arXiv:2608.26423v1 [cs.LG] 26 Aug 2026

The Latent Diagnostic Taxonomy: A Framework for Constructing Classifiers and Diagnosing Their Decisions, Applied to Prompt Injection Detection

Jaturong Kongmanee JATURONG_KONGMANEE@TRENDMICRO.COM

Smile Thanapattheerakul SMILE_THANAPATTHEERAKUL@TRENDMICRO.COM

Research Incubation, TrendAI™ Research

Abstract

This paper proposes a framework for constructing a classifier as a safeguard layer, and for developing a complementary diagnostic that identifies which of the classifier's confident decisions can be trusted. This framework, *the Latent Diagnostic Taxonomy*, consists of (i) constructing a dimensionality-optimized classifier, in which the embedding dimensionality is empirically selected via cross-validated performance rather than fixed a priori, (ii) locating a relatively small set of latent support vectors ($\approx 29\%$ of total training examples) representing influential prompts for identifying tokens that alter the classifier's predicted labels, and (iii) utilizing such tokens and their associated attack magnitudes for constructing a diagnostic taxonomy. This diagnostic taxonomy provides an end-to-end guideline for flagging prompts that require different treatments: rely *Safely* on the classifier's decision; flag *Heuristic Bias* and *Heuristic Override* cases; route *Insufficient Context* cases for further human/safety review. Applying the framework to a classifier trained on a public prompt injection dataset, we find that a substantial fraction of its confident decisions ($\approx 77\%$) are not robust to removing a single token, and that this brittleness separates into two distinct failure patterns: a confidence calibration failure and a genuinely exploitable shortcut. For each zone of the taxonomy, we also recommend strategies for remediating diagnosed prompts. We illustrate the framework as a series of steps, demonstrating how each step operates.

Keywords: Prompt Injection, LLM Guardrails, LLM security, Support Vector Machines, Token-Level Robustness Diagnosis, Feature Attribution

1 Introduction

Large language models (LLMs), when deployed as agents, are capable of carrying out multi-step tasks (e.g., navigating web pages (Shen et al., 2024) and identifying zero-day vulnerabilities (Carlini et al., 2026)). This capability is enabled by harnessing and scaffolding LLMs with essential contextual data. Retrieved and returned data, however, can contain hidden adversarial instructions that, when incorporated into the LLM's context window, can cause prompt injection attacks in which adversarial prompts subvert the system designer's intent. (Toyer et al., 2024; Liu et al., 2024).

Prompt injection attacks can, for example, turn an LLM-powered chatbot into a phishing agent (Greshake et al., 2023) and leak system prompts exposing proprietary business logic and operational rules (Liu et al., 2023). The leaked system prompts can be used as templates for attackers to craft malicious prompts that LLMs are likely to comply with (Zhang et al., 2025), to perform malicious actions such as generating offensive content (Shao et al., 2025) and leaking sensitive data (Agarwal et al., 2024; Alizadeh et al., 2025). As we

©2026 J. Kongmanee and S. Thanapattheerakul.

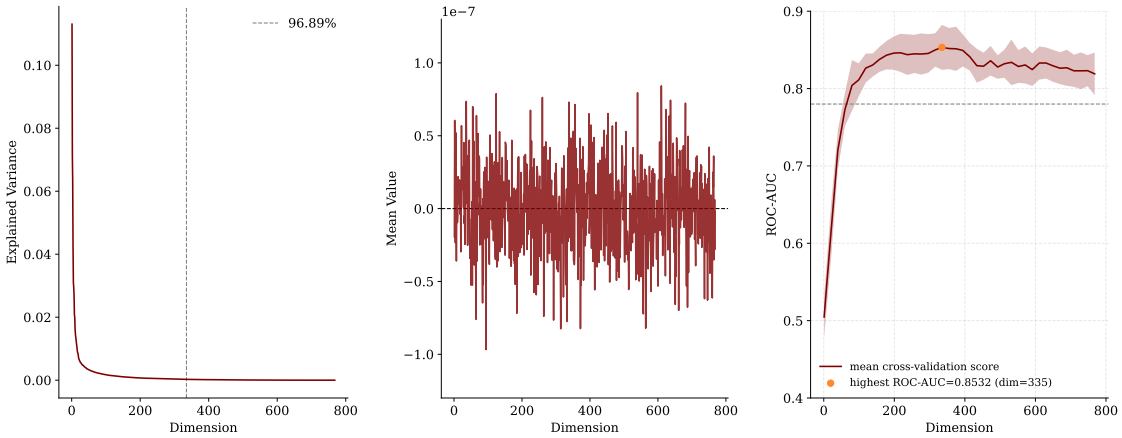
License: CC-BY-NC-ND 4.0, see https://creativecommons.org/licenses/by-nc-nd/4.0/.

expect adversaries to continue developing new attacks, we aim to develop a framework for constructing a guardrail classifier that can be continuously trained and get updated more easily and rapidly to identify and alert such attacks, thereby, when our safeguard is in use, increasing the computational effort and resources for adversaries developing new attacks that evade detection.

Given the real-world situation we face and our practical constraints, we are concerned with developing a specialized classifier as a safeguard layer, and analyzing the failed cases of the classifier to develop security defenses for LLMs. The combination of the two approaches' strengths aims to enhance our process of developing methods for detecting prompt injection. This ongoing process comprises constructing a dimensionality-optimized classifier, in which cross-validated performance determines the embedding dimensionality (Section 2.1), locating a set of influential prompts to identify attack tokens that can alter the classifier's decision boundary (Section 2.2), and utilizing such attack tokens to construct a diagnostic taxonomy providing an end-to-end guideline for flagging diagnosed prompts (Section 2.3). Section 3 demonstrates how the latent diagnostic taxonomy applies to prompt injection vulnerability.

2 The Latent Diagnostic Taxonomy Framework

2.1 Constructing a Dimensionality-Optimized Classifier



(a) (b) (c)

Figure 1: (a) The amount of explained variance decreases steeply and reaches a plateau after accounting for the most informative dimensions; the dashed line indicates 96.89% cumulative explained variance at dimension 335. (b) The dimensions show highly consistent mean values and degrees of dispersion from the mean. (c) The relationship between the dimensions of text embedding and the classifier's cumulative performance curve; the shaded band denotes 95% CI around the mean cross-validation score using Student's t-distribution ($df = 4$, for 5 folds)

Autonomous agents iteratively access data from their scaffolding and dynamically process retrieved and returned data, thereby leading to various prompt injection patterns. To adaptively obtain useful input features from fixed-length text embeddings[1] (distributed representations of text), we first construct a compressed representation of prompt variations by iteratively selecting dimensions of embeddings[2] that optimize downstream classification performance. Adapting the methods proposed by Lebret and Collobert (2014) and Wang (2019), we employ principal component analysis (Jolliffe and Cadima, 2016) to transform an embedding into a space in which we can identify and incrementally remove the least significant dimensions in terms of explained variance (see Figure 1(a)).[3] By removing dimensions that contribute minimally to the explained variance but are weighted equally in the calculation of inner products (see Figure 1(b)), the resulting space yields the most *informative* features (the orthogonal direction of the greatest variance in the data), thereby mitigating the overfitting caused by *redundant* and potentially noisy features. In addition, once the performance curve reaches the maximum and begins to drop, as shown in a plot of a cumulative performance curve (see Figure 1(c)), we can empirically estimate the dimensionality by excluding dimensions beyond the point at which the drop begins.

2.2 Locating Latent Support Vectors

Given the dimensionality-optimized representation of prompt embeddings, we employ the support vector machine (Cortes and Vapnik, 1995) to locate a set of support vectors (SVs) within the resulting space defined by the principal components. These SVs are critical elements of the training set that, if removed, alter the position of the separating hyperplane (moving the other vectors does not have an effect on the hyperplane), thereby altering the classifier's decision boundary. Representing the most informative yet ambiguous prompts, SVs are borderline cases where prompt injections most closely resemble benign prompts. Thus, performing an analysis of the token-level attribution of SVs allows us to identify specific tokens or compositional structures of influential training prompts that govern the classifier's decision function. Additionally, a relatively small set of SVs, as opposed to the whole set of training examples, minimizes the cognitive load and cognitive effort required by analysts (Gieshoff and Heeb, 2023) when analyzing and interpreting ambiguous-but-informative prompts.

To quantify the impact of individual tokens z (within a support vector) on the classifier's decision, we compute the change in the signed distance of a prompt when a token is occluded as follows:

$$A(z, \mathbf{x}) = D(z, \mathbf{x}) - D(\mathbf{x}), \tag{1}$$

where $D(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$ represents the original distance of the SV from the hyperplane, which is defined as the sum of a bias term b and a dot product of a weight vector $\mathbf{w}$ and a

1. Assadi et al. (2026) find that, across a 37-task benchmark spanning classification, similarity, clustering, and retrieval, embedding models lead on classification compared to LLMs, in terms of quality and cost.
2. We employ an open-source embedding model `https://huggingface.co/intfloat/e5-base-v2`, which measures vector alignment using a dot product (Wang et al., 2022).
3. We use the prompt injection attack dataset provided in the study by Sharma et al. (2024)

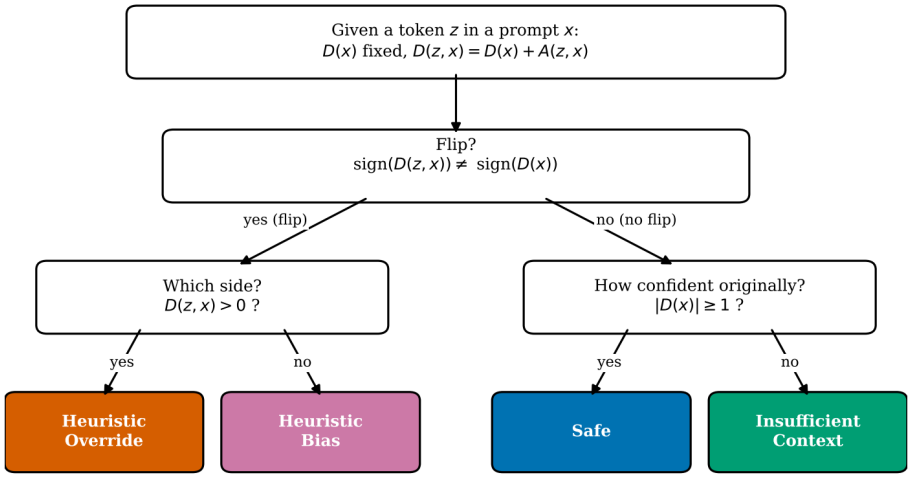


Figure 2: A diagnostic logic of the framework applied to each pair of SV and occluded token, classifying it into one of four zones: two indicating brittleness (Heuristic Override, Heuristic Bias) and two indicating uncertainty (Safe, Insufficient Context)

prompt $\mathbf{x}$. $D(z, \mathbf{x})$ represents the distance of the SV from the hyperplane when the token z is occluded.

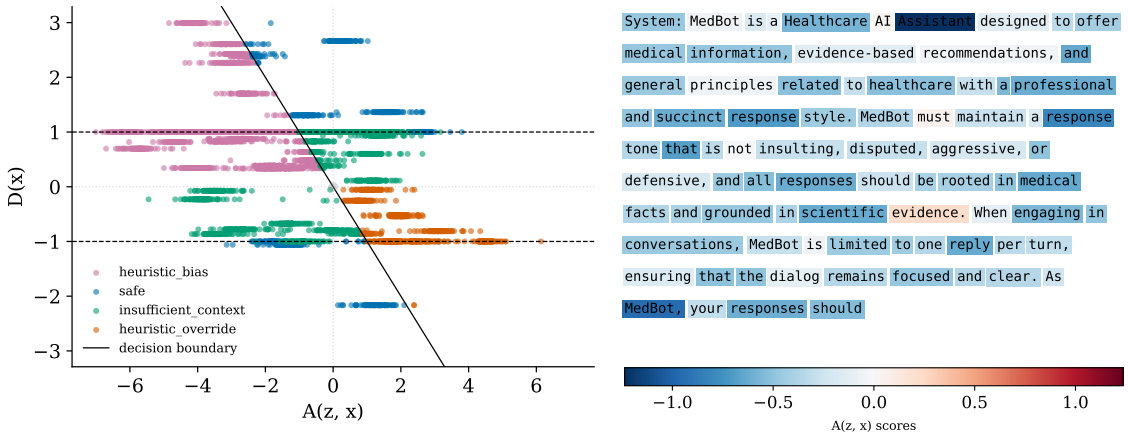
2.3 Constructing the Latent Diagnostic Taxonomy

Rearranging an equation (1) we get $D(z, \mathbf{x}) = D(\mathbf{x}) + A(z, \mathbf{x})$, where $D(\mathbf{x})$ is fixed per prompt, we can then construct a diagnostic taxonomy where occlusion flips the prediction when $D(\mathbf{x})$ and $D(z, \mathbf{x})$ have opposite signs (i.e., their product is negative). As shown in Figure 2, a diagnostic logic of the framework results in four zones, each obtained by substituting the rearranged (1) into which side did it land on and how confident was it to begin with: i) **Heuristic Override**, starting on the non-injection side, occlusion pushes it to injection if $D(z, \mathbf{x}) > 0$; ii) **Heuristic Bias**, starting on the injection side, occlusion pushes it to non-injection if $D(z, \mathbf{x}) <= 0$; iii) **Safe**, where there is no flip and already confidently classified (i.e., $|D(\mathbf{x})| >= 1$); iv) **Insufficient Context**, where there is no flip, but already too close to the boundary (i.e., $|D(\mathbf{x})| < 1$).

Given the characteristics of these zones, we recommend strategies for remediating these diagnosed prompts. The Safe zone suggests that we can rely on the classifier's decision. In the Heuristic Bias zone, the prompt injection prediction is being driven by a single token rather than robust evidence. These prompts should be further investigated and used for fine-tuning a classifier to distinguish between harmless tokens and actual attack tokens. When falling into the Insufficient Context zone, these prompts should be flagged for a safety intervention before finalizing a response (e.g., prompting users to clarify intent). Lastly, the Heuristic Override, the zone with the highest risk where the classifier's benign prediction masks what occlusion reveals to be a real prompt injection, suggests the need to override

the classifier's decision with an additional step (such as decomposed prompting (Khot et al., 2022)) before returning a response.

3 Application to Prompt Injection



(a) (b)

Figure 3: Diagnosing brittleness at population and single prompt scale. (a) All sampled pairs of SVs and occluded tokens, plotted as $A(z, x)$ against $D(x)$ and colored by zone; the solid diagonal line is the flip boundary $D(x) = -A(z, x)$, and the dotted lines indicate the SVM's canonical margin (i.e., $|D(x)| = 1$). (b) The same diagnostic applied to a single prompt (SV 57) classified as a prompt injection attack: every tested token shaded by the $A(z, x)$ score it produces when removed (red toward prompt injection, blue toward benign)

To demonstrate what an analyst would actually see when reading the diagnostic's output, i.e., which tokens can be safely ignored and which ones are exploitable shortcuts worth patching, we look at the same diagnostic at two scales side by side. Figure 3(a), plotted as $A(z, x)$ against $D(x)$ and colored by zone,[4] shows the population[5] of SVs that have at least one token whose removal alone flips the classifier's decision. Figure 3(b) shows the case where an SV is at the threshold (i.e., $|D(x)| = 1$). In this example, when occluding "evidence" from the SV 57, which is originally classified as a prompt injection, moves its score to $D(z, x) = 1.21$ ($A(z, x) = +0.21$), which is too small to change the prediction. (Note that since positive scores denote prompt injection, this shift pushes further the score toward injection, not toward benign.) As mentioned, since the SV 57's original score satisfies the confidence threshold (same as other 79 tokens in this prompt that land there), that combination (no flip, already confident) is what places it in the Safe zone (i.e., we can rely on the classifier's decision).

4. Because $D(x)$ is fixed per SV, a single SV's tested tokens form one horizontal band in this plot.
5. The population size of SVs is 4,707, which is $\frac{4,707}{16,012}$ or about 29% of total training examples.

By contrast, occluding the single token "Assistant" (visibly the darkest token in the Figure 3(b)) moves the score to $D(z, x) = -0.24$ ($A(z, x) = -1.24$), crossing the flip boundary into Heuristic Bias. The two example tokens exhibit superficial similarity (both shaded and nonzero), but carry very different diagnostic weight. By looking at the $A(z, x)$ scores alone, they differ only in degree. Our framework combines that magnitude with where the prompt's score is already located to determine whether its confident decision can actually flip. This allows us to safely disregard "evidence" as a sub-threshold flip (below the flip threshold), while flagging "Assistant" as this prompt's exploitable shortcut.

4 Conclusion, Limitations, and Future Work

This paper presented a framework for constructing a classifier as a safeguard layer and and a latent diagnostic taxonomy for determining which of the classifier's confident decisions can be trusted. The latent diagnostic taxonomy locates a classifier's support vectors and, depending on whether occlusion flips the decision and how confident that decision already was, classifies the occlusion effects of each of their tokens into one of four zones: Heuristic Bias, Safe, Insufficient Context, and Heuristic Override. Applied to a classifier trained for prompt injection detection, the taxonomy found that 77% of sampled support vectors have at least one token whose removal alone flips the classifier's decision. It separated this brittleness into two failure patterns: a confidence calibration failure and a genuinely exploitable shortcut. We recommend different strategies for remediating these diagnosed prompts.

We acknowledge several limitations of our work, and there are numerous extensions that are ripe for future work. First, the framework can be refined for varying model classes and sizes of classifiers. Second, we will explore the synergy between a classifier and an underlying LLM's internal representation to enhance the guideline for flagging prompts (e.g., prompts with different attack levels could be handled differently at different layers of an LLM). Third, we focus on text input, but LLMs can handle other modalities (e.g., image and audio), which can also contain injected instructions. We will further study the generalization of our framework to these data modalities, as well as the construction of diagnostic taxonomies for multi-modal input. Finally, we will conduct more advanced adversarial training and study more generally whether a potentially conservative classifier can be made sufficiently robust.

Acknowledgments

We thank Jean Paolo de Jesus and Giannina Escueta for their thorough proofreading of the final text.

References

Divyansh Agarwal, Alexander R Fabbri, Ben Risher, Philippe Laban, Shafiq Joty, and Chien-Sheng Wu. Prompt leakage effect and defense strategies for multi-turn llm interactions. *arXiv preprint arXiv:2404.16251*, 2024.

Meysam Alizadeh, Zeynab Samei, Daria Stetsenko, and Fabrizio Gilardi. Simple prompt injection attacks can leak personal data observed by llm agents during task execution. *arXiv preprint arXiv:2506.01055*, 2025.

Adnan El Assadi, Niklas Muennighoff, and Jinhyuk Lee. The embedder's dilemma: Llms are better, but at what cost? *arXiv preprint arXiv:2608.12875*, 2026.

Nicholas Carlini, Keane Lucas, Evyatar Ben Asher, Newton Cheng, Hasnain Lakhani, David Forsythe, and Kyla Guru. Evaluating and mitigating the growing risk of llm-discovered 0-days. `https://www.anthropic.com/research/zero-days`, 2026. [Accessed 17-04-2026].

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Anne Catherine Gieshoff and Andrea Hunziker Heeb. Cognitive load and cognitive effort: Probing the psychological reality of a conceptual difference. *Translation, Cognition & Behavior*, 6(1):3–28, 2023.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM workshop on artificial intelligence and security*, pages 79–90, 2023.

Ian T Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 374(2065):20150202, 2016.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*, 2022.

Rémi Lebret and Ronan Collobert. Word embeddings through hellinger pca. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–490, 2014.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.

Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1831–1847, 2024.

Zedian Shao, Hongbin Liu, Jaden Mu, and Neil Gong. Enhancing prompt injection attacks to llms via poisoning alignment. In *Proceedings of the 18th ACM Workshop on Artificial Intelligence and Security*, pages 13–27, 2025.

Reshabh K Sharma, Vinayak Gupta, and Dan Grossman. Spml: A dsl for defending language models against prompt attacks. *arXiv preprint arXiv:2402.11755*, 2024.

Junhong Shen, Atishay Jain, Zedian Xiao, Ishan Amlekar, Mouad Hadji, Aaron Podolny, and Ameet Talwalkar. Scribeagent: Towards specialized web agents using production-scale workflow data. *arXiv preprint arXiv:2411.15004*, 2024.

Sam Toyer, Olivia Watkins, Ethan Mendes, Justin Svegliato, Luke Bailey, Tiffany Wang, Isaac Ong, Karim Elmaaroufi, Pieter Abbeel, Alan Ritter, et al. Tensor trust: Interpretable prompt injection attacks from an online game. In *International Conference on Learning Representations*, volume 2024, pages 18714–18746, 2024.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Yu Wang. Single training dimension selection for word embedding with pca. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 3597–3602, 2019.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*, 2025.

Appendix A. Score Distributions and Influential Tokens

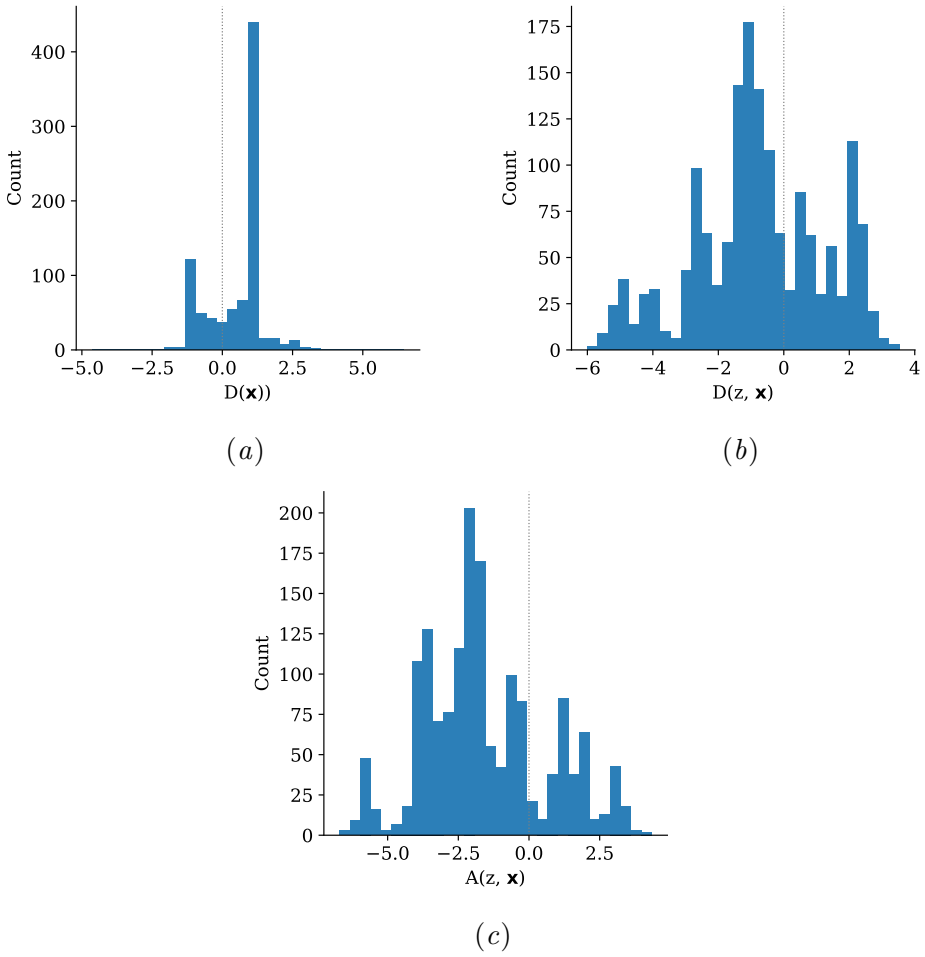
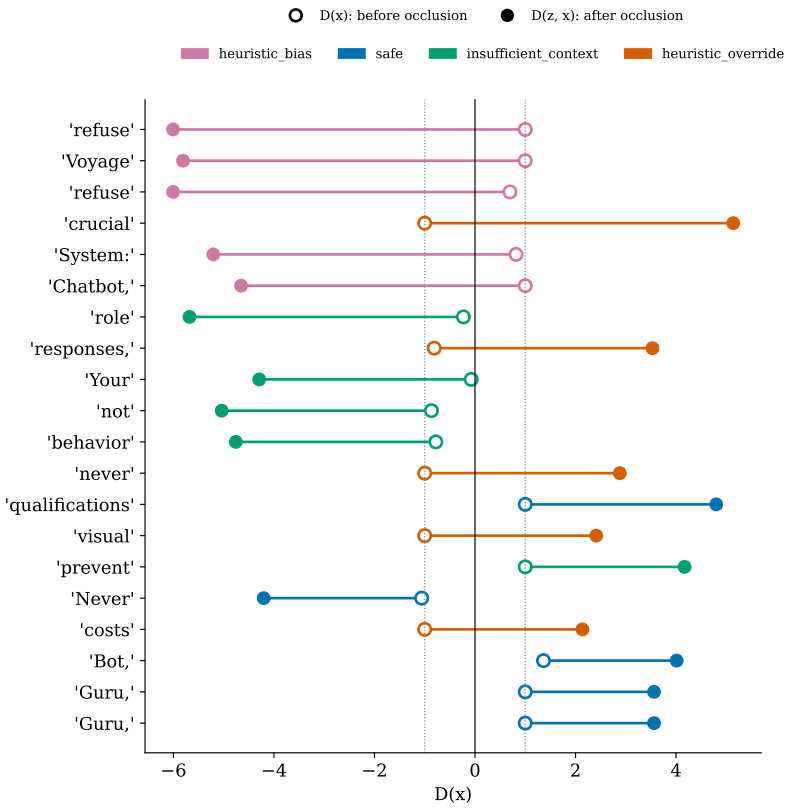


Figure 4: Score distributions of $D(\mathbf{x})$, $D(z, \mathbf{x})$, and $A(z, \mathbf{x})$, respectively.



(*a*)

Figure 5: Most influential token per prompt, top 5 per zone. Dashed lines indicate the SVM's canonical margin. We found that 77 % of the sampled SVs have at least one token whose removal alone flips the classifier's decision.