electronics

MDPI

Article

A Review and Comparative Study of Explainable Deep Learning Models Applied on Action Recognition in Real Time

Sidi Ahmed Mahmoudi [1,*], Otmane Amel [1], Sédrick Stassin [1], Margot Liagre [1], Mohamed Benkedadra [1,2] and Matei Mancas [2]

[1] ILIA Lab, Faculty of Engineering, University of Mons, 7000 Mons, Belgium
[2] ISIA Lab, Faculty of Engineering, University of Mons, 7000 Mons, Belgium
* Correspondence: sidi.mahmoudi@umons.ac.be

Abstract: Video surveillance and image acquisition systems represent one of the most active research topics in computer vision and smart city domains. The growing concern for public and workers' safety has led to a significant increase in the use of surveillance cameras that provide high-definition images and even depth maps when 3D cameras are available. Consequently, the need for automatic techniques for behavior analysis and action recognition is also increasing for several applications such as dangerous actions detection in railway stations or construction sites, event detection in crowd videos, behavior analysis, optimization in industrial sites, etc. In this context, several computer vision and deep learning solutions have been proposed recently where deep neural networks provided more accurate solutions, but they are not so efficient in terms of explainability and flexibility since they remain adapted for specific situations only. Moreover, the complexity of deep neural architectures requires the use of high computing resources to provide fast and real-time computations. In this paper, we propose a review and a comparative analysis of deep learning solutions in terms of precision, explainability, computation time, memory size, and flexibility. Experimental results are conducted within simulated and real-world dangerous actions in railway construction sites. Thanks to our comparative analysis and evaluation, we propose a personalized approach for dangerous action recognition depending on the type of collected data (image) and users' requirements.

Keywords: action recognition; computer vision; deep learning; explainable artificial intelligence; depth maps

Citation: Mahmoudi, S.A.; Amel, O.; Stassin, S.; Liagre, M.; Benkedadra, M.; Mancas, M. A Review and Comparative Study of Explainable Deep Learning Models Applied on Action Recognition in Real Time. *Electronics* **2023**, *12*, 2027. https://doi.org/10.3390/electronics12092027

Academic Editors: Leonardo Galteri, Claudio Ferrari and Stefanos Kollias

Received: 25 February 2023
Revised: 18 April 2023
Accepted: 21 April 2023
Published: 27 April 2023

1. Introduction

Action recognition applications actually represent a very important research topic in computer vision and video surveillance domains. Indeed, the high increase in public and workers' safety concerns has caused a great deal of growth in the number of connected captors such as surveillance cameras. In this context, the technologies of Big Data, Cloud, and Edge Computing have made it possible to increase the production of real scenes and videos. The automatic detection, recognition, and localization of actions are increasingly useful. The main objective of these methods is to detect moving objects within different situations (mobile cameras, variable weather conditions, etc.) using various capture equipment (embedded camera, drone, etc.). In the literature, one can find several action recognition methods based on: 1. image and video processing algorithms that extract motion features in order to recognize the type of actions [1–5]; 2. machine and deep learning techniques that learn from previous examples in order to define a model that should bind each video sequence with its corresponding type such as that proposed in [6] where a recognition pipeline exploits the detected motion features and a recurrent neural network (LSTM) for fall prediction. Authors of [7] proposed a method of action recognition that exploits limited accelerometer and gyroscope data and applies a combination of a convolutional neural network (CNN), Long-Short Term Memory (LSTM), and classical machine learning algorithms

which yields an accuracy of 97.4% with the UCI HAR dataset. In [8], the authors proposed a method called Tree-Network-Tree (TNT) learning framework that provides explainable decisions (partially) through knowledge transfer between the tree model and DNNs. Even with the significant evolution of these methods in terms of precision, they are not always able to provide real-time and generic solutions within various situations. Moreover, deep learning models that provide generally better results are highly hampered by their low explainability and interpretability where they are generally considered black boxes. In this paper, we propose a review and a comparative analysis of deep learning solutions in terms of precision, explainability, computation time, and flexibility. Based on this analysis, we propose a personalized method for action recognition depending on the type of collected data (images) and users' requirements (explainability, real-time, embedded, etc.).

The remainder of this paper is organized as follows: Section 2 describes the related work within classic and deep learning approaches of action recognition. Section 3 presents our experimental and comparative analysis of the literature solutions using public databases; in Section 4, we present our application and selection of the appropriate deep learning model for action recognition in railway stations. The proposed approach is evaluated with different metrics: accuracy, explainability, computation time, memory size, and flexibility. Section 5 concludes and indicates future work.

2. Related Work

In this section, we present the state-of-the-art related to the methods of action recognition and dangerous scenes analysis and prediction. The aim is to target works related to action classification and prediction from videos that can be captured in different situations (mobile cameras, day, night, noisy scenes, etc.). In the literature, one can find two kinds of action recognition methods: 1. generic and semi-automatic approaches; 2. automatic and deep learning approaches.

2.1. Semi-Automatic and Generic Action Recognition Approaches

This approach consists of building pipelines that combine different methods for action recognition. While some pipelines are quite straightforward and only require doing ROI and object detection, others require multiple steps to detect, identify, and track actors in the scene. These pipelines usually comprise some or all of these phases (Figure 1):

- **Object Detection:** Advanced pipelines use some deep-based object detection methods [9] as an initial phase in a multi-phase approach. These object detection methods can be categorized into either one-stage methods such as YOLO [10] and SSD [11], or a two-stage region of interest proposal based methods such as Mask RCNN [12]. Other studies propose a much simpler pipeline by relying solely on single-frame object detection for action recognition through labeling objects with both identity and state information. For example, SW Pienaar et al. [13] used a dataset of toy soldiers, where bounding box annotations specified the state of each soldier who is either standing or lying down, to train an SSD MobileNet model [14] for action recognition through object detection. Improving this approach is carried out through multi-frame information fusion [15], which consists of applying object detection on multiple frames. Then, a majority vote system is applied, where the most detected action including a high number of frames represents the correct action to recognize.
- **Object Tracking:** Numerous approaches exist for multi-object tracking [16], with some building on object detection methods [17]. JK Tsai et al. [18] proposed to detect objects on multiple frames, which are then passed to the DeepSORT [19] object-tracking algorithm coupled with FaceNet [20] for subject identification. This information is fed to an I3D CNN [21] deep neural architecture that is able to take into account this mix of spatiotemporal information to recognize actions in a certain range of frames [22]. Other tracking solutions are not dependent on object detection, such as Siamese-based tracking [23] or correlation filter-based tracking [24]. These methods do not focused on objects in the scene. Thus, they are more abstract in the way they process scene

information. They output different information such as pixel flow or shift as well as latent spatiotemporal features that can be used to perform other tasks.

- **Action Recognition:** This phase of the pipeline consists of feeding the acquired information through object tracking to a model that can relate these features to actions or events. In addition to the above-mentioned example, and in the case of subject-dependent object tracking, a trajectory can be used as in [25] where the authors applied action recognition from temporal trajectory data through Long Short-Term Memory (LSTM) and Temporal CNNs.

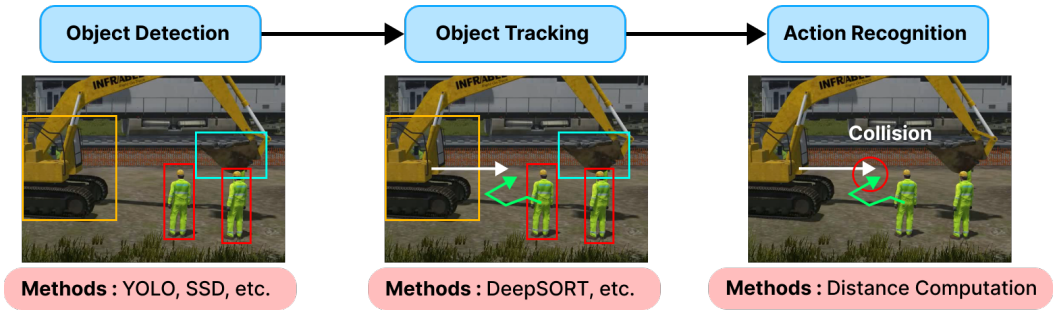


Figure 1. Example of semi-automatic and generic action recognition approach.

Other approaches such as [26–29] use object-independent tracking methods that rely more on spatiotemporal-based features. While these features decrease the accuracy of recognizing different actions, they employ methods such as 2D CNNs [30] but more commonly 3D CNNs [31] to predict actions in the scene in a more generalized manner.

2.2. Automatic Approaches for Action Recognition

Machine and deep learning methods can be used for both action detection and prediction, where the main advantage is represented by their independence since they do not require system initialization or scene annotations. In fact, thanks to the learning process, the models can benefit from the variety of training databases where several types of actions and environments are presented. However, the architecture of these models must be deep and complex to effectively extract relevant features while accounting for spatial, temporal, and depth information of actions, as well as the diverse environmental factors, such as noise, weather conditions, and varying day/night conditions. Additionally, the decision-making process of deep neural networks lacks explainability and interpretability, often leading to their perception as "black boxes". Fortunately, several explainability approaches have been proposed in recent years, mainly enabling the explanation of deep neural decisions by the identification of the responsible parameters and image pixels of each model output. As a consequence, this subsection is further developed within five parts: (1) features extraction methods for action recognition; (2) deep learning methods for action recognition; (3) depth-based methods for action recognition (4) explainability of deep neural networks; and (5) contribution.

2.2.1. Features Extraction Methods for Action Recognition

These methods consist generally of modeling normal movements (actions) to distinguish abnormal movements. The main idea is that the major available data are related to normal behaviors and abnormal movements are generally unavailable. Thus, the deviations from normal movements are considered an abnormality [1,2,4,5]. The modeling and description of movements are generally based on the extraction of optical flow vectors that provide a pertinent description of movements in terms of velocity and direction for all frame pixels (dense optical flow) or for the detected points of interest only (sparse optical flow). The main advantage of these methods is their interpretability and explainability since we can easily define the responsible features of each decision. However, these methods are hampered by their lesser flexibility and genericity (to various situations). They are

more adapted to global actions that are related to crowd videos without a specific focus on workers or people. Figure 2 illustrates the general process of this kind of approach.

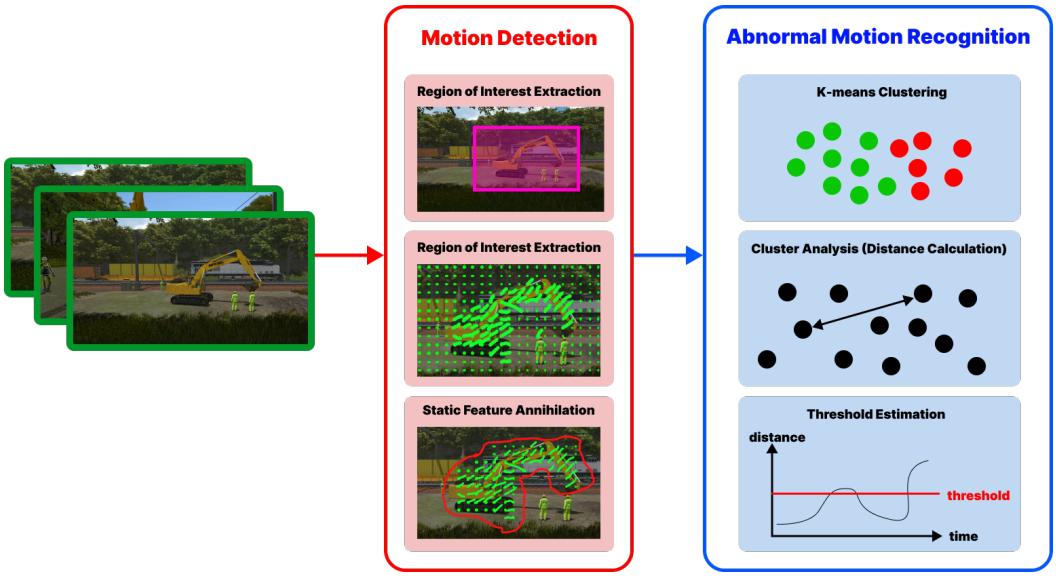


Figure 2. Features extraction process for action recognition: green (input), red (detection), blue (recognition).

2.2.2. Deep Learning Methods for Action Recognition

The main current existing deep learning approaches are based on convolutional neural networks (CNNs) [32], recurrent neural networks (RNNs) [33], and transformer architectures [34,35]. CNNs are used to consider the spatial information of actions (pixels and objects' positions within video frames), while RNNs are used to consider the temporal information of actions (moment and duration of actions within videos). Each neuron in RNNs is connected to the neurons of the previous layer and to its previous result (corresponding to the processing of the previous frame in this specific case). Notice that RNNs are hampered by the problem of a vanishing gradient generating a small change in weights and, therefore, a small improvement in accuracy within successive iterations during the training process [36]. To deal with this problem, the LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) neural networks [37] are used to limit and select the data in the memory. On the other hand, transformers are able to handle long-term memories by focusing on important aspects independently of the fact that they are close or far in the past. We describe here the main deep learning models in the literature: Two-Stream networks, Convolutional networks, Temporal networks, and transformers. Figure 3 illustrates the general process of deep learning-based approaches.

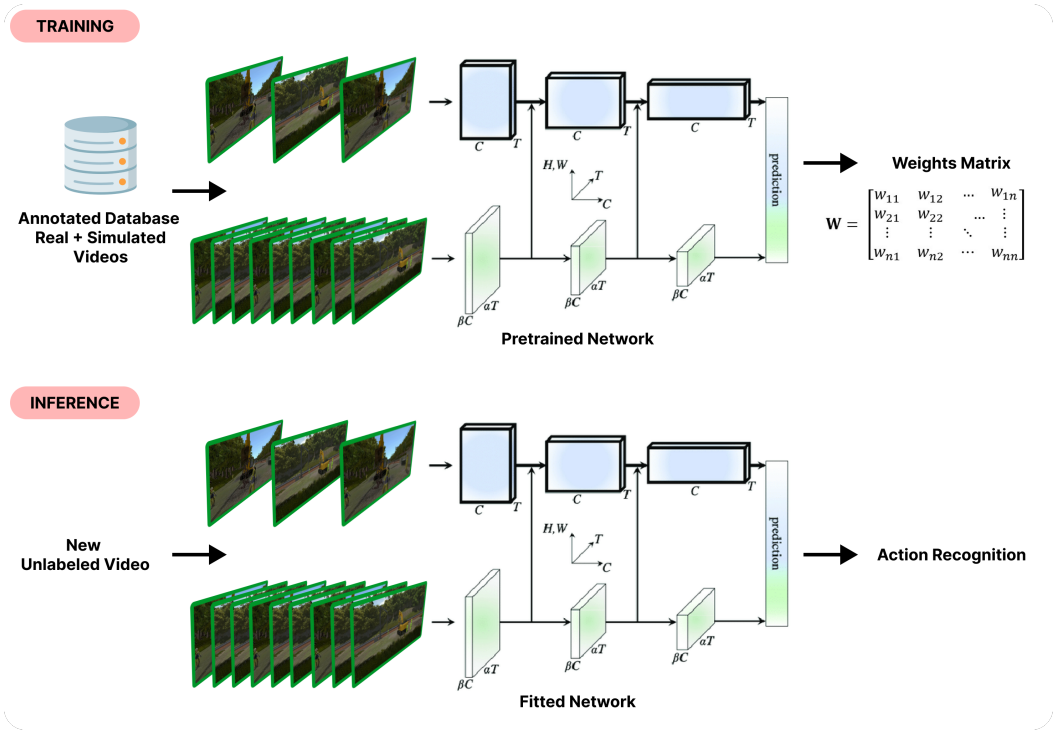


Figure 3. Deep Learning process for action recognition.

1. **Convolutional networks:** generally, two-dimensional CNNs are used for 2D image classification and object detection. In contrast, 3D convolutional networks (3D ConvNet) can be used to incorporate the third dimension, which, in the context of action recognition, represents a brief temporal sequence of frames that compose the actions and movements, such as those proposed in [22]. Authors of [38] proposed to combine a 2D convolutional (CNN) with recurrent networks (RNNs) in order to take into account, respectively, spatial and temporal information. The recurrent connection within RNNs, and more particularly LSTMs, facilitates taking into account the previous image features within actions and thus detecting more accurately the type of action. Another method is proposed in [39] that applied spatial 2D convolutions followed, separately, by temporal convolution (1D) providing a specific convolution called (2+1D). This factorization offers significant gains in terms of precision thanks to the extraction of fine features.
2. **Two-Stream networks:** the two-Stream reference method [40] is based on the operating principle of the human visual cortex, which is represented by two parts: the ventral part is involved to recognize objects, while the dorsal part is involved to recognize movements. Similarly, the two-Stream neural network architecture is composed of two parts: spatial and temporal where each one consists of a convolutional neural network. The spatial part allows the detection of actions based on the extracted images and object features while the temporal part allows improving the precision of action recognition by considering the order and frame succession in time. This evolution in time is calculated within motion vectors, estimated by the optical flow method [41]. The merging of the two parts' features can be performed at the first layers "Early fusion", the last layers "Late fusion" or in a progressive way "Slow fusion". Authors in [42] found that the late fusion yields the best results. The two-Stream architecture has been improved by several works such as [43] that integrated the trajectory features. The authors of [44] proposed a novel spatiotemporal pyramid network for combining the spatial and temporal features in a pyramid structure in order to improve their reinforcement. In [45], the authors proposed to connect static and optical flow channel streams in order to increase the interactions between the streams. In [46], a new video representation was proposed for action detection by

the aggregation of local convolutional features across the complete spatiotemporal extent of the video. The authors of [21] proposed to replace 2D convolutions (in a two-Stream network) with 3D convolutions in order to take into account the frames' succession within the actions. Alternatively, the authors of [47] proposed a 3D CNN network that operates on RGB images and generates motion in order to avoid the computation of optical flow vectors, which are computationally intensive. During training, this method consists of minimizing the difference between the produced motion stream and motion features from the optical flow vectors. This allowed providing accurate results without the computation of optical flow offering real-time inference.

3. **Temporal networks:** these methods allow the consideration of temporal information without using recurrent cells or motion vectors. Indeed, the methods try to extract the short, medium, and long temporal changes between successive frames. In this context, authors in [48] proposed a flexible solution that models long-range temporal structures where the video is divided into segments having the same duration. A short video extract is randomly selected from each segment and passed to two-Stream networks composed of two CNNs, one for spatial characteristics and the other for temporal characteristics. In [49], the authors exploit 2D CNNs for action recognition by adding the particularity of moving the learned features between neighboring images. It is a very simple approach that considers the time dimension by keeping the use of 2D CNNs, less intensive in computation, compared to 3D CNNs or two-Stream networks. Another method called "SlowFast" [50] is inspired by biological studies, considering a video as a sequence of slow and fast movements. SlowFast is designed to combine two different paths in the network: one with a low rate (slow), and the other with a high rate (fast) of images from the video. The slow rate path enables the capturing of spatial information of slow movements, while the fast part is devoted to detect spatial information of fast movements.
4. **Transformers:** a transformer is a deep learning architecture, introduced in 2017, based on the consideration of attention and self-attention mechanisms, which allow quantifying (weight) the importance of each part of the input data. Transformers were initially proposed for natural language processing (NLP) [51], and due to their promising results, their usage was extended for computer vision (CV) applications such as proposed within Vision transformers Vit [35]. As proposed within recurrent neural networks, Transformers can take into account temporal information. The main difference is that transformers can process the complete input at once thanks to the attention mechanism. Several transformer architectures have been proposed such as BERT [52] and GPT [53], which have provided excellent results for natural language processing and interpretation such as provided within ChatGPT tool [54]. Furthermore, transformers represent a powerful solution for video understanding. The authors of [55] proposed an improvement of **R2+1D** architecture [39] by replacing the Temporal Global Average Pooling (TGAP) layer present at the end of **R(2+1)D** network by the BERT attention mechanism. In fact, the TGAP does not consider the order or the importance of the temporal features learned by the network. With BERT and its "multi-headed" attention mechanism, we can learn this importance in addition to the R(2+1)D method. This permitted the improvement of the accuracy of action recognition within several public databases. In [56], authors proposed an extension of the vision transformer architecture (Vit) by integrating a temporal dimension. The TimeSformer takes as input a video clip, which is represented as a four-dimensional tensor of dimension: $h \times w \times c \times f$, where h and w are the height and width of each frame, c is the number of channels, and f the number of frames in the video. Then, the Timesformer divides each frame into n patches where each one is flattened into a vector representing spatial information within the patch.

2.2.3. Depth-Based Methods for Actions Detection

The use of the depth information, provided by 3D cameras, can be very helpful for action recognition since the 3D information is insensitive to brightness and can therefore work in various situations (darkness, noise, etc.), which can be a real advantage to get a generic solution. In this context, [57] proposed an approach called **DMM-HOG** that projects the depth information, collected under the three Cartesian views (front, side, and top over time) and apply the HOG (Histogram of Oriented Gradients) [58] descriptor on each view to extract the shape features of the moving objects. Based on these features, a support vector machine (SVM) classifier [59] is applied to recognize actions. The DMM-HOG method was improved in **DMMs, DMM** [60] by calculating a depth map "Depth Motion Maps" (DMMs) represented by the difference between two images, for the entire sequence of the video. Then, the classification is provided with the "Sparse Coding" approach [61]. The method called **HON4D**, [62] included the depth information « z » which permits to obtain 4 dimensions (x, y, z and t for time). The depth information makes it possible to capture the surfaces representing the geometry of each action and thus recognize the type of movement. In [63], authors proposed a method, that we call **HistRF**, combining the features extracted from two or three histograms (spatial-temporal depth histogram, depth derivatives histogram, RGB histogram) where each one represents an additional contribution. Then, the extracted features from histograms are combined and treated by a Random Forest algorithm [64] to select the pertinent features and recognize actions. Authors in [65] proposed a new approach, called **MPDMM**, that does not project the 3D depth information along the three Cartesian views but along various angles and positions, allowing the extraction of more features to represent actions. Then, the difference between the projected images is applied for the whole video, and the LBP [66] descriptor is used to extract texture features. Finally, a classifier (ELM; Extreme Learning Machine) [67] is applied for action recognition.

2.2.4. Explainability of Deep Neural Networks

Aside from intrinsically interpretable models (e.g., linear models or decision trees), there are black-box models such as deep neural networks whose results do not directly produce explanations. Thus, post-hoc explainability methods, applied to input samples, are commonly used in the state-of-the-art to obtain an explanation of the reasoning of the model. Post-hoc methods can be divided into three types.

First, gradient-based methods start from the output neuron and backpropagate its value to the input [68], obtaining significance-related weights for each input feature. These weights can be interpreted as input relevance for text features or visualized by a saliency map in an image or action classification to understand the most important parts for the decision. Backpropagation variants are, for example, SmoothGrad and Integrated Gradients: SmoothGrad [69] creates neighbor input samples by adding Gaussian noise to the features and averages the gradients computed on each of them. Integrated Gradients [70] interpolates the input sample with a baseline image (e.g., black, white, or uniform image) over an $\alpha[0, 1]$ parameter and averages the gradients obtained.

Second, perturbation-based methods disturb the inputs using a baseline value and analyze the performance drop of the metric used for evaluation. The first such method was Occlusion [71], which perturbed the inputs using square patches and a widely used method in the state-of-the-art is RISE [72]. The latter computes masks based on an upsampling of randomly filled binary masks and creates a heatmap based on the relevance of each mask for the prediction. Those methods are computationally heavier compared to most of the post-hoc methods as they require replacing input values and computing the result of the model at each iteration.

Finally, CAM-based methods are applied to convolutional neural networks and have the particularity of utilizing the activations from the convolutional layers (usually the last one) to produce saliency maps. GradCAM [73] is the most widely used and weights the activation maps by the gradients obtained by a backpropagation from the output neuron

of a chosen class to the convolutional layer. Many variants exist that combine activations from different layers (Layer-CAM [74], Poly-CAM [75]), aggregate the results for the input image at different scales (CAMERAS [76]), or use the activations as masks to predict their importance (Score-CAM [77]).

2.2.5. Contribution

Our contribution is represented by a review and a comparative analysis of deep learning solutions in terms of precision, explainability, computation time, and flexibility for action recognition. Based on this analysis, we propose a personalized method for action recognition depending on the type of collected data (images) and users' requirements (explainability, real-time, embedded, etc.). To provide a fair analysis, we applied standard metrics (accuracy, computation, memory) and developed new metrics related to the explainability of our models: XAI_Acc and XAI_Bias. The metric of XAI_Acc compares the human explanation with the model explanation. XAI_Bias, on the other hand, quantifies the number of situations where actions can be detected as a consequence of the presence of biased information confusing the model decision (Section 4.4). We contribute also by analyzing the generality of deep learning models with a global analysis, taking into account various situations and datasets (real, simulated, and mixed). Through this comprehensive analysis, we could identify the optimal model ensuring accurate, explainable, and embedded action recognition solutions in a real-life use case related to workers' safety within railway stations. Figure 4 illustrates the above-mentioned categories of action recognition methods.

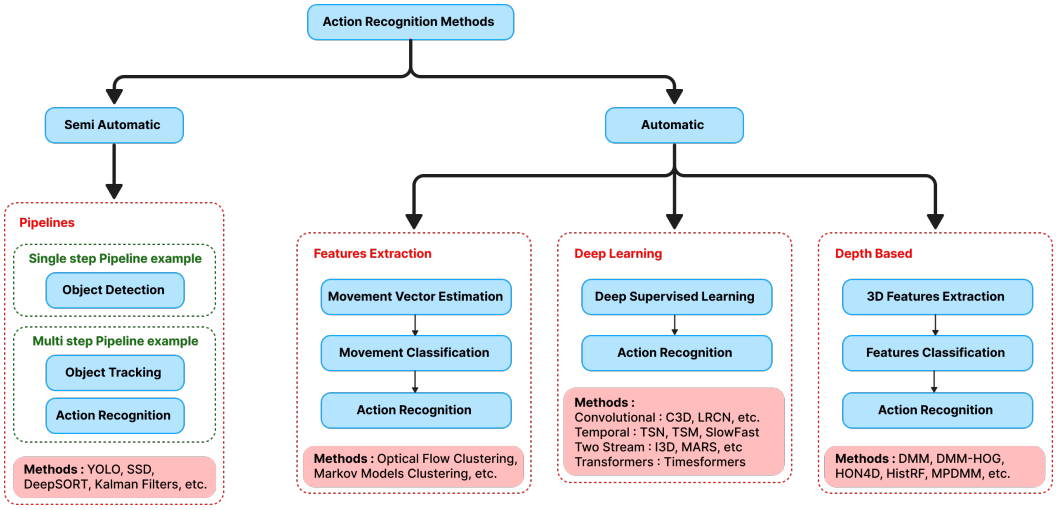


Figure 4. Action recognition approaches, categories and methods.

3. Comparative Analysis of Existing Solutions

The above-mentioned Deep Learning architectures (Section 2.2) demonstrated a high potential for action recognition. The common point between these models is the consideration of both spatial and temporal information but each approach applied a different combination algorithm (3D convolutions, recurrent cells, Two-Stream, transformers, etc.). On the other hand, several methods have been proposed in Section 2.2.3, which consider the depth information for action recognition. These methods are not mainly based on deep learning since they consist of the extraction of the most representative features (with three dimensions) followed by a classifier. To complete this study, we proposed a comparative analysis of these approaches in terms of precision (top1 accuracy), computation time, explainability, and flexibility since these metrics are the most required for action recognition methods within video surveillance and smart cities domain applications. In order to provide a fair comparative analysis, we have selected the same public database "UCF101" which consists of of 101 categories of realistic actions [78] collected from YouTube.

1. **Precision:** we analyzed the precision (top-1 accuracy) of the above-mentioned methods and noticed several observations :
   - The use of 2D convolution networks is not sufficient for action recognition. They are generally replaced by 3D convolutional neural networks or combined with other networks that consider the temporal information of movements.
   - The use of RNNs is designed to deal with temporal information, but they are hampered by the Vanishing gradient problem during the training process. As a solution, Transformer architectures are employed.
   - By considering slow and fast movements during the training process, precision can be improved with a more generalized solution.
   - The incorporation of depth information helps to enhance the results, but it cannot benefit from a deep learning process since there are few annotated databases including the depth information.
2. **Computation time:** the two-Stream networks are very intensive in computation due to the calculation of optical flow and the application of two parts of the neural network. The recurrent neural networks are also very intensive since they need to consider long-term memory in the case of action recognition. On the other side, the use of transformers provided fast training and inference phases thanks to their use of a fast attention mechanism.
3. **Explainability:** as shown in Section 2.2.3, several methods have been proposed recently to identify the responsible parameters (pixels) of each deep learning model and mainly those based on convolutional layers such as proposed within convolution networks, two-Stream, and some temporal networks (Section 2.2.1). However, the explainability of transformer architecture is more complex due to the use of the mechanism of attention. The depth-based approaches are better in terms of explainability since they are mainly based on classical models.
4. **Flexibility:** the presented deep learning approaches methods can be generalized but require regular retraining of the model with the consideration of a significant variation of actions, which is not so easy. However, the depth-based approaches are more suitable for generalization since they are based on texture, shape, and color features extraction before applying a classifier which makes them more independent from the learning data.

More in detail, from Table 1, we can note that the deep learning methods provide better results in terms of accuracy than the depth-based ones. Although the consideration of the 3D information, the accuracy of depth-based methods remains less important since they do not benefit from a learning process using deep neural architectures. Actually, we have a few annotated databases presenting both movements and depth information. Thus, we propose to continue our experiments with deep learning methods for the rest of this paper.

Otherwise, the four categories (convolutional, two-Stream, temporal, and transformers) of deep learning methods provide highly accurate results, where the best model of each category achieves a top-1 accuracy of around 98%.

In terms of explainability (XAI), the models that are mainly based on 2D convolutional neural networks are more easily explainable with the methods described in Section 2.2.3 such as R(2+1D) and Slowfast. The other architectures can be explained partially since they generally use 2D convolutions for one part of the neural network only. In terms of computation time, the temporal networks and transformers provide fast solutions since neither the calculation of optical flow vectors nor the application of 3D convolutions is required. In terms of memory size, we have similar conclusions to those of computation time since there is a natural dependence between the model, the number of extracted features, and thus the computation time. This confirms that temporal networks and transformers can provide the smallest models, which makes them well convenient for deployment on embedded hardware. In terms of flexibility, the temporal network solutions are well-ranked since they have been pre-trained and tested, successfully, on different

databases (UCF101, HDMB [79], etc.). The other architectures are less flexible, which is due to two main reasons:

- They have been tested/pre-trained with one database only.
- They have been pre-trained/tested with different databases but high variation is noted in terms of accuracy.

Table 1. Comparative Study between deep-based and depth-based methods of action recognition.

			Acc	XAI	Time	Mem	Flexibility
Deep Learning methods	**Convolutional**	**C3D**	90.4%	+	- - -	-	-
		LRCN	82.92%	+	- - -	- -	-
		R(2+1D)	**98.17%**	++	++	- - -	-
	Two-Stream	**Two Stream**	86.9%	+	- - -	- -	-
		I3D	**98%**	+	++	+++	-
		MARS	97.8%	+	- - -	-	-
	Temporal	**TSN**	94%	+	- -	+++	++
		TSM	**95.9%**	**+**	**++**	**++**	**++**
		SlowFast	**95.7%**	**++**	**++**	**+**	**+**
	Transformers	**R(2+1)D BERT**	**98.69%**	-	++	- - -	-
		Timesformer	95.43%	-	++	- - -	-
Depth based methods	**With 3D**	**DMM-HOG**	85.52%	+++	+	+	+
		DMM	90.5%	+++	+	+	+
		HON4D	88.89%	+++	+	+	+
		HistRF	88%	+++	+	+	+
		MPDMM	**94.8%**	+++	+	+	+

4. Proposed Approach for Dangerous Action Recognition

The above-mentioned comparative study allowed us to select the most appropriate methods and models in terms of precision, explainability, computation time, memory size and flexibility. In this paper, we are focused on the development of real-time dangerous action recognition that can occur in railway stations. As constraints, we have to provide an accurate, explainable, real-time and embedded solution since the main requirement of customers is to be able to deploy the solution on railway construction sites with a high level of precision, explainability, and flexibility. The metrics of computation time and memory size are also important since we need to deploy the solution in the end on Edge AI hardware such as Jetson Nano (Jetson Nano. https://www.nvidia.com/fr-fr/autonomous-machines/embedded-systems/jetson-nano.Accessedon24April2023 (accessed on 26 April 2023)), Jetson Xavier (Jetson Xavier. https://www.nvidia.com/fr-fr/autonomous-machines/embedded-systems/jetson-agx-xavier (accessed on 24 April 2023)) or Jetson Orin (Jetson Orin). https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin (accessed on 24 April 2023)). This section can be summarized with the following parts: (1) experimental setup; (2) data generation and preparation; (3) deep learning models' analysis and selection; (4) experimental results.

4.1. Experimental Setup

Experiments were conducted using the cluster of the Faculty of Engineering at the University of Mons (UMONS) with the following configuration:

- CPU Processor: Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz with 32 cores, 48 GB of RAM.
- GPU Processor: GTX 1080ti, 3584 CUDA cores with 12 GB of RAM.

In the following parts, the deep learning models are trained and evaluated using the MM action framework [80] implemented in PyTorch [81]. The database underwent a preprocessing step, as described in Section 4.2 and was split into train, test, and validation sets with a ratio of 70%, 15%, and 15%, respectively.

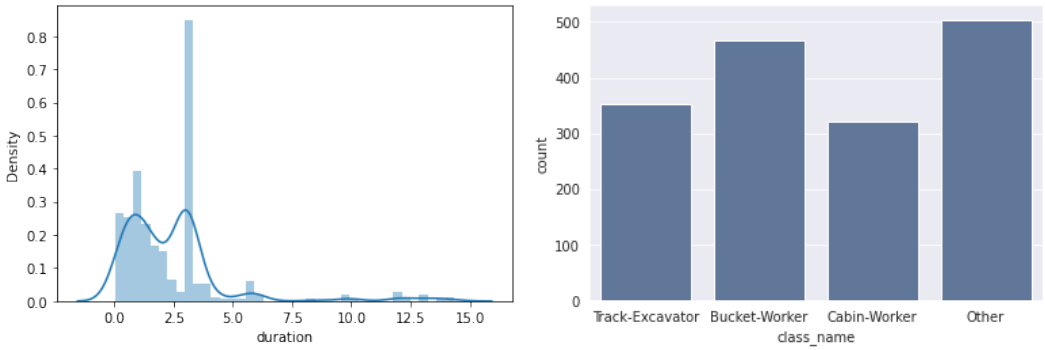
4.2. Data Generation and Preparation

The main difficulty faced in this study is the hard collection of real scenes presenting different dangerous situations since this kind of danger happens rarely. Still, when it happens it can be very dangerous and even deadly in some situations. As a solution, we proposed to generate, within the game engine of Unity (Unity. https://unity.com/ (accessed on 24 April 2023)), artificial videos representing similar environments of dangerous actions. In fact, through the animation of three-dimensional worker personas and a three-dimensional excavator, a wide range of scenarios can be simulated within these environments. The generated dataset is represented by four classes (normal, bucket-worker, cabin-worker, and track excavator). Each scene is represented by four views in order to have a high variation of situations. Notice that each recording has a duration of 2 min where the annotations are represented by the delimitation of the actions with frame numbers (begin to end) as references. The generated dataset, following a trimming step, is outlined in Table 2. Figure 5 illustrates some of the actions that are textually defined in Table 3, while Figure 6 presents the class distribution and duration of actions in our generated dataset.

Table 2. This table summarizes the simulated dangerous actions. 'Classes' lists the different action types, 'Clips' gives the total number of samples per class, 'Duration' provides the duration of clips in seconds, and 'Frames' is the total number of frames per class.

Classes	# Clips	Duration			Frames Number		
		Min	Max	Avg	Min	Max	Avg
Other	504	0.03	3.03	2.89	1	91	86.71
Cabin-Worker	320	0.23	14.33	3.07	7	430	92.12
Bucket-Worker	468	0.20	5.73	1.16	6	172	35.07
Track-Excavator	352	0.20	13.96	2.72	6	419	81.67

Figure 5. Dataset action examples from left to right are " Bucket-Worker", "Cabin-Worker", "Normal (no danger)", and "Track Excavator".



(**a**) (**b**)

Figure 6. Dataset statistics. (**a**) Action's duration (in seconds) distribution. (**b**) Class distribution of the simulated dangerous actions.

Table 3. Dataset actions definition.

Actions	Definition
Other	No dangerous actions to be notified
Cabin-Worker	Worker moving too close to the cabin while the excavator is being operated
Bucket-Worker	Worker moving under the bucket, In danger of getting hit, or materials may fall from the bucket
Track-Excavator	The excavator moving forward to the tracks (active railway line or electric wires)

On the other side, we have a small database of real dangerous actions represented by three classes: normal, bucket-worker, and cabin-worker. The normal class is represented by 101 videos, the class Bucket-Worker disposes of 88 videos, while the class "Cabin-Worker" is represented by 16 videos. We note that this database is very small compared to the generated one, the class distribution is not fair and we have one class missing "Track Excavator".

4.3. Deep Learning Model Selection and Optimization

We propose to select the most convenient models from those proposed in Section 2, the best model is selected from each category as follows:

- Selected convolutional network: R(2+1D)
- Selected two-Stream network: I3D
- Selected Temporal networks: TSN and SlowFast
- Selected transformer networks: R(2+1D) and Timesformer

We propose to compare experimentally between these models by the use of simulated, real, and mixed (real and generated) video sequences. The objective is to:

1. Define the most convenient model for dangerous action recognition within railway construction sites. This selection is performed based on several metrics: precision, explainability, computation time, model memory size, and flexibility.
2. Define the appropriate database needed for the training of these models among real, simulated, and mixed databases.
3. Define the convenient ratio of real and simulated databases in case of using a mixed database.
4. Propose an approach of action recognition based on users' requirements in terms of precision, explainability, computation time, memory size, and flexibility.

4.4. Experimental Results

In this section, we compare experimentally the five selected models in terms of several metrics:

- **Top-1 accuracy:** provided by the models using the same parameters (simulated dataset, epochs, loss function = cross-entropy, learning rate decay for weights optimization, etc.). This top-1 accuracy is calculated for both the UCF-101 dataset and our simulated database in order to analyze the flexibility of the selected models.
- **Explainability "XAI":** represented by two factors: XAI_Acc and XAI_Bias.
    1. **XAI_Acc:** calculated by comparing human explanation (responsible regions and pixels of each detected action) with the results obtained by one of the best XAI methods. This metric allows assessment of whether the model is focused on the correct regions during the decision phase, as presented in Figure 7.
    2. **XAI_Bias:** defined by the percentage of situation where the model detects accurately an action where the responsible regions of this decision are not correct, which represent a bias. As an example, Figure 8 illustrates an example of accurate detection of "Bucket-worker" action, where the responsible region detected by GradCAM is represented by the good collision region but accompanied by another region "Cloud", which has no relation with this kind of action (this bias is due the quality of our simulated database). Low values of XAI_Bias correspond to a good model having a low level of bias and vice versa.
- **Time:** represented by the computation time of the training process and the test process (248 short videos).
- **Model size:** enables the definition of the suitable models for deployment on embedded resources that dispose of low capacity of memory and calculation.

Figure 7. Correct explainability for detecting the action "bucket-worker".

Figure 8. Example of biased recognition of the action "bucket-worker" detected by GradCAM.

The remainder of this section is presented within two subsections: action recognition within simulated videos and action recognition within mixed videos.

4.4.1. Action Recognition within Simulated Videos

Table 4 shows that the SlowFast model provides the best results for our simulated databases, although it was not the best when using other public databases (UCF-101, HMDB-51, etc.). This means that the Slowfast architecture is the best in terms of flexibility since the model can capture both slow and fast movements. The variation of movement velocities is so present in the case of railway sites. Moreover, we have not noticed overfitting during the training of the model where a small difference is noted between the train, validation, and test top-accuracy values. In terms of explainability, the Slowfast architecture is also providing the best results since the responsible regions of each decision correspond to the human explanation with a precision of 85%. However, we note a high bias rate in

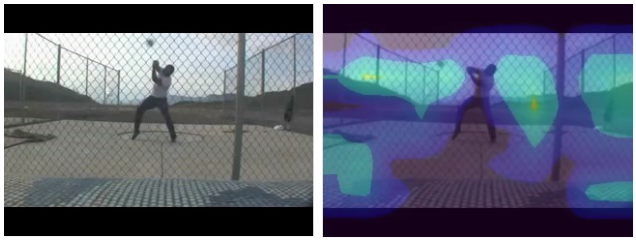
this explainability where the model considers the color of sky and cloud for its decision in addition to the real responsible regions (collision areas) (Figure 8). We extended our experiments by running the same bias detection test on TSN [48] as illustrated in Figure 9. The first example (a) concerns a hammer-throwing action, we can notice that the model's bias stems from its focus on the environment rather than the person. In the second example (b), which depicts a person playing basketball, the model displays a random focus on other regions (such as the floor) instead of the ball or the player. However, in the third example (c), the model demonstrates no bias; despite the presence of text, it accurately identifies the balls and the table and correctly categorizes them as belonging to the "billiard" class.

Table 4. Results analysis: Accuracy, explainability, computation time, memory size and flexibility.

Model Category	Model Name	UCF 101	Simulated Videos (Railway Site)							
		Top-1 Acc	Top-1 Acc			XAI		Time		Model Size (MB)
		Test	Train	Valid	Test	XAI_Acc	XAI Bias	Train	Test	
Conv	**R(2+1D)**	98.17%	93.96%	86.17%	87.90%	55%	**40%**	2 h	25 s	243
Two stream	**I3D**	98%	93.79%	85.38%	85.48%	55%	90%	2h25	60 s	**107**
Temp	**TSM**	95.9%	83.55%	83.40%	86.29%	45%	45%	**0h32**	**16 s**	123
	SlowFast	95.7%	**96.24%**	**90.91%**	**91.53%**	**85%**	55%	2h01	33 s	196
Trans-formers	**R(2+1D) BERT**	**98.69%**	94.60%	88.62%	89.83%	—	—	1h02	17 s	464
	Times-former	95.43%	83.62%	82.21%	81.60%	—	—	1h45	20 s	309

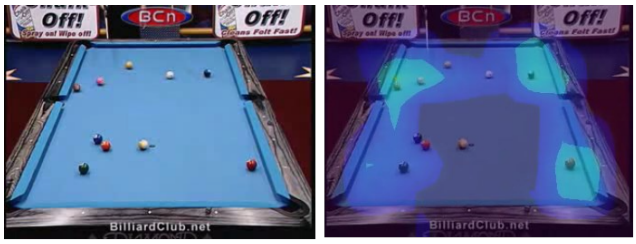
In terms of computation time, the fast models are TSM and R(2+1D) BERT, which benefit from a relatively short training (around 1 h) and provide a fast inference (around 16 s for treating 248 short videos). However, these models provide lower precision and explainability compared to Slwofast. Notice that, the explainability results of the R(2+1D) BERT model are not provided since there is not yet a suitable method for this interpretation.

In terms of memory size, I3D provided the smallest model followed by TSM, SlowFast, and R(2+1D) respectively. On the other side, the transformer architectures provide very large models due to their consideration of the attention mechanism. To summarize, we can consider that SlowFast architecture represents the most convenient model since it has the ability to consider both slow and fast movements where the detected collision can concern a worker moving slowly and excavators that can move faster. This explains the excellent results provided by this model in terms of accuracy and explainability. The main drawback of Slowfast architecture is the computation time compared to the other model but this problem can be solved by the compression and optimization of the deep neural architecture with the use of pruning [82], quantization [83] and knowledge distillation [84] methods.



(**a**)

(**b**)



(**c**)

Figure 9. GradCAM explainability results on UCF101 dataset. (**a**) Example of biased detection of the action "hammer throw" detected by GradCAM. (**b**) Example of biased detection of the action "basketball" detected by GradCAM. (**c**) Correct explainability of the action "billiard" detected by GradCAM.

4.4.2. Action Recognition within Mixed Videos

As shown in the previous section, the bias rate (detected by XAI) remains high when using the best model, which is mainly due to the quality of simulated videos that can not provide sufficient variety of sky and cloud for example. Therefore, we propose to complete our analysis by the use of real videos and also mixed videos since we have a small database of real videos. This dataset consists of recordings taken at railway construction sites, with a total of 101 video clips showing both dangerous and safe actions. Table 5 illustrates the results obtained by the best model "Slowfast", where we can note that the model trained with real videos provides less accurate results, which is due to the small size of this database, the imbalanced classes, and the missing samples of one class (Track Excavator). To deal with these problems, we proposed to work with a mixed database composed of both real and simulated videos, where we noted that the use of simulated videos in addition to real videos contributes to increasing the model accuracy when treating real videos. Notice that in Table 5, when we increased the percentage of simulated videos (ranging from 50% to 75%), the number of real videos was not reduced since we had initially small video samples.

Table 5. Deep Learning for action recognition using simulated, real and mixed databases.

	Training on Simulated Videos	**Training on Real Videos**	**Training on Mixed Videos**		
	100% Simulated 0% Real	**0% Simulated 100% Real**	**50% Simulated 50% Real**	**60% Simulated 40% Real**	**75% Simulated 25% Real**
Test on simulated videos	91.53%	91.53%	87.88%	91.78%	**96.81%**
Test on real videos	39.08%	51.52%	42.42%	48.48%	**54.07%**
Test on mixed videos	66.28%	72.99%	70.08%	73.29%	**78.44%**

5. Conclusions

This paper addressed the problem of action recognition in general and more particularly dangerous action detection in railway stations and construction sites. After a theoretical review of deep learning-based and depth-based methods of action recognition that allowed us to select the most convenient solutions, we have designed a complete comparative study in order to propose the best solution in terms of accuracy, explainability, computation time, memory size, and flexibility. For this study and analysis, we applied standard metrics (accuracy, computation, memory) and developed new metrics related to the explainability (**XAI_Acc and XAI_Bias**) that allow the validation of the accuracy of our explainability algorithm and its ability to recognize biased decision. This study was developed using simulated videos, real videos, and mixed videos. As result, we have identified the Slowfast model as the most appropriate one for dangerous action recognition within railway stations thanks to its high accuracy, flexibility, and explainability. This study demonstrated the advantage of using simulated videos if we combine real and simulated videos for training. In future work, we plan to accelerate the computation time and reduce the model memory size of the Slowfast model by the use of compression methods such as pruning [82], quantization [83], and knowledge distillation [84]. This optimization will be so helpful for the deployment of our solution on embedded and edge AI resources such as Jetson Nano, Jetson Xavier, and Jetson Orin. We also plan to improve the usage of transformers for action recognition by the proposition of an adapted solution for the explainability of these transformer-based solutions.

Author Contributions: Conceptualization, S.A.M., O.A., S.S., M.L. and M.B.; methodology, S.A.M. and M.L.; formal analysis, S.A.M. and M.M.; investigation, S.A.M., O.A. and S.S.; writing—original draft preparation, S.A.M., O.A., S.S., M.L. and M.B.; writing—review and editing, S.A.M.; visualization, S.A.M.; supervision, S.A.M. and M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the company of Infrabel, responsible for the management and maintenance of Belgium's railway infrastructure. Its mission is to ensure the safe, reliable, and sustainable operation of the Belgian railway network. This research was founded within the expertise project between UMONS and Infrabel called "Project Field Worker Protection with AI".

Informed Consent Statement: Experiments were conducted with simulated and real videos provided by Infrabel company. Real videos, provided with hidden faces, are not visualized for confidential reasons.

Data Availability Statement: The database composed of simulated and real videos cannot be published for confidential reasons.

Acknowledgments: We highly acknowledge the support of Infrabel Company for providing the necessary databases and pertinent feedback during this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mahmoudi, S.A.; Sharif, H.; Ihaddadene, N.; Djeraba, C. Abnormal event detection in real time video. In Proceedings of the 1st International Workshop on Multimodal Interactions Analysis of Users in a Controlled Environment, ICMI, Chania, Greece, 20–22 October 2008.
2. Benabbas, Y.; Lablack, A.; Ihaddadene, N.; Djeraba, C. Action Recognition Using Direction Models of Motion. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 4295–4298. [CrossRef]
3. Mahmoudi, S.A.; Kierzynka, M.; Manneback, P. Real-time GPU-based motion detection and tracking using full HD videos. In Proceedings of the Intelligent Technologies for Interactive Entertainment: 5th International ICST Conference, INTETAIN 2013, Mons, Belgium, 3–5 July 2013; pp. 12–21.
4. Benabbas, Y.; Ihaddadene, N.; Djeraba, C. Motion Pattern Extraction and Event Detection for Automatic Visual Surveillance. *Eurasip Jbenabbas2 Video Process.* **2011**, *2011*, 163682. [CrossRef]
5. Mahmoudi, S.A.; Kierzynka, M.; Manneback, P.; Kurowski, K. Real-time motion tracking using optical flow on multiple GPUs. *Bull. Pol. Acad. Sci. Tech. Sci.* **2014**, *62*, 139–150. [CrossRef]
6. Li, J.; Xia, S.T.; Ding, Q. Multi-level recognition on falls from activities of daily living. In Proceedings of the 2020 International Conference on Multimedia Retrieval, Dublin, Ireland, 8–11 June 2020; pp. 464–471.
7. Tufek, N.; Yalcin, M.; Altintas, M.; Kalaoglu, F.; Li, Y.; Bahadir, S.K. Human Action Recognition Using Deep Learning Methods on Limited Sensory Data. *IEEE Sensors J.* **2020**, *20*, 3101–3112. [CrossRef]
8. Li, J.; Li, Y.; Xiang, X.; Xia, S.T.; Dong, S.; Cai, Y. TNT: An Interpretable Tree-Network-Tree Learning Framework using Knowledge Distillation. *Entropy* **2020**, *22*, 1203. [CrossRef] [PubMed]
9. Jiang, P.e.a. A Review of Yolo algorithm developments. *Procedia Comput. Sci.* **2022**, *199*, 1066–1073. [CrossRef]
10. Zhao, Z.Q.; Zheng, P.; Xu, S.T.; Wu, X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef] [PubMed]
11. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
12. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN, 2017. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
13. Pienaar, S.W.; Malekian, R. Human Activity Recognition using Visual Object Detection. In Proceedings of the 2019 IEEE 2nd Wireless Africa Conference (WAC), Pretoria, South Africa, 18–20 August 2019; pp. 1–5. [CrossRef]
14. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
15. Shinde, S.; Kothari, A.; Gupta, V. YOLO based Human Action Recognition and Localization. *Procedia Comput. Sci.* **2018**, *133*, 831–838. [CrossRef]
16. Guo, S.; Wang, S.; Yang, Z.; Wang, L.; Zhang, H.; Guo, P.; Gao, Y.; Guo, J. A Review of Deep Learning-Based Visual Multi-Object Tracking Algorithms for Autonomous Driving. *Appl. Sci.* **2022**, *12*, 10741. [CrossRef]
17. Zhang, Y.; Tokmakov, P.; Hebert, M.; Schmid, C. A structured model for action detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 9975–9984.
18. Tsai, J.K.; Hsu, C.C.; Wang, W.Y.; Huang, S.K. Deep Learning-Based Real-Time Multiple-Person Action Recognition System. *Sensors* **2020**, *20*, 4758. [CrossRef]
19. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 3645–3649.
20. Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 815–823.
21. Carreira, J.; Zisserman, A. Quo vadis, action recognition? A new model and the kinetics dataset. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6299–6308.
22. Ji, S.; Xu, W.; Yang, M.; Yu, K. 3D convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *35*, 221–231. [CrossRef]
23. Zhang, J.; Huang, B.; Ye, Z.; Kuang, L.D.; Ning, X. Siamese anchor-free object tracking with multiscale spatial attentions. *Sci. Rep.* **2021**, *11*, 22908. [CrossRef] [PubMed]
24. Liu, S.; Liu, D.; Srivastava, G.; Połap, D.; Woźniak, M. Overview and methods of correlation filter algorithms in object tracking. *Complex Intell. Syst.* **2021**, *7*, 1895–1917. [CrossRef]
25. Luo, F.; Poslad, S.; Bodanese, E. Temporal convolutional networks for multiperson activity recognition using a 2-d lidar. *IEEE Internet Things J.* **2020**, *7*, 7432–7442. [CrossRef]
26. He, Z.; He, H. Unsupervised Multi-Object Detection for Video Surveillance Using Memory-Based Recurrent Attention Networks. *Symmetry* **2018**, *10*, 375. [CrossRef]
27. Meng, L.; Zhao, B.; Chang, B.; Huang, G.; Sun, W.; Tung, F.; Sigal, L. Interpretable spatiotemporal Attention for Video Action Recognition. *arxiv* **2018**, arXiv:1810.04511.

28. Yan, S.; Xiong, Y.; Lin, D. Spatial temporal graph convolutional networks for skeleton-based action recognition. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
29. Sun, L.; Jia, K.; Yeung, D.Y.; Shi, B.E. Human action recognition using factorized spatiotemporal convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4597–4605.
30. An, J.; Cheng, X.; Wang, Q.; Chen, H.; Li, J.; Li, S. Summary of continuous action recognition. *J. Physics: Conf. Ser.* **2020**, *1607*, 012116. [CrossRef]
31. Wang, J.; Wen, X. A Spatiotemporal Attention Convolution Block for Action Recognition. *J. Physics: Conf. Ser.* **2020**, *1651*, 012193. [CrossRef]
32. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. Recent advances in convolutional neural networks. *Pattern Recognit.* **2018**, *77*, 354–377. [CrossRef]
33. Medsker, L.; Jain, L.C. *Recurrent Neural Networks: Design and Applications*; CRC press: Boca Raton, FL, USA, 1999.
34. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *AI Open* **2022**, *3*, 111–132. [CrossRef]
35. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint* **2020**, arXiv:2010.11929.
36. Hochreiter, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertainty, Fuzziness -Knowl.-Based Syst.* **1998**, *6*, 107–116. [CrossRef]
37. Jozefowicz, R.; Zaremba, W.; Sutskever, I. An empirical exploration of recurrent network architectures. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; pp. 2342–2350.
38. Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 2625–2634.
39. Tran, D.; Wang, H.; Torresani, L.; Ray, J.; LeCun, Y.; Paluri, M. A closer look at spatiotemporal convolutions for action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6450–6459.
40. Simonyan, K.; Zisserman, A. Two-stream convolutional networks for action recognition in videos. *Adv. Neural Inf. Process. Syst.* **2014**, *27* , 1–9.
41. Beauchemin, S.S.; Barron, J.L. The computation of optical flow. *ACM Comput. Surv.* **1995**, *27*, 433–466. [CrossRef]
42. Feichtenhofer, C.; Pinz, A.; Zisserman, A. Convolutional two-stream network fusion for video action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1933–1941.
43. Wang, L.; Qiao, Y.; Tang, X. Action recognition with trajectory-pooled deep-convolutional descriptors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4305–4314.
44. Wang, Y.; Long, M.; Wang, J.; Yu, P.S. Spatiotemporal Pyramid Network for Video Action Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
45. Christoph, R.; Pinz, F.A. Spatiotemporal residual networks for video action recognition. *arXiv* **2016**, arXiv:1611.02155.
46. Girdhar, R.; Ramanan, D.; Gupta, A.; Sivic, J.; Russell, B. ActionVLAD: Learning Spatiotemporal Aggregation for Action Classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
47. Crasto, N.; Weinzaepfel, P.; Alahari, K.; Schmid, C. Mars: Motion-augmented rgb stream for action recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7882–7891.
48. Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; Van Gool, L. Temporal segment networks: Towards good practices for deep action recognition. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 20–36.
49. Lin, J.; Gan, C.; Han, S. Tsm: Temporal shift module for efficient video understanding. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 7083–7093.
50. Feichtenhofer, C.; Fan, H.; Malik, J.; He, K. Slowfast networks for video recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6202–6211.
51. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
52. Tenney, I.; Das, D.; Pavlick, E. BERT rediscovers the classical NLP pipeline. *arXiv preprint* **2019**, arXiv:1905.05950.
53. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training. *OpenAI* **2018**. Available online: https://openai.com/research/language-unsupervised (accessed on 25 April 2023).
54. Thorp, H.H. ChatGPT Is Fun, but not an Author, 2023. Available online: https://openai.com/blog/chatgpt (accessed on 25 April 2023).
55. Kalfaoglu, M.E.; Kalkan, S.; Alatan, A.A. Late temporal modeling in 3d cnn architectures with bert for action recognition. In Proceedings of the Computer Vision–ECCV 2020 Workshops: Glasgow, UK, 23–28 August 2020; pp. 731–747.
56. Bertasius, G.; Wang, H.; Torresani, L. Is space-time attention all you need for video understanding? In Proceedings of the ICML, Virtual, 18–24 July 2021; Volume 2, p. 4.

57. Yang, X.; Zhang, C.; Tian, Y. Recognizing actions using depth motion maps-based histograms of oriented gradients. In Proceedings of the 20th ACM International Conference on Multimedia, New York, NY, USA, 29 October–2 November 2012; pp. 1057–1060.
58. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; IEEE: Piscataway, NJ, USA, 2005; Volume 1; pp. 886–893.
59. Noble, W.S. What is a support vector machine? *Nat. Biotechnol.* **2006**, *24*, 1565–1567. [CrossRef]
60. Chen, C.; Liu, K.; Kehtarnavaz, N. Real-time human action recognition based on depth motion maps. *J. -Real-Time Image Process.* **2016**, *12*, 155–163. [CrossRef]
61. Lee, H.; Battle, A.; Raina, R.; Ng, A. Efficient sparse coding algorithms. *Adv. Neural Inf. Process. Syst.* **2006**, *19*, 801–808.
62. Oreifej, O.; Liu, Z. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 716–723.
63. Rahmani, H.; Mahmood, A.; Huynh, D.Q.; Mian, A. Real time action recognition using histograms of depth gradients and random decision forests. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Steamboat Springs, CO, USA, 24–26 March 2014; IEEE: Piscataway, NJ, USA; pp. 626–633.
64. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
65. Satyamurthi, S.; Tian, J.; Chua, M.C.H. Action recognition using multi-directional projected depth motion maps. *J. Ambient. Intell. Humaniz. Comput.* **2018**; 1–7. [CrossRef]
66. He, D.C.; Wang, L. Texture unit, texture spectrum, and texture analysis. *IEEE Trans. Geosci. Remote. Sens.* **1990**, *28*, 509–512.
67. Huang, G.B.; Zhu, Q.Y.; Siew, C.K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. [CrossRef]
68. Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv* **2014**, arXiv:1312.6034.
69. Smilkov, D.; Thorat, N.; Kim, B.; Viégas, F.; Wattenberg, M. Smoothgrad: Removing noise by adding noise. *arXiv* **2017**, arXiv:1706.03825.
70. Sundararajan, M.; Taly, A.; Yan, Q. Axiomatic attribution for deep networks. In Proceedings of the ICML, Sydney, Australia, 6–11 August 2017.
71. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 818–833.
72. Petsiuk, V.; Das, A.; Saenko, K. RISE: Randomized Input Sampling for Explanation of Black-box Models. *arXiv* **2018**, arXiv:1806.07421.
73. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the ICCV, Venice, Italy, 22–29 October 2017.
74. Jiang, P.T.; Zhang, C.B.; Hou, Q.; Cheng, M.M.; Wei, Y. LayerCAM: Exploring hierarchical class activation maps for localization. *IEEE Trans. Image Process.* **2021**, *30*, 5875–5888. [CrossRef]
75. Englebert, A.; Cornu, O.; Vleeschouwer, C.D. Poly-CAM: High resolution class activation map for convolutional neural networks. *arXiv* **2022**, arXiv:2204.13359v2.
76. Jalwana, M.A.; Akhtar, N.; Bennamoun, M.; Mian, A. CAMERAS: Enhanced resolution and sanity preserving class activation mapping for image saliency. In Proceedings of the CVPR, Virtual, 19–25 June 2021.
77. Wang, H.; Wang, Z.; Du, M.; Yang, F.; Zhang, Z.; Ding, S.; Mardziel, P.; Hu, X. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In Proceedings of the CVPR Worshop on TCV, Seattle, WA, USA, 14–19 June 2020.
78. Soomro, K.; Zamir, A.R.; Shah, M. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint* **2012**, arXiv:1212.0402.
79. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; Serre, T. HMDB: a large video database for human motion recognition. In Proceedings of the 2011 International Conference on Computer Vision, IEEE, Barcelona, Spain, 6–13 June 2011; pp. 2556–2563.
80. Contributors, M. OpenMMLab's Next Generation Video Understanding Toolbox and Benchmark. 2020. Available online: https://github.com/open-mmlab/mmaction2 (accessed on 25 April 2023). .
81. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Curran Associates, Inc.: Red Hook, NY, USA, 2019; pp. 8024–8035.
82. Reed, R. Pruning algorithms-a survey. *IEEE Trans. Neural Networks* **1993**, *4*, 740–747. [CrossRef]
83. Gray, R.M.; Neuhoff, D.L. Quantization. *IEEE Trans. Inf. Theory* **1998**, *44*, 2325–2383. [CrossRef]
84. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge distillation: A survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. [CrossRef]