{\log Z}$.

Figure E.25: The two subfigures show the CDF of Z for all instances in the calibration dataset of the Reddit dataset. The colors determine the partition in which an instance falls according to the distance d_Z on the left and $d_{\log Z}$ on the right, as introduced in Section 5.5.4. The legend denotes the size of each cluster of the partition.

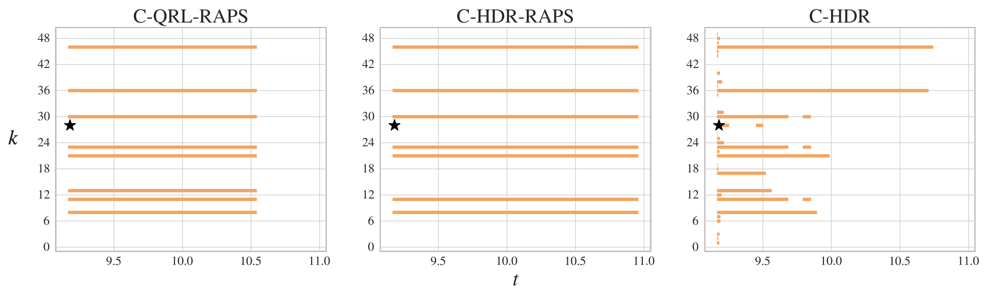
E.7. Additional Examples of Joint Prediction Regions

Figure E.26 presents additional predictions regions generated by conformal methods on the datasets MOOC, Reddit and Stack Overflow, respectively. We observe that C-HDR generally selects more marks than C-QRL-RAPS and C-HDR-RAPS. However, the joint region produced by C-HDR is usually smaller.

(a) MOOC



(b) Reddit



(c) Stack Overflow

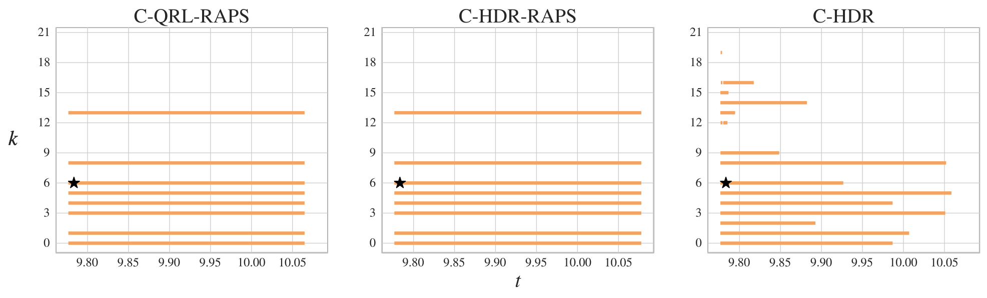


Figure E.26: Example of joint prediction regions generated for the last event of a test sequence.

List of Figures

List of Tables