Vulnerability Detection via Topological Analysis of Attention Maps

Snopov P.[1], Golubinsky A.N.[1]

Institute for Information Transmission Problems of Russian Academy of Sciences
snopov@iitp.ru

Abstract. Recently, DL-based approaches to the vulnerability detection task became popular. They show promising results, outperforming traditional static code analysis tools.
In this work, we test a novel approach to the vulnerability detection using topological data analysis of the attention matrices of BERT model. We show that

Keywords: Vulnerability detection, Persistent homology, Large language models

1 Introduction

The problem of the source code vulnerability detection becomes an important problem in the field of software engineering nowadays. This is a complex task involving the analysis of both syntax and semantics of the source code. The static analysis methods were dominating during most of the time in the vulnerability detection. The static tools tend to rely heavily only on the syntax of the program, thus loosing the potential information that is hidden in the semantics. Therefore such approaches suffer from high rates of false positives.

But recently the different ML-based and DL-based methods started to appear [6, 9]. Most DL-based solutions provide the methods based on graph neural networks (GNNs) [10, 4, 11]. Besides GNN-based models, there exists LLM-based approaches [7, 8, 1]. Both GNN-based and LLM-based methods are targeted to learn the semantics together with the syntax of the program. This approaches show the promising results, in particular such methods demonstrate low rates of false positives as well as false negatives [2].

However some studies show that the generalization of DL-based methods is very poor, and that there are no existing models that would perform well in real-world settings [5]. This situation shows that there is still no silver bullet for the vulnerability detection task.

Our work demonstates a novel approach to the vulnerability detection problem. As mentioned above, the superiority of DL-based models over static ones are potentially due to the use of the semantic information of the program. This semantic information is captured somehow with neural networks during training or fine-tuning. Moreover, LLM-based models should capture this information even wihout any fine-tune.

On the hand, as well as with the syntax of the source code, the semantics can also be represented as a tree or a graph. Such representation opens the doors for the use of topological methods.

In this work, we leverage the BERT model, pretrained on the source code, for capturing the semantic information of the programs, and we also use the tools from topological data analysis [**something-1**] to retrieve the relations in this semantics. This is an attempt to analyse the interpret the attention matrices via topological instruments.

Our work is inspired by the works of Laida Kushareva et. al. [3], in which the authors apply the topological methods for the artificial text detection, and show that the topology of the semantics captured by a LLM model (such as BERT) provides enough information for the successful classification, beating neural baselines and performing on par with a fully fine-tuned BERT model, but being more robust towards the unseen data.

2 Background

2.1 BERT Model

Description of the chosen model (RoBERTa-small).

2.2 Attention Graph

An {attention graph} is a weighted graph representation of an attention matrix W^{attn}, in which the vertices represent the tokens and the edges connect a pair of tokens if the corresponding weight is higher than the predefined threshold value.

This threshold value is set to distinguish weak and strong relations between tokens. But the choice of the threshold value seems to be a very hard problem. Moreover, varying the threshold, the graph structure may change dramatically.

Hopefully, TDA methods can extract the properties of graph structure, without specifying the concrete value of the threshold.

2.3 Topological Data Analysis

Topological data analysis is a young and rapidly evolving field that applies some of the powerful methods from algebraic topology to data science. There are already exist a plethora of good tutorials and surveys for non-mathematicians [**something-1**, **something-2**, **something-3**] as well as for those who has a mathematical background [**something-1**, **something-2**].

The main instrument in topological data analysis, *persistent homology*, allows one to track the changes in the topological structure for different objects, such as point clouds, scalar functions, images, weighted graphs [**something**].

Specifically, in our work, we are given with a set of tokens V and an attention matrix W. Next, we build a family of attention graphs, which is being indexed by increasing threshold values This family is called a *filtration*, and it is a crucial object in TDA.

When such seqeunce of graphs is given, the persistent homology in dimension 0 and 1 are computed. Dimension 0 illustrates if there are connected components or clusters in our data, and dimension 1 shows if our data possesses cycles or «loops». These calculations provide us with the *persistence diagram*, which can be further used to derive specific topological features, such as the amount of connected components or the amount of cycles (such features are also called the *Betti numbers*).

3 Topological Features of the Attention Graphs

For each code sample, we calculate the persistent homology in dimension 0 and 1, obtaining the persistence diagram on each attention head of the BERT model. We compute the following features in each dimension from the diagrams:

- The mean lifespan of points on the diagram
- The variance lifespan of points on the diagram
- The max lifespan of points on the diagram
- The overall number of points on the diagram
- The persistence entropy

4 Experiments

5 Discussion

6 Consclusion

This paper introduces a novel approach for the vulnerability detection task based on TDA.

Bibliography

[1] Daya Guo et al. *GraphCodeBERT: Pre-training Code Representations with Data Flow.* arXiv:2009.08366 [cs]. Sept. 2021. DOI: `10.48550/arXiv.2009.08366`. URL: `http://arxiv.org/abs/2009.08366` (visited on 06/03/2024).

[2] Evangelos Katsadouros and Charalampos Patrikakis. "A Survey on Vulnerability Prediction using GNNs". en. In: *Proceedings of the 26th Pan-Hellenic Conference on Informatics.* Athens Greece: ACM, Nov. 2022, pp. 38–43. ISBN: 978-1-4503-9854-1. DOI: `10.1145/3575879.3575964`. URL: `https://dl.acm.org/doi/10.1145/3575879.3575964` (visited on 04/24/2024).

[3] Laida Kushnareva et al. "Artificial Text Detection via Examining the Topology of Attention Maps". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.* arXiv:2109.04825 [cs, math]. 2021, pp. 635–649. DOI: `10.18653/v1/2021.emnlp-main.50`. URL: `http://arxiv.org/abs/2109.04825` (visited on 05/16/2024).

[4] Van-Anh Nguyen et al. *ReGVD: Revisiting Graph Neural Networks for Vulnerability Detection.* arXiv:2110.07317 [cs]. Feb. 2022. URL: http://arxiv.org/abs/2110.07317 (visited on 04/24/2024).

[5] Saikat Chakraborty et al. "Deep Learning Based Vulnerability Detection: Are We There Yet? | IEEE Journals & Magazine | IEEE Xplore". In: (Sept. 2022). URL: https://ieeexplore.ieee.org/abstract/document/9448435 (visited on 06/03/2024).

[6] Tushar Sharma et al. "A survey on machine learning techniques applied to source code". In: *Journal of Systems and Software* 209 (Mar. 2024), p. 111934. ISSN: 0164-1212. DOI: 10.1016/j.jss.2023.111934. URL: https://www.sciencedirect.com/science/article/pii/S0164121223003291 (visited on 06/03/2024).

[7] Alexey Shestov et al. *Finetuning Large Language Models for Vulnerability Detection.* arXiv:2401.17010 [cs] version: 4. Mar. 2024. DOI: 10.48550/arXiv.2401.17010. URL: http://arxiv.org/abs/2401.17010 (visited on 04/24/2024).

[8] Benjamin Steenhoek et al. *A Comprehensive Study of the Capabilities of Large Language Models for Vulnerability Detection.* arXiv:2403.17218 [cs]. Mar. 2024. DOI: 10.48550/arXiv.2403.17218. URL: http://arxiv.org/abs/2403.17218 (visited on 04/24/2024).

[9] Benjamin Steenhoek et al. *An Empirical Study of Deep Learning Models for Vulnerability Detection.* arXiv:2212.08109 [cs]. Feb. 2023. DOI: 10.48550/arXiv.2212.08109. URL: http://arxiv.org/abs/2212.08109 (visited on 04/28/2024).

[10] Xin-Cheng Wen et al. *Vulnerability Detection with Graph Simplification and Enhanced Graph Representation Learning.* arXiv:2302.04675 [cs]. Feb. 2023. DOI: 10.48550/arXiv.2302.04675. URL: http://arxiv.org/abs/2302.04675 (visited on 04/24/2024).

[11] Yaqin Zhou et al. "Devign: Effective Vulnerability Identification by Learning Comprehensive Program Semantics via Graph Neural Networks". In: *Advances in Neural Information Processing Systems.* Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/49265d2447bc3bbfe9e76306ce40a31f-Abstract.html (visited on 06/03/2024).