Detection of Prompt Injection Attacks Using a Hierarchical Approach

Sujin Lee
School of Computer Science and Engineering
Yeungnam University
Gyeongsan, Republic of Korea
sujinlee@yu.ac.kr

Wooguil Pak
School of Computer Science and Engineering
Yeungnam University
Gyeongsan, Republic of Korea
wooguilpak@yu.ac.kr

Abstract— **Large Language Models are vulnerable to prompt injection attacks. Although various detection techniques have been proposed, approaches relying on external classifiers are limited by high computational overhead. Accordingly, this paper proposes Multi-layer Hierarchical Detection, a fast and lightweight method that utilizes hidden states from each Transformer layer to perform classification. This method reduces the average number of classification layers required for detection by 75.3% to 88.5% compared to the baseline method, while maintaining performance within a marginal F1-score difference of 0.5%. Consequently, the proposed method reduces detection latency and provides an efficient solution.**

***Keywords*— Artificial Intelligence, Large Language Models, Prompt Injection Detection.**

I. INTRODUCTION

Large Language Models (LLMs) are seeing widespread adoption across diverse fields, owing to their outstanding performance [1]. This expansion, however, increases their exposure to prompt injection attacks that manipulate prompts to intentionally bypass a model's guidelines [2, 3, 4]. Given that such attacks can precipitate severe consequences, including the generation of unintended content and the execution of harmful operations, robust and practical prompt injection detection is essential.

Existing prompt injection detection techniques have employed independent Transformer models [5] as external classifiers [6, 7]. This approach, however, inherently incurs high computational overhead. Alternative approaches utilize the model's internal representations, such as attention patterns [8] or hidden states [9], to detect prompt injection. While these approaches offer the advantage of significantly reducing detection overhead, several limitations persist, such as inconsistent performance across datasets or a failure to fully leverage the hierarchical characteristics of multi-layer structures.

To address these limitations, we propose a hierarchical detection technique called *Multi-layer Hierarchical Detection (MHD)*. This method achieves fast and lightweight classification by identifying each input at an appropriately early layer. By processing easily detectable inputs in lower layers and passing those that are difficult to classify to higher layers, this approach significantly reduces average detection latency and computational overhead.

The remainder of this paper is structured as follows: Section II reviews existing research; Section III presents the proposed methodology; Section IV details the experimental results; and Section V discusses the limitations and concludes the study.

II. RELATED WORK

Research on prompt injection detection has been actively conducted in various directions. One line of research employs an external classifier that operates independently of the service model—the primary model responsible for generating responses to users. In contrast to this external approach, another line of work detects attacks by analyzing the internal representations of the service model itself. This section reviews key studies from both methodologies that serve as baselines for our comparison.

A. *Prompt injection detection using external classifiers*

Deberta-v3-base-prompt-injection-v2 [6], developed by ProtectAI, is a model specifically fine-tuned for prompt injection detection based on DeBERTa [10]. Similarly, Prompt Guard [7], developed by Meta, is a specialized detection model built upon DeBERTa [10]. Although these models are on a million-parameter scale, they have the significant disadvantage of incurring high computational overhead, as they require an independent inference pass for every input.

B. *Prompt injection detection using internal representations*

Attention Tracker [8] is a technique that detects prompt injection by utilizing the service model's internal attention scores. It is based on the observation that prompt injection causes a distraction effect, shifting attention from the original system prompt to the injected user prompt. A key advantage of this method is that it is training-free. However, this method has limitations: while it achieves high detection performance on specific datasets, its performance varies considerably across different domains, indicating high domain-dependence.

Layer Enhanced Classification (LEC) [9] is a technique for detecting prompt injection by utilizing hidden states extracted from a single intermediate layer. It involves training a penalized logistic regression classifier on these representations. While LEC is lightweight and demonstrates strong performance, it is structurally constrained, as it is designed to classify all inputs using the fixed intermediate layer.

III. Method

This section details the core concepts and structural design of the proposed MHD. We first provide motivation by analyzing the inefficiencies inherent in existing detection methods, followed by a comprehensive description of the overall architecture and its detailed mechanisms.

A. Motivation

The classification results of LEC, shown in Fig. 1, reveal a critical inefficiency. As illustrated, most data is accurately classified at lower layers (e.g., Layer 1 achieves 96.38% F1-score). Consequently, passing all data through to a fixed deeper layer results in unnecessary computations. To address this, we propose a hierarchical approach that processes data that can be accurately classified at lower layers and only passes data requiring further analysis to higher layers.

B. Proposed Architecture

To address the structural inefficiency of existing methods, we propose MHD. The overall architecture is illustrated in Fig. 2. As data passes through each layer, MHD classifies what can be classified at that point, while data requiring further analysis is passed to the next layer. Therefore, like LEC, final classification occurs at an intermediate layer (MHD last layer) using a binary classifier, but unlike LEC, we attach a ternary classifier to all preceding layers before it.

C. Detailed Structure

The MHD utilizes two types of layer-specific classifiers: binary and ternary. The binary classifiers (Safe, Harmful) serve two distinct purposes. First, they are used to create the ternary classification dataset. That is, a binary classifier is trained for each layer, and any data samples misclassified by this classifier are re-labeled as ‘Uncertain’.

These ternary datasets (Safe, Harmful, Uncertain) are then used to train the ternary classifier for that layer. The ternary classifiers are applied at each layer before the *MHD last layer*. During inference, inputs classified as ‘Safe’ or ‘Harmful’ stop the MHD process immediately, while those classified as ‘Uncertain’ are forwarded to the next layer.

When data reaches the *MHD last layer*, the binary classifier’s second role is activated. Inputs that passed through all preceding ternary classifiers as ‘Uncertain’ are finally categorized by the binary classifier—the same one used in the ternary dataset creation step—into a definitive ‘Safe’ or ‘Harmful’ status.

IV. Experiments

In this section, we present the experimental evaluation of MHD. To assess its effectiveness and efficiency, we compare its performance against several existing approaches: external classifier-based methods (deberta-v3-base-prompt-injection-v2 [6] and Prompt Guard [7]), and the internal representation-based methods (Attention Tracker [8] and LEC [9]). Our experiments were conducted using the SPML dataset [11] across two different service models serving as feature extractors. The following subsections detail the experimental setup and discuss the comparative results.

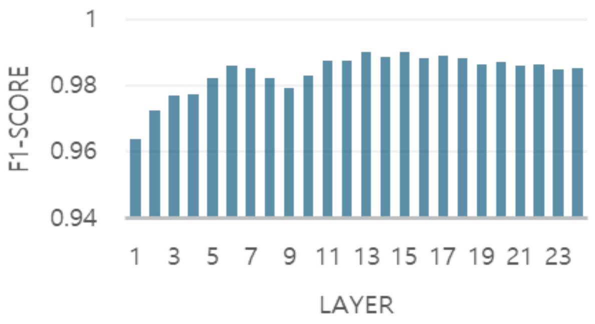

Figure 1. LEC Classification results.

A. Experimental settings

1) Feature extraction model: While the original LEC study [9] examined both service model and detection-specific fine-tuned models, we utilize only the hidden states generated by the service model itself. This approach prioritizes resource efficiency and ensures seamless integration with the existing service model. Specifically, we employed two pre-trained models as feature extractors: Qwen2.5-0.5B-Instruct (24 layers, 896-dimensional hidden states) [12] and Qwen2.5-14B-Instruct (48 layers, 5,120-dimensional hidden states) [12].

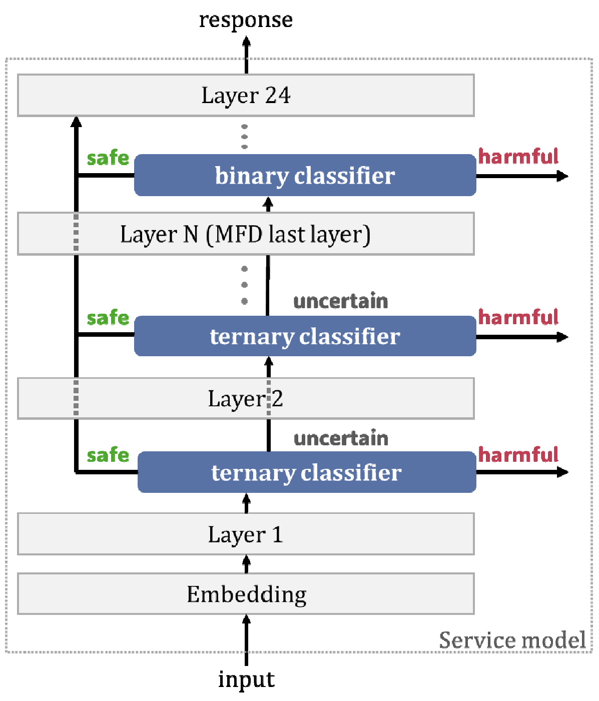

Figure 2. MHD Structure.

TABLE I. SPML DATASET EVALUATION RESULTS

Feature Extraction Model	Method	Accuracy (%)	F1 (%)	Maximum Classification Layer (1~)	Average Classification Layer (1~)
Qwen2.5-0.5B	MHD	**99**	**98.99**	5	**3.70**
	LEC	98.53	98.52	15	15
	Attention-tracker	80.76	76.53	15	15
Qwen2.5-14B	MHD	98.59	98.57	3	**2.53**
	LEC	**98.88**	**98.88**	22	22
	Attention-tracker	86.29	84.22	48	48
-	Deberta-v3-base-prompt-injection-v2	94.47	94.7	-	-
-	Prompt Guard2	77.12	70.33	-	-

2) Training: We utilized the SPML dataset [11], which consists of system prompt-user prompt pairs. A sample is labeled as 1 (harmful) if a user prompt attempts to elicit an output that violates the system prompt's instructions, regardless of the user's actual malicious intent. The dataset was split into 3,300 training, 1,700 validation, and 1,700 test samples, with the class labels uniformly distributed. For training the classifiers, we employed a learning rate of 0.05 and an early stopping patience of 20 epochs.

B. Experimental results

The evaluation results for the SPML dataset are presented in Table I. MHD achieved significantly higher performance than external baseline methods and the Attention Tracker. When compared to LEC, MHD demonstrated comparable performance levels. As shown in Table I, MHD showed slightly higher performance when using Qwen2.5-0.5B-Instruct, while LEC showed a marginal advantage with Qwen2.5-14B-Instruct. Overall, the F1-score difference between the two methods remained within a narrow range of ±0.5%.

Critically, MHD improved detection efficiency by reducing the average number of classification layers by 75.3% to 88.5% compared to LEC, all while maintaining equivalent performance. This reduction translates into a significant decrease in overall detection latency. Furthermore, early detection at lower layers enables the rapid blocking of harmful prompts and minimizes redundant inference computations. These characteristics ensure model safety while simultaneously enhancing operational efficiency.

V. CONCLUSION

In this paper, we introduced MHD to address the inefficiency of existing detection methods that process every input through a fixed single layer. While our study proposes a fast and lightweight multi-layer detection approach, several limitations remain for future work. First, MHD requires a dedicated classifier for each layer, resulting in a larger total number of trainable parameters compared to single-classifier methods like LEC. Second, inputs that are not classified at early layers must undergo sequential classification up to the *MHD last layer*. In such worst-case scenarios, computational overhead may slightly increase compared to a single-pass method. These limitations highlight the need for further optimization strategies in future research.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (No. NRF-RS-2025-24683865).

REFERENCES

[1] W. X. Zhao *et al.*, "A survey of large language models," arXiv preprint, arXiv:2303.18223, Mar 2023.

[2] Y. Liu *et al.*, "Prompt injection attack against LLM-integrated applications," arXiv preprint, arXiv:2306.05499, Jun 2023.

[3] Lakera, "Prompt Injection Attacks Handbook," Lakera AI, 2024. [Online]. Avaliable: https://www.lakera.ai/ai-security-guides/prompt-injection-attacks-handbook

[4] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, Yang Zhang, "Do Anything Now: Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models," arXiv preprint, arXiv:2308.03825, Aug 2023.

[5] A. Vaswani *et al.*, "Attention is all you need," Advances in Neural Information Processing Systems, vol 30, 2017.

[6] ProtectAI.com, "Fine-Tuned DeBERTa-v3-base for Prompt Injection Detection," HuggingFace, 2024. [Online]. Available: https://huggingface.co/ProtectAI/deberta-v3-base-prompt-injection-v2

[7] Meta, "Llama-Prompt-Guard-2-86M," HuggingFace, 2025. [Online]. Available: https://huggingface.co/meta-llama/Llama-Prompt-Guard-2-86M

[8] K.-H. Hung *et al.*, "Attention tracker: Detecting prompt injection attacks in LLMs," Findings of the Association for Computational Linguistics: NAACL 2025, pp. 2309–2322, Apr 2025.

[9] M. Sawtell, T. Masterman, S. Besen, and J. Brown, "Lightweight safety classification using pruned language models," arXiv preprint, arXiv:2412.13435, Dec 2024.

[10] Pengcheng He, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, "DeBERTa: Decoding-enhanced BERT with Disentangled Attention," ArXiv:2006.03654, Jun 2020.

[11] R. K. Sharma, V. Gupta, and D. Grossman, "SPML: A DSL for defending language models against prompt attacks," arXiv preprint, arXiv:2402.11755, Feb 2024.

[12] Qwen, "Qwen2.5 technical report," arXiv preprint, arXiv:2412.15115, Dec 20 .