CyberExcellence
By CyberWal

InfortechDay (29/05/2024)

UMONS
Université de Mons

Topic :
Deep learning based vulnerability detection in source code

Researcher : **Dyna Soumhane Ouchebara**

Working on CyberExcellence project from Feb 2024 to Dec 2025

PHD student at UMONS (2024-2027)

Supervisor : **Pr. Stéphane Dupont**

Vulnerability detection in source code

Vulnerability ?

A **flaw** in the code, which **when exploited**, can cause a **violation of the security** policy

Vulnerability detection ?

Detect vulnerabilities in the code **at early stages of development** in order to fix them (preventive approach)

Vulnerability detection in source code

Example : SQL injection

- Code :

```
$user_id = $_GET['user_id’];
$sql_query = "SELECT * FROM users WHERE id = '$user_id’ ";
$result = mysqli_query($connexion, $sql_query);
```

Vulnerability

- Attack :

http://website.com/?**user_id**=' **OR '1'=‘1**

- Consequence :

SELECT * FROM users **WHERE id = '' OR '1'=‘1’** → **always “True”**

→ **Select all users data from database**

Conventional techniques

Automated, ***but…***

Manual inspection

+ Reliable

− Time+Effort consuming

Static analysis

+ -Scalable
-Early stage

− -Can’t spot runtime vulnerabilities

Dynamic analysis

+ Spot runtime vulnerabilities

− Resource intensive

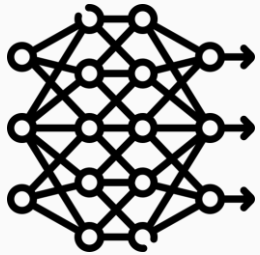
Deep learning

Labeled Data

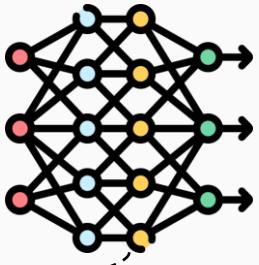
Code	Label
class Person {…}	Vulnerable
def calculate {…}	Non vulnerable
…	…

Training

Neural network



Trained model

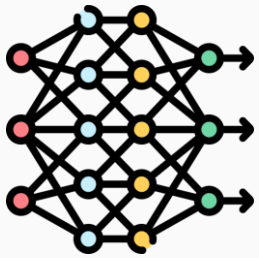


New data

Code	Label
def get_url {…}	?

Prediction

Trained model



Label

Label
Vulnerable

(2016) **Deep Learning** : Code as Natural Language *(Text)*

VulDeePecker: A Deep Learning-Based System for Vulnerability Detection

(5 Jan 2018)

Zhen Li*†, Deqing Zou*‡♯, Shouhuai Xu§, Xinyu Ou*, Hai Jin*,
Sujuan Wang*, Zhijun Deng* and Yuyi Zhong*

Tokens → Word2vec → Recurrent neural networks (RNNs)

(Representation) *(Embedding)* *(DL model architecture)*

(2019) **Deep Learning** : Code as a Structure *(Graph/Tree)*

Devign: Effective Vulnerability Identification by Learning Comprehensive Program Semantics via Graph Neural Networks

(8 Dec 2019)

Yaqin Zhou[1], Shangqing Liu[1,*], Jingkai Siow[1], Xiaoning Du[1,*], and Yang Liu[1]

AST → Graph embeddding → Graph neural networks (GNNs)

(Representation) *(Embedding)* *(DL model architecture)*

(2021) **Deep Learning** : Code as Text again, **but**

LineVul: A Transformer-based Line-Level Vulnerability Prediction

(17 Oct 2022)

Michael Fu

Chakkrit Tantithamthavorn*

Tokens → Transformer-based Pre-trained model

(Representation) *(Embedding + DL model architecture)*

Transformer based pre-trained models

Phase 1 : Pre-training

Transformer architecture

Massive unlabeled data

Generic tasks
(Next token prediction …)

Pre-training
(Self-supervised learning)

Pre-trained model

Phase 2 : Fine-tuning

Pre-trained model

Domain-related labled data

Specific task
(Classification…)

Fine-tuning
(Supervised learning)

Fine-tuned model

Challenges

1. Erronous datasets

Dataset	Correct Label
CodeXGLUE [8, 19]	24.0 %
VulnPatchPairs [20]	36.0 %
BigVul [9]	25.0* %
CrossVul [11]	47.8* %
CVEFixes [10]	51.7* %

2. Generalization capability

3. Detection granularity

4. Explainability

Research directions

Face the challenges

- Better datasets
- Better generalization capability
- Best detection granularity
- Explainability of the models

Better representation and/or embedding of code

Better DL architectures

More suited pre-training tasks

Better fine-tuning strategies

Final note

Deep learning based vulnerability detection

Software testing

Cybersecurity

Artificial intelligence

Thank you for your attention !

References :

1. Li Z, Zou D, Xu S, Ou X, Jin H, Wang S, Deng Z, Zhong Y (2018) VulDeePecker: a deep learning-based system for vulnerability detection
2. Zhou Y, Liu S, Siow J, Du X, Liu Y (2019) Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks
3. Michael Fu and Chakkrit Tantithamthavorn (2022) Linevul: A transformer-based line-level vulnerability prediction