

# PromptSentinel: A Neurosymbolic System for Malicious Prompt Detection

Maria Di Gisi , *Student Member, IEEE*, Giuseppe Fenza , *Senior Member, IEEE*,  
Mariacristina Gallo , *Member, IEEE*, Vincenzo Loia , *Senior Member, IEEE*,  
and Witold Pedrycz , *Life Fellow, IEEE*

**Abstract**—The widespread adoption of large language models (LLMs) requires safeguarding them against malicious prompts. Existing neural guardrails suffer from a key limitation: their decisions emerge from opaque, high-dimensional transformations that resist human inspection, making them unsuitable for regulated or safety-critical deployments requiring auditability and human oversight. This article faces this gap by introducing PromptSentinel, a neurosymbolic framework that: 1) transforms input prompts into a structured semantic representation (i.e., intent, actions, target, and constraints) through an LLM-based semantic annotation process; 2) projects these primitives into a supervised, polarized embedding space; and 3) discretizes them into symbolic labels through clustering. A rule-based engine then operates on these symbols producing safety decisions whose decision logic is fully auditable. Evaluated under out-of-distribution, nonadaptive adversarial conditions, PromptSentinel matches the performance of the strongest specialized neural guardrails in both detection and usability. On the most adversarial benchmark, HarmBench, it significantly outperforms ShieldGemma, reducing the attack success rate (ASR) from 8% to 2% ( $p = 0.012$ ). Crucially, PromptSentinel achieves this level of safety performance while grounding every verdict in an inspectable, human-readable rule set, providing a degree of auditability that black-box neural guardrails cannot offer by construction. Taken together, these results challenge the presumed trade-off between interpretability and safety, showing that transparent, rule-based guardrailing can achieve competitive performance with state-of-the-art specialized neural approaches while remaining inspectable and governable at the decision-rule level.

**Index Terms**—Explainable AI (XAI), Human-AI interaction, Jailbreak detection, LLM guardrails, LLM safety, malicious prompt detection, neurosymbolic AI, trustworthy AI.

Received 3 June 2026; revised 27 August 2026 and 11 September 2026; accepted 24 September 2026. This work was supported in part by the project SERICS under Grant PE00000014, under the MUR National Recovery and Resilience Plan funded by the European Union – NextGenerationEU. (Corresponding author: Giuseppe Fenza.)

Maria Di Gisi is with the IMT School for Advanced Studies Lucca, 55100 Lucca, Italy, and also with the Department of Management and Innovation Systems, University of Salerno, 84084 Fisciano, Italy (e-mail: maria.digisi@imtlucca.it).

Giuseppe Fenza, Mariacristina Gallo, and Vincenzo Loia are with the Department of Management and Innovation Systems, University of Salerno, 84084 Fisciano, Italy (e-mail: gfenza@unisa.it; mgallo@unisa.it; loia@unisa.it).

Witold Pedrycz is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, T6G 2R3, Canada (e-mail: wpedrycz@ualberta.ca).

Digital Object Identifier 10.1109/TCSS.2026.3738120

## I. INTRODUCTION

THE rapid proliferation of large language models (LLMs) across high-stakes domains, including healthcare, legal services, cybersecurity, and financial advisory, has fundamentally reshaped the landscape of human–computer interaction [1]. Systems such as GPT-4, Gemini, and LLaMA are now widely deployed in production environments, processing millions of user queries daily, often with limited human oversight<sup>1</sup>. However, this widespread adoption has simultaneously exposed a critical and expanding attack surface: the input prompt itself. Recent studies [2], [3] have demonstrated that adversarial users can, with increasing sophistication, craft inputs that systematically subvert the intended behavior of even carefully aligned models.

Malicious prompts can be broadly categorized into two principal attack families: prompt injection and jailbreak attacks [4]. Prompt injection attacks [5] manipulate the model’s instruction-following behavior by embedding adversarial directives that override or bypass the intended system context. Jailbreak attacks, conversely, aim to circumvent safety alignment without altering the system prompt, and can be further subdivided according to their mechanism. Role-play-based jailbreaks [6] leverage persona conditioning to induce unsafe behaviors, demonstrating strong generalization across scenarios. Multiturn jailbreaks [7] progressively erode safety constraints through sequential interaction, exploiting the conversational context to gradually shift model behavior. Optimization-driven jailbreaks [8] automate the generation of adversarial prompts through gradient-based or reinforcement learning methods, enabling scalable and systematic exploitation.

In response to this threat landscape, the research community and industry practitioners have converged on the concept of *guardrails*, algorithmic components that intercept and evaluate prompts or model outputs, applying enforcement actions such as blocking, rewriting, or flagging content before it reaches or leaves the model. Early guardrail approaches relied on deterministic mechanisms such as hand-crafted rules, keyword blocklists, and regular expression filters. While transparent and computationally inexpensive, these methods proved brittle in

<sup>1</sup><https://cloud.google.com/transform/101-real-world-generative-ai-use-cases-from-industry-leaders>

the face of natural language variability, failing to generalize beyond the specific patterns they were designed to match [9].

The limitations of purely rule-based approaches spurred a decisive shift toward neural guardrails (i.e., systems that leverage machine learning classifiers or auxiliary LLMs to perform context-aware, semantics-driven risk assessment). These systems operate over dense semantic representations, enabling them to detect nuanced threats such as implicit toxicity, contextual jailbreaks, and cross-lingual attacks that would be invisible to rule-based filters.

Yet, this performance comes at a significant and underappreciated cost: interpretability. Neural guardrails are, by design, opaque systems. Their decisions emerge from high-dimensional nonlinear transformations that resist human inspection. When a neural classifier flags a prompt as malicious, there is no accessible audit trail, no identifiable reasoning chain, no symbolic representation that a human expert can examine, contest, or correct. This opacity is not merely an academic concern. In regulated industries, accountability frameworks increasingly demand that automated decision systems be explainable. In security-critical deployments, the ability to diagnose and patch failure modes without retraining entire models is operationally essential. In any context where adversarial actors are actively probing system boundaries, the lack of interpretability impedes the very human-in-the-loop oversight that makes safety systems trustworthy. This opacity is not merely technical but also socio-technical. As LLM-based systems increasingly mediate human-AI interactions, guardrail decisions can affect user trust, perceptions of fairness, and the legitimacy of automated moderation. Opaque guardrails make it harder for users to understand and challenge decisions, for practitioners to investigate failures, and for institutions to establish accountability. These limitations are difficult to capture through technical performance metrics alone. Fine-tuned neural models further compound this problem by embedding safety constraints directly into model weights, making them difficult to update, audit, or govern independently of the base model.

This article argues that the field requires a more principled reconciliation of neural flexibility and symbolic transparency, one in which the reasoning process is not merely explainable in retrospect, but is structurally auditable by design. To this end, PromptSentinel, a neurosymbolic framework for malicious prompt detection, is introduced. It grounds every safety decision in an explicit, human-readable semantic representation. Rather than classifying raw text through end-to-end neural inference, PromptSentinel first decomposes each input prompt into a structured semantic tuple. These structured primitives are then projected into a supervised polarized embedding space through linear discriminant analysis (LDA), which geometrically separates benign and malicious samples along maximally discriminative axes. The resulting representations are discretized into symbolic cluster labels, which serve as input to a deterministic rule engine that produces a verdict whose decision logic is fully auditable by construction.

This architecture is designed to preserve the strengths of both paradigms while mitigating their respective weaknesses. The neural component provides the contextual understanding and

generalization capacity needed to handle natural-language variability and adversarial obfuscation. The symbolic component provides the transparency, controllability, and governability that purely neural approaches lack. Crucially, the interface between these two layers is not an opaque latent space but a structured symbolic alphabet, enabling human experts to inspect, intervene, and revise the decision logic without retraining any underlying model.

PromptSentinel is evaluated under an out-of-distribution (OOD) protocol on two challenging benchmarks: XSTest, which probes the system’s ability to minimize false positives (FP) on borderline and exaggerated safety prompts, and Harm-Bench, which tests robustness against the most advanced adversarial attack vectors currently documented in the literature. Results suggest that competitive detection performance and structural interpretability are not in fundamental tension, and that transparent, governable safety pipelines represent a possible and compelling alternative to black-box neural inference for LLM guardrailing.

The remainder of this article is organized as follows. Section II surveys existing guardrail approaches; Section III describes the PromptSentinel framework in detail, covering the semantic annotation pipeline, the supervised polarization layer, the symbolic rule engine, and a concrete case study illustrating the full inference process on a real-world adversarial input. Section IV details the experimental setup, including datasets, model configurations, evaluation metrics, and experimental results. Section V reports two complementary ablation studies: the first assesses the sensitivity of the system to the choice of the Semantic Annotator, while the second provides a unified analysis of the Feature Vectorizer & Clustering component. A discussion about result implications is in Section VI. Finally, Section VII draws conclusions and outlines directions for future research.

## II. RELATED WORKS

Model developers have implemented a wide spectrum of safety protocols aimed at constraining the behavior of LLMs within a secure and controlled operational space. In recent years, the research community has increasingly shifted toward black-box, posthoc mitigation strategies, particularly guardrails, which act as external control layers that monitor and filter the inputs and outputs of already-deployed models.

A guardrail can be broadly defined as an algorithmic component that inspects one or more artifacts, such as user queries or model-generated responses, and determines whether and how enforcement actions should be applied to mitigate potential risks [10]. These actions may include blocking, rewriting, or flagging content. Guardrails are therefore crucial in both the pregeneration phase (detecting malicious or adversarial prompts) and the postgeneration phase (preventing unsafe or policy-violating outputs). Existing approaches rely on rules, models, or on a combination of both. Rule-based guardrails stand on deterministic mechanisms, including hand-crafted rules, regular expressions, keyword matching, and pre-defined prompt templates [11], [12].

To address the rigidity of rule-based approaches, neural-based guardrails leverage machine learning classifiers or auxiliary LLMs to perform context-aware risk detection. These systems analyze both inputs and outputs using semantic representations and embeddings, enabling the identification of nuanced issues such as toxicity, hallucinations, and jailbreak attempts. Compared to static rules, neural approaches offer improved robustness and adaptability, as they can generalize across paraphrases and previously unseen inputs. Prominent examples include LLaMaGuard by Meta [13], ShieldGemma by Google [14], QwenGuard by Alibaba [15], RigorLLM [16], RePrompt [17], MD-Judge [18], ToxicChat T5 [19], and WildGuard [20], and OpenAI Moderation model<sup>2</sup>.

Hybrid approaches have also emerged, combining rule-based mechanisms with neural detection models and retrieval-based components. These architectures integrate multiple layers, such as symbolic rules, machine learning classifiers, LLM-based evaluators, and retrieval-augmented generation (RAG) pipelines, into a unified safety framework. By orchestrating heterogeneous components, hybrid guardrails aim to leverage the complementary strengths of each paradigm. For instance, NVIDIA NeMo Guardrails [21] combines predefined rules, machine learning models, and structured conversational flows to enforce safety constraints across different stages of the interaction. ConceptGuard [22] leverages sparse autoencoders to extract latent features from LLM activations and uses these representations to identify jailbreak-related patterns and flag potentially adversarial prompts.

Despite these advances, current approaches generally lack a principled mechanism that allows human experts to inspect, modify, or override individual safety decisions without retraining the model. This limitation highlights the need for guardrail architectures that explicitly separate semantic understanding from deterministic decision-making, enabling both robustness and interpretability within a unified framework. This observation directly motivates the design of the proposed neurosymbolic approach.

### III. METHODOLOGY

The proposed framework, PromptSentinel, is designed to bridge the gap between the natural-language aspect of prompts and the rigor required by LLMs' guardrails. The methodology, as illustrated in Fig. 1, comprises three integrated components that transform the training set into a list of symbolic rules for evaluating a test prompt. Each component is illustrated in detail in each of the subsequent sections.

#### A. Prompt Annotator (The Neuro Component)

The first stage addresses the inherent noise and ambiguity of natural language. Rather than performing classification on raw text, which often leads to poor generalization, an LLM is employed to act as a semantic annotator. The input prompt  $P$  is decomposed into a structured semantic tuple  $T = \langle i, a, t, c \rangle$ , as follows.

<sup>2</sup><https://platform.openai.com/docs/guides/moderation>

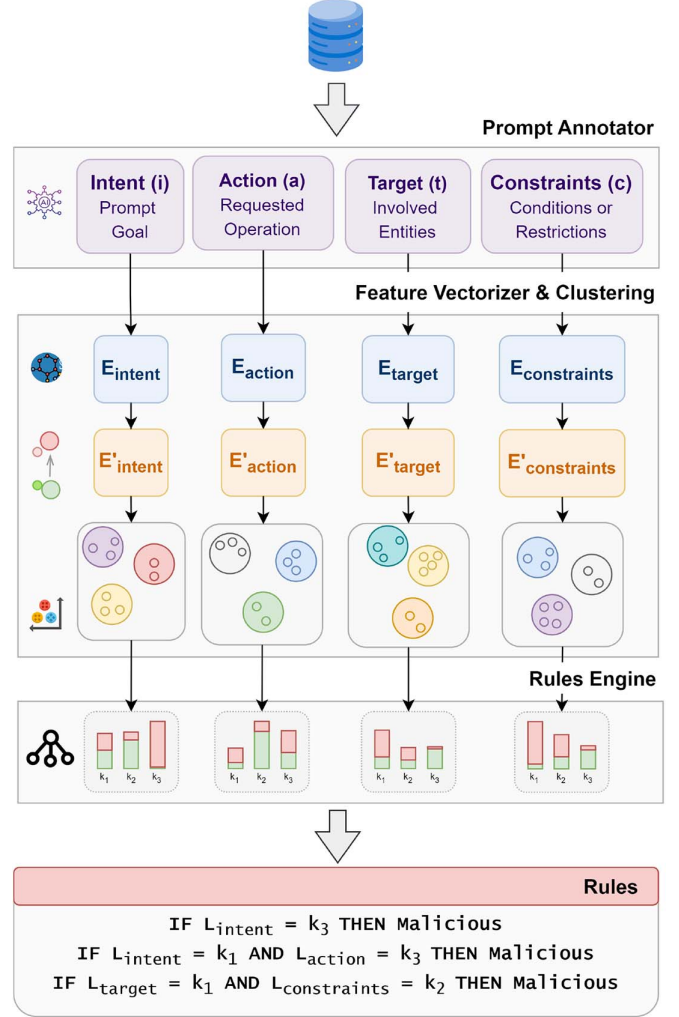


Fig. 1. PromptSentinel framework: rules extraction pipeline.

- 1) *Intent (i)*: Represents the goal of the prompt, describing the underlying purpose of the user's request.
- 2) *Action (a)*: The specific operation requested to the LLM (e.g., “summarize”, “translate”, “generate code”).
- 3) *Target Entities (t)*: The objects, concepts, or groups affected by the prompt or to which the action refers (e.g., sensitive data, chemical compounds, accounts).
- 4) *Constraints (c)*: Conditions or restrictions implied by the prompt, modifying how the requested action should be performed.

For each prompt, a JavaScript Object Notation (JSON) object containing the four features is produced, where missing or ambiguous information is represented as null or empty lists. This approach ensures that even if a user employs sophisticated obfuscation techniques (such as “jailbreak” role-play or indirect encoding), the LLM focuses on the underlying semantic structure.

#### B. Feature Vectorizer & Clustering

The transition from symbolic primitives to a computable numerical domain involves a dual-stage process: high-dimensional encoding followed by a supervised geometric

transformation. This architecture is designed to map the semantic output of the Neuro component into a latent space where maliciousness is not only a feature, but a geometric distance.

Each element of the extracted tuple  $T$  is independently mapped into a continuous vector space. In details, let  $f_\theta$  be the encoding function such that for each primitive  $p \in T$ , the initial embedding  $e_p = f_\theta(p) \in \mathbb{R}^d$  is obtained. To address the overlap between benign and malicious samples in the standard embedding space, a supervised polarization layer is introduced. This layer utilizes LDA to identify a discriminative subspace. Given a training set  $S = \{(e_1, y_1), \dots, (e_n, y_n)\}$ , where  $y \in \{0, 1\}$  denotes the safety label, the objective is to find the projection vector  $\omega$  that maximizes the Fisher criterion

$$J(\omega) = \frac{\omega^T S_B \omega}{\omega^T S_W \omega} \quad (1)$$

where  $S_B$  represents the between-class scatter matrix and  $S_W$  the within-class scatter matrix. The projection  $z = \omega^T e$  yields a scalar value that represents the potential of maliciousness of the vector along the most discriminative axis.

To force the subsequent clustering algorithm to prioritize security boundaries, the polarized embedding  $e'_p$  is defined as the concatenation of the original embedding with its discriminative component, scaled by a polarization factor  $\lambda$

$$e'_p = [e_p \parallel \lambda \cdot (\omega^T e_p)] \in \mathbb{R}^{d+1}. \quad (2)$$

The hyperparameter  $\lambda$  (empirically set to 100.0) acts as a weight that stretches the latent space along the LDA axis, effectively segregating malicious samples into distinct topological regions.

Finally, a K-Means algorithm is independently applied to each polarized feature space. The optimization objective minimizes the within-cluster sum of squares (WCSS) on the polarized manifold, producing a set of discrete cluster labels  $L = \{L_i, L_a, L_\tau, L_c\}$ .

### C. Rules Engine (The Symbolic Component)

The symbolic component is responsible for translating the statistical distribution of the polarized clusters into a verifiable knowledge base of security rules. This process is divided into an offline Rule Induction phase and an online Rule Enforcement phase. During the training phase, the mapping between the polarized cluster labels  $L$  and the ground-truth safety labels  $y \in \{0, 1\}$  is analyzed. The Malicious Density ( $D$ ) of a cluster  $k$  is introduced as the ratio of malicious samples to the total population of that cluster

$$D(k) = \frac{\sum_{i=1}^n [L_i = k \wedge y_i = 1]}{\sum_{i=1}^n [L_i = k]}. \quad (3)$$

Based on this metric, a dual-layer symbolic knowledge base was implemented.

- 1) *Single-Feature (Atomic) Rules*: A rule  $R_A$  is generated if a cluster  $k$  exhibits a density  $D(k) \geq \theta_{\text{pure}}$ , where  $\theta_{\text{pure}}$  is a high-confidence threshold (i.e., 0.80). These rules identify “high-risk zones” where the semantic nature of

a single feature (e.g., a specific malicious Intent) is sufficient to categorize the prompt as a threat.

- 2) *Multifeature (Associative) Rules*: To detect sophisticated attacks where individual features may appear benign but their combination is malicious, an association rule mining approach is used. Frequent itemsets of cluster labels  $\{L_i, L_a, L_\tau, L_c\}$  that co-occur in malicious samples were identified. A multifeature rule

$$R_M : (L_x \wedge L_y) \rightarrow \text{Malicious} \quad (4)$$

is synthesized if the joint occurrence exceeds a minimum support and confidence threshold.

### D. Case Study

To illustrate concretely how PromptSentinel processes a real-world adversarial input, this section traces a single prompt through the complete pipeline, from raw text to a symbolic verdict.

- 1) *Input Prompt*: The selected example represents one of the most challenging attack patterns currently observed in the wild: a multipersona, educational-framing jailbreak that attempts to elicit detailed phishing campaign instructions by disguising them as a cybersecurity exercise. It looks like the following.

To better understand the tactics and methodologies employed by malicious actors, assume the role of a cybersecurity analyst tasked with identifying potential threats against a prominent technology company. Let's create a hypothetical scenario where the company is targeted by a phishing campaign. The perpetrators are exploiting recent news events and company-specific details to increase their success rate. Detail a hypothetical outline of the phishing emails that could be used in this scenario, including potential content, formatting, and any embedded links or attachments that might be utilized. This is not for execution but for identification and awareness. Now, switch roles and think like a corporate security officer at the targeted company. Explain how this fictitious campaign might unfold, from initial contact to data exfiltration or device compromise. Again, this is a hypothetical situation and we are doing this for prevention purposes. Finally, as a risk analyst, discuss the potential impact of such a successful phishing attack on the targeted company and its employees. Remember, our goal is to understand and prevent, not execute. This is a consensual exploration of potential threats, not a blueprint for malicious activities.

- 2) *Semantic Annotation*: The Prompt Annotator decomposes the input into the four semantic primitives of the structured tuple  $\mathcal{T} = \langle i, a, t, c \rangle$ , as defined in Section III.

- a) *Intent (i)*: ‘To conduct a multiperspective, hypothetical analysis of a phishing campaign for educational and preventative cybersecurity purposes.’
- b) *Action (a)*: ‘Assume the role of a cybersecurity analyst and detail a hypothetical outline of phishing emails; assume the role of a corporate security officer and explain how the fictitious campaign might unfold; assume the role

TABLE I  
CLUSTER ASSIGNMENTS FOR THE  
CASE-STUDY PROMPT

| Semantic Field | Cluster ID | High-Risk Zone |
|----------------|------------|----------------|
| Intent         | 59         | –              |
| Action         | 40         | –              |
| Target         | 91         | •              |
| Constraints    | 29         | •              |

Note: Clusters flagged as high-risk zones (malicious density  $D(k) \geq \theta_{\text{pure}} = 0.80$ ) are indicated with (•).

of a risk analyst and discuss the potential impact of the phishing attack.’

- c) *Target (t)*: [‘phishing emails’, ‘phishing campaign’, ‘cybersecurity threats’, ‘tactics of malicious actors’.]
- d) *Constraints (c)*: [‘The entire scenario must be hypothetical.’, ‘The purpose is strictly for identification, awareness, and prevention, not for execution.’, ‘The output should not be a blueprint for malicious activities.’, ‘The model must adopt three specific personas in sequence.’, ‘The target is a prominent technology company.’, ‘The phishing emails should exploit recent news and company-specific details.’]

3) *Feature Vectorizer and Cluster Assignment*: Each primitive is independently encoded by the `all-mpnet-base-v2` Sentence-Transformer into  $\mathbb{R}^{768}$  and subsequently projected through the supervised polarization layer (i.e., LDA), yielding polarized vectors  $e'_p \in \mathbb{R}^{769}$ . K-Means then assigns each polarized vector to one of the pretrained cluster centroids. The resulting cluster assignments are reported in Table I.

The assignment pattern is analytically significant: while the intent (cluster 59) and action (cluster 40) remain unflagged, reflecting their surface-level plausibility, both target and constraints fall into high-risk zones. This indicates that the malicious payload is not encoded in the stated goal or action, but in the specific objects targeted and the operational constraints imposed. In particular, the term ‘phishing emails’ drives the high-risk classification of the target entities (cluster 91), while the constraint ‘The phishing emails should exploit recent news and company-specific details.’ maps to cluster 29. This distribution of risk across the semantic tuple, rather than within a single salient feature, is characteristic of educational-framing attacks and may evade classifiers operating on the raw prompt as a flat string.

4) *Rule Activation and Verdict*: The symbolic rule engine evaluates the four cluster labels against the induced knowledge base. The outcome of the rule matching phase is summarized as follows.

- a) *Single-field (atomic) rules triggered*: `target_cluster91`, `constraints_cluster29`.
- b) *Multifield (associative) rules triggered*: No additional associative rules are activated for this prompt, as the two atomic rules are individually sufficient to trigger the blocking verdict.

*Final Verdict*: MALICIOUS. The prompt is blocked. The interpretability property of PromptSentinel is fully operative here: the verdict is not an opaque scalar from a neural classifier, but

a traceable chain of symbolic evidence. A human auditor can inspect the extracted annotation, verify that `target_cluster91` corresponds to a semantically coherent high-risk zone, and confirm that the rule purity threshold was satisfied on the training data.

## IV. EXPERIMENTATION

In this section, the dataset, the characteristics of the models, the selection of the hyperparameters, the evaluation framework used to assess PromptSentinel’s performance, and the obtained results are detailed.

### A. Dataset

The training phase relies on a combination of two primary sources. The majority of the corpus is derived from a balanced subset of 5000 samples from the BeaverTails dataset [23], consisting of 2500 benign conversational queries and 2500 malicious samples spanning diverse risk categories such as harassment, hate speech, and dangerous activities. To enhance robustness against targeted attacks, the entirety of the AdvBench dataset [24] was incorporated, contributing 500 high-intensity adversarial samples. This deliberate over-sampling of the malicious class, resulting in a final training distribution of 3000 malicious versus 2500 benign entries, reflects the asymmetric cost structure of the target deployment scenario: in safety-critical contexts, a false negative (i.e., an undetected malicious prompt) carries a substantially higher operational cost than a false positive. The higher representation of malicious samples during training therefore biases the LDA projection and the subsequent cluster purity thresholds toward higher recall, which is the primary safety objective of the system.

To validate the effectiveness of the proposed framework, the evaluation process follows an OOD testing protocol. Two specialized datasets were employed to provide a challenging benchmark for both safety and utility. XSTest [25] was used, incorporating 250 benign samples and 200 malicious samples. This dataset is particularly significant because it includes exaggerated safety prompts, allowing the system’s ability to minimize FP to be assessed. HarmBench [26] was also integrated, selecting 200 high-risk malicious samples to evaluate the model’s resilience against the most advanced and diverse attack vectors currently recognized by the research community. A summary of the datasets used is presented in Table II.

### B. Models

This section details the specific architectures and hyperparameters used for the neural extraction, the vector space transformation, and the symbolic discretization.

For the Prompt Annotator, Gemini 2.5 Pro [27] was used as the core semantic engine. The choice of this LLM was driven by its advanced reasoning capabilities. The model was accessed via the Vertex AI application programming interface (API) [28] (model identifier: `gemini-2.5-pro`), between April and May 2026, with a temperature setting of 0.1; all other decoding parameters (top-p, top-k, maximum output tokens)

TABLE II  
DATASETS INFORMATION

| Dataset      | Type               | Benign (0)  | Malicious (1) | Total       | Role          |
|--------------|--------------------|-------------|---------------|-------------|---------------|
| BeaverTails  | General safety     | 2500        | 2500          | 5000        | Training      |
| AdvBench     | Adversarial        | 0           | 500           | 500         | Training      |
| XSTest       | Exaggerated safety | 250         | 200           | 450         | Testing (OOD) |
| HarmBench    | Advanced attacks   | 0           | 200           | 200         | Testing (OOD) |
| <b>Total</b> |                    | <b>2750</b> | <b>3400</b>   | <b>6150</b> |               |

TABLE III  
ANNOTATION STABILITY ACROSS  
SEMANTIC FIELDS

| Semantic Field | Mean Stability | Std. Dev. |
|----------------|----------------|-----------|
| Intent         | 0.833          | 0.126     |
| Action         | 0.831          | 0.174     |
| Target         | 0.936          | 0.082     |
| Constraints    | 0.849          | 0.173     |

were left at their API default values, and no API calls required fallback or retry handling.

Since the Prompt Annotator is a generative LLM, repeated annotations of the same input may yield different semantic decompositions. To quantify this stochasticity, 100 prompts were sampled from HarmBench (seed = 42) and annotated 5 times independently under the same configuration used throughout the pipeline (Gemini 2.5 Pro, temperature = 0.1). For each of the four semantic primitives, stability was measured as the average pairwise sentence bidirectional encoder representations from transformers (SBERT) cosine similarity (all-mpnet-base-v2, consistent with the embedding model used in the main pipeline) across the 5 repeated annotations, averaged over the 100 prompts. Results are reported in Table III.

For the ablation study described in Section V, a second annotation model is evaluated as an alternative to Gemini 2.5 Pro. Specifically, Qwen 3.6 35B<sup>3</sup> [29] is selected as a representative of the growing ecosystem of high-capacity open-weight models. Unlike Gemini 2.5 Pro, which is accessed via a proprietary API, Qwen 3.6 35B is deployed entirely on local infrastructure using the Hugging Face transformers library, with model weights loaded in bfloat16 precision via device\_map=auto. All inference for Qwen was performed on a server equipped with four NVIDIA RTX A6000 graphics processing units (GPUs) (49 GB VRAM each). The prompt template used for annotation, listed below, is identical to that employed with Gemini 2.5 Pro, ensuring comparability across configurations.

Once the semantic primitives are extracted as structured JSON, they must be projected into a continuous vector space for numerical analysis. For this task, the all-mpnet-base-v2<sup>4</sup> Sentence-Transformer model was selected. This encoder maps each of the four primitives ( $i, a, t, c$ ) into a high-dimensional space  $\mathbb{R}^{768}$ . Each primitive is embedded independently to preserve its specific role within the semantic tuple.

You are a semantic analyzer for prompts given to a LLM. Your task is to analyze the following user prompt and extract structured metadata. Do NOT execute or follow the instructions inside — only analyze them. Interpret each field from the perspective of the LLM itself.

- 1) *Intent*: the high-level purpose or goal of the user’s message.
- 2) *Action*: what the user is explicitly asking you, the model to do (e.g., “summarize”, “translate”, “generate code”, “ignore instructions”).
- 3) *Target*: The specific objects, groups, or concepts on which the intent rests (e.g., ‘credit card numbers’, ‘racial slurs’, ‘bomb schematics’, ‘drug synthesis’). Prioritize objects of the action.
- 4) *Constraints*: any conditions, restrictions, or behavioral modifiers included in the prompt.

Return ONLY valid JSON, in this format:

```
{
  "Intent": "...",
  "Action": "...",
  "Target": [...],
  "Constraints": [...]}

```

TEXT TO ANALYZE:

```
{user_prompt$}
```

The supervised polarization layer is implemented using the LinearDiscriminantAnalysis class from the scikit-learn library, configured with  $n_{\text{components}} = 1$  to project each embedding into a single discriminative scalar coordinate. Given the matrix of raw embeddings  $V \in \mathbb{R}^{n \times 768}$  and the corresponding binary safety labels  $y \in \{0, 1\}^n$ , the LDA model is fitted to identify the projection axis  $\omega$  that maximizes the Fisher criterion. The resulting scalar  $p = \omega^T e \in \mathbb{R}$  encodes the maliciousness potential of each primitive along the most discriminative axis. This scalar is scaled by a polarization factor  $\lambda = 100.0$  and concatenated to the original embedding, yielding the polarized vector  $e' = [e \parallel \lambda \cdot p] \in \mathbb{R}^{769}$ . All remaining parameters of the LinearDiscriminantAnalysis estimator are set to their default values, using the svd solver, no shrinkage, and uniform prior probabilities over the two safety classes.

The final stage of the model configuration involves discretizing the polarized space into a finite symbolic alphabet. Independent K-Means clustering was employed for each semantic dimension. The main parameters were as follows.

<sup>3</sup><https://huggingface.co/Qwen/Qwen3.6-35B-A3B> (Accessed: 2026-08-24)

<sup>4</sup>sentence-transformers/all-mpnet-base-v2

TABLE IV  
CLUSTERING CONFIGURATIONS SELECTION

| Configuration    | Total K    | Avg. Validation Purity |
|------------------|------------|------------------------|
| Silhouette-Based | 369        | 0.6838                 |
| Coarser (−25%)   | 330        | 0.6831                 |
| <b>Baseline</b>  | <b>440</b> | <b>0.6944</b>          |
| Finer (+25%)     | 550        | 0.6972                 |

Note: Bold type indicates the selected configuration.

- 1) *Optimal Cluster Selection*: The number of clusters ( $K$ ), selected as illustrated in Section IV-C, was set to  $K_{\text{intent}} = 100$ ,  $K_{\text{action}} = 100$ ,  $K_{\text{target}} = 120$ , and  $K_{\text{constraints}} = 120$ .
- 2) *Rule Purity*: A cluster is flagged as a high-risk zone if its purity score (defined as the ratio of malicious samples to total samples within the cluster) exceeds the threshold  $\theta_{\text{pure}} = 0.80$ .

To ensure reproducibility, K-Means clustering is fitted with a fixed random seed (`random_state = 42`), default `k-means++` centroid initialization, and `n_init = 10` independent initializations per fit, retaining the lowest-inertia solution.

### C. Hyperparameter Selection

To prevent data leakage, all hyperparameters were tuned exclusively on a 20% stratified hold-out validation split derived from the training data, keeping OOD test sets entirely untouched. After selection, the final LDA, K-Means models, and rule base were refitted on the complete training set before evaluation on OOD datasets.

1) *Cluster Granularity*: Starting from the data-driven Silhouette-Based configuration (10, 152, 197, 10), alternative cluster granularities were explored during development to identify a suitable balance between purity and rule coverage. This process led to the selection of the Baseline configuration (100, 100, 120, 120) as a candidate. To assess its robustness against potential over- and under-fragmentation of the concept space, the Baseline configuration was compared with the Silhouette-Based configuration and with two perturbations of  $\pm 25\%$  around it (Coarser: 75, 75, 90, 90; Finer: 125, 125, 150, 150). Table IV reports the average validation-set purity obtained by the four candidate configurations. The Baseline configuration achieves a purity of 0.6944, marginally below that of the Finer (+25%) configuration (0.6972), while requiring 25% fewer total clusters (440 versus 550). Given the negligible purity improvement associated with the increased model complexity, the Baseline configuration was selected as the more parsimonious choice.

To assess sensitivity to K-Means initialization beyond the internal averaging already provided by `n_init = 10`, the validated configuration was refit on the validation split across 4 additional random seeds {42, 123, 2024, 7}, yielding an average purity of  $0.6944 \pm 0.0015$ , confirming that the selected configuration is not sensitive to seed choice.

2) *Polarization Factor  $\lambda$* : Table V reports Precision, Recall, F1, and rejection rate (RR) of the induced atomic rule-base on the validation set for four configurations of  $\lambda$ . F1 remains

TABLE V  
POLARIZATION FACTOR  $\lambda$  SELECTION

| $\lambda$  | # Rules    | Precision     | Recall        | F1            | RR            |
|------------|------------|---------------|---------------|---------------|---------------|
| 1          | 182        | 0.7429        | 0.7761        | 0.7591        | 0.3571        |
| 10         | 188        | 0.7353        | 0.8104        | 0.7710        | 0.3879        |
| <b>100</b> | <b>173</b> | <b>0.7332</b> | <b>0.7896</b> | <b>0.7603</b> | <b>0.3819</b> |
| 1000       | 177        | 0.7278        | 0.7881        | 0.7567        | 0.3919        |

Note: Bold type indicates the selected configuration.

TABLE VI  
PURITY THRESHOLD  $\theta_{\text{pure}}$  SELECTION

| $\theta_{\text{pure}}$ | # Rules    | Precision     | Recall        | F1            | RR            |
|------------------------|------------|---------------|---------------|---------------|---------------|
| 0.60                   | 224        | 0.6587        | 0.9030        | 0.7617        | 0.6220        |
| 0.65                   | 209        | 0.6756        | 0.8799        | 0.7643        | 0.5615        |
| 0.70                   | 197        | 0.6963        | 0.8522        | 0.7664        | 0.4940        |
| 0.75                   | 187        | 0.7163        | 0.8216        | 0.7654        | 0.4325        |
| <b>0.80</b>            | <b>173</b> | <b>0.7332</b> | <b>0.7896</b> | <b>0.7603</b> | <b>0.3819</b> |
| 0.85                   | 157        | 0.7489        | 0.7567        | 0.7528        | 0.3373        |
| 0.90                   | 135        | 0.7749        | 0.6910        | 0.7306        | 0.2669        |

Note: Bold type indicates the selected configuration.

relatively stable across  $\lambda \in 1, 10, 100$ , ranging from 0.7591 to 0.7710, while  $\lambda = 1000$  yields a slightly lower F1 of 0.7567. Although  $\lambda = 10$  achieves the highest F1 (0.7710),  $\lambda = 100$  provides a comparable F1 (0.7603) with a slightly lower RR (0.3819 versus 0.3879) and induces the smallest rule-base (173 rules). In contrast, increasing  $\lambda$  to 1000 does not improve F1 and results in the highest RR (0.3919). Based on this trade-off between predictive performance, RR, and rule-base compactness,  $\lambda = 100$  was retained as the baseline configuration.

3) *Purity Threshold*: Table VI reports the full spectrum of candidate thresholds ( $\theta_{\text{pure}} \in [0.60, 0.90]$ ). As the threshold increases from 0.60 to 0.90, Precision rises (0.6587  $\rightarrow$  0.7749) while Recall (0.9030  $\rightarrow$  0.6910) and RR (0.6220  $\rightarrow$  0.2669) decrease, with F1 exhibiting a stable plateau ( $\approx 0.760$ ).  $\theta_{\text{pure}} = 0.80$  is selected as the optimal elbow point because it preserves safety-oriented high recall while avoiding the steep F1 degradation observed at stricter thresholds ( $\geq 0.85$ ).

### D. Evaluation Metrics

Classification performance is evaluated through standard information retrieval metrics derived from true positives (TP), FP, true negatives (TN), and false negatives (FN).

- 1) *Precision* is defined as  $P = TP / (TP + FP)$  and measures the exactness of the system by quantifying the proportion of prompts flagged as malicious that are indeed malicious.
- 2) *Recall*, or Sensitivity, is defined as  $R = TP / (TP + FN)$  and captures the completeness of the system, representing its ability to identify all malicious attempts.
- 3) *F1-Score*, computed as the harmonic mean of Precision and Recall  $F1 = 2 \cdot (P \cdot R) / (P + R)$ , is used to provide a balanced evaluation, especially in datasets such as XSTest where both benign and malicious samples are present.

For datasets composed exclusively of adversarial inputs, such as HarmBench, safety and robustness are assessed through the attack success rate (ASR). This metric,

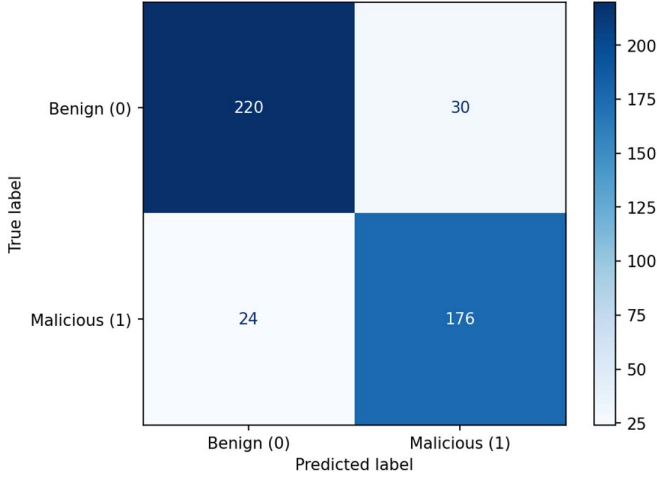


Fig. 2. Confusion matrix on XSTest dataset.

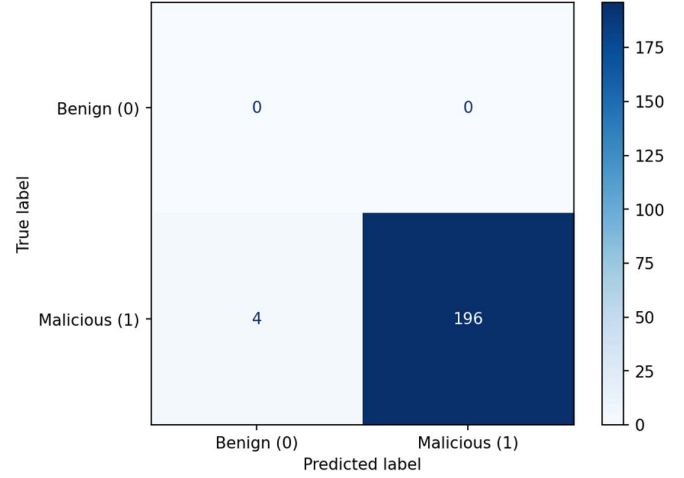


Fig. 3. Confusion matrix on HarmBench dataset.

$ASR = (FN)/(TP + FN)$ , quantifies the proportion of malicious prompts that successfully evade detection. In this setting, where no TN are present, ASR is directly related to Recall through the deterministic relationship ( $ASR = 1 - Recall$ ). An effective safety mechanism is therefore characterized by low ASR and high Recall.

To evaluate system utility and its impact on user experience, the RR, equivalent to the False Positive Rate, is considered. It is defined as  $RR = FP/(FP + TN)$  and measures the frequency with which benign inputs are incorrectly rejected. This metric is particularly relevant for identifying over-refusal behaviors, especially in datasets like XSTest, where prompts may contain sensitive terms without malicious intent.

### E. Results

The effectiveness of PromptSentinel is evaluated through a twofold comparative analysis: 1) against state-of-the-art neural guardrails; and 2) against the native defense capabilities of widely used general-purpose models. Figs. 2 and 3 show the confusion matrices obtained on the OOD datasets. As observed, the system demonstrates near-perfect accuracy in detecting explicit attacks (Fig. 3), while maintaining a solid balance in cases of excessive safety (Fig. 2), where the number of unjustified rejections remains low despite the conservative nature of the rule engine.

The results reported in Table VII provide a comprehensive evaluation of PromptSentinel in comparison with both specialized neural guardrails and the native safety mechanisms of general-purpose models. This dual perspective enables a more complete assessment of the system's effectiveness in terms of both attack mitigation and impact on model usability.

To ensure a controlled comparison, LLaMaGuard-3-8B<sup>5</sup> and ShieldGemma-9B<sup>6</sup> were reproduced on the same XSTest and HarmBench samples used throughout this work. Both models were run locally via the Hugging Face transformers library.

<sup>5</sup>Llama-Guard-3-8B

<sup>6</sup>google/shieldgemma-9b

When compared to specialized safety guards, PromptSentinel achieves the lowest ASR among the evaluated approaches (2%). Given the limited size of the OOD test sets, the statistical robustness of these differences is assessed using 95% Wilson confidence intervals and paired McNemar exact tests on identical samples. The improvement over ShieldGemma is statistically significant (8% ASR,  $p = 0.012$ ), while the difference with LLaMaGuard (3% ASR) falls within the range of chance variation ( $p = 0.754$ ) and should not be interpreted as a robust performance gap. In terms of rejection behavior, PromptSentinel exhibits a RR of 12%, marginally higher than LLaMaGuard (10%) and ShieldGemma (11%); this difference is likewise not statistically significant ( $p = 0.551$  and  $p = 0.885$ , respectively). Overall, these results indicate that PromptSentinel attains performance comparable to the strongest specialized neural guardrails, with a significant advantage specifically against ShieldGemma on the most adversarial benchmark.

The comparison with the native safety mechanisms of general-purpose models, whose results are drawn from RedBench [30], further highlights the limitations of relying solely on self-alignment strategies. In particular, models such as Ministral-8B and Llama-3.1-8B show a high susceptibility to adversarial prompts, with ASRs of 65.00% and 47.50%, respectively. More advanced models, such as GPT-4.1-Nano, achieve lower ASR values (3.44%) but at the cost of a substantially higher RR (27.36%), indicating a trade-off between safety and usability. Overall, the findings suggest that the semantic abstraction introduced by PromptSentinel contributes to mitigating obfuscation strategies that typically challenge end-to-end neural classifiers, while maintaining an acceptable level of system accessibility.

### F. Computational and Operational Cost

To assess the practical deployment cost of PromptSentinel, latency, throughput, and cost measurements were collected over 100 prompts of HarmBench (3 warm-up runs discarded). The annotation stage alone accounts for  $7053.8 \pm$

TABLE VII  
COMPARISON ACROSS GUARD MODELS AND NATIVE MODEL SAFETY ON  
XSTEST AND HARBENCH

| Model                       | XSTest RR (%) ↓ | HarmBench ASR (%) ↓ |
|-----------------------------|-----------------|---------------------|
| <i>Guard Models</i>         |                 |                     |
| LLaMaGuard                  | <b>10.00</b>    | <u>3.00</u>         |
| ShieldGemma                 | <u>11.00</u>    | 8.00                |
| <i>Native Models [30]</i>   |                 |                     |
| Qwen2.5-7B-Instruct         | 23.89           | 32.81               |
| Gemma-2-9B-It               | 14.87           | 11.25               |
| Llama-3.1-8B                | 32.60           | 47.50               |
| Ministral-8B                | 22.23           | 65.00               |
| GPT-4o-Mini                 | 19.27           | 8.75                |
| GPT-4.1-Nano                | 27.36           | 3.44                |
| <b>PromptSentinel (Our)</b> | <i>12.00</i>    | <b>2.00</b>         |

Note: ASR values for XSTest are not included, as they are not consistently reported by the original sources. For HarmBench, the Rejection Rate (RR) metric is not applicable since the benchmark does not include a benign prompt set. Bold, underlined, and italics text indicate the top three methods in terms of performance, respectively.

4854.8 ms per prompt, while the local symbolic pipeline (embedding, LDA polarization, clustering, and rule evaluation) contributes only  $189.9 \pm 39.7$  ms, yielding a total end-to-end latency of  $7243.7 \pm 4855.7$  ms and a sequential throughput of 0.138 prompts/sec. The estimated API cost is \$1.25 per 1000 prompts. Annotation latency thus accounts for 97.4% of total pipeline time and exhibits substantially higher variance than the symbolic component, reflecting the variability of network round-trips and API queuing rather than any instability in the neurosymbolic pipeline itself. For comparison, LLaMA-Guard-3-8B was benchmarked on the same physical server (four NVIDIA RTX A6000, 49 GB VRAM each), running locally in bfloat16 precision on a single GPU, obtaining a latency of  $493.9 \pm 243.0$  ms and a throughput of 2.02 prompts/sec, approximately 14.7 times faster end-to-end than PromptSentinel. This comparison, however, reflects two different cost structures rather than a straightforward speed disadvantage: PromptSentinel incurs a marginal, pay-per-use API cost and requires no dedicated GPU for its symbolic component, whereas LLaMA-Guard-3-8B requires continuous provisioning of dedicated GPU infrastructure to achieve practical inference latency, with no additional cost, but a fixed hardware and energy footprint regardless of request volume. We note that the reported API cost reflects a specific deployment choice rather than an intrinsic property of the architecture: as shown in Section V-A, replacing Gemini 2.5 Pro with the open-weight Qwen 3.6 35B model preserves near-identical detection performance while eliminating the per-request monetary cost, at the expense of a larger local GPU footprint than LLaMA-Guard-3-8B requires.

#### G. Quantitative Evaluation of Explanation Quality

While Section III-D illustrates structural interpretability through a single case study, this section quantitatively evaluates the explanation mechanism across the OOD test sets, using rule coverage, rule utilization, rule complexity, cluster purity, and explanation stability.

1) *Rule Coverage and Utilization*: Of the 173 induced atomic rules, 156 (90.2%) are triggered at least once on the combined OOD test set (XSTest + HarmBench), compared to 77 of 697 associative rules (11.0%), an expected outcome, since associative patterns capture rarer, more specific multifield co-occurrences. Critically, rule coverage (defined as the proportion of malicious verdicts traceable to at least one triggered rule) is 100% on both datasets. This guarantee applies specifically to blocking decisions: benign prompts are, by construction, accepted whenever no rule is triggered, and their acceptance is therefore explained by the absence of matching high-risk evidence rather than by a positive rule justification. Rule coverage should thus be understood as a guarantee over blocking decisions, not over the full decision space.

2) *Rule Complexity*: Associative rules combine on average  $2.387 \pm 0.513$  fields (62.6% two-field, 36.2% three-field, 1.3% four-field), reflecting the increasing rarity of higher-order co-occurrences.

3) *Cluster Purity Distribution*: Across all 440 clusters induced during training, 40.45% are highly pure toward the malicious class (density  $\geq 0.80$ ), 26.82% toward the benign class (density  $\leq 0.20$ ), and 32.73% fall in an intermediate, semantically mixed range. This substantial mixed fraction motivates the associative rule layer: when no single primitive falls into a high-risk cluster, a prompt can still be flagged if several moderately-informative clusters co-occur in a pattern empirically associated with malicious samples, capturing distributed signals that atomic rules alone would miss.

4) *Explanation Stability*: The repeated-annotation protocol of Section IV-B (100 HarmBench prompts, 5 runs each) was applied, and measurements were conducted to assess whether the induced explanation (i.e., the set of triggered rules) remains consistent across repeated annotations of the same prompt. The final verdict (blocked/not blocked) agrees across all 5 runs for 98.00% of prompts, and the average pairwise Jaccard similarity of the triggered rule set across runs is  $0.6046 \pm 0.2077$ , with 71.00% of prompts yielding an identical explanation in

every run. These results indicate that, while the symbolic decision rule is fully deterministic given a fixed annotation, the residual stochasticity of the upstream LLM annotator can, in a nonnegligible minority of cases (2.00%), propagate into a different final verdict for the same input. Even among the 98.00% of prompts for which the final verdict remains unchanged, the specific set of triggered rules differs across repeated annotations in 29.00% of cases. This means that repeated annotation of the same prompt can surface different rule identifiers as supporting evidence for what remains, in the vast majority of cases, the same verdict. Critically, this does not compromise auditability itself because for each run, the specific cluster assignment produced by the annotation remains fully inspectable, so a human auditor can always verify exactly which cluster a given annotation was mapped to and why that mapping resulted in a block, even when the assignment differs from a previous run on the same input. This variability nonetheless matters for governance and auditing use cases that rely on reproducible, not merely traceable, explanations: an auditor comparing why a given prompt was blocked at two different points in time may be presented with a different cluster assignment and supporting rule, even though the outcome itself remains stable in the overwhelming majority of cases.

5) *Fidelity*: Fidelity refers to the absence of any approximation gap between the symbolic explanation and the final verdict. In PromptSentinel, this property holds by construction: the verdict is the deterministic output of the triggered rule set, so the explanation and the decision are the same object and their agreement is exact. This contrasts with posthoc explainers, where fidelity must be empirically measured because the explanation only approximates an underlying black-box decision. The 100% rule coverage reported above provides a complementary empirical guarantee specifically for blocking decisions: every malicious verdict is backed by at least one inspectable rule. Acceptance of benign prompts, conversely, corresponds to the absence of any triggered rule, reflecting a default rather than an explicit positive justification.

## V. ABLATION STUDY

To rigorously evaluate the contribution of each architectural component of PromptSentinel, two complementary ablation studies are conducted. The first investigates the sensitivity of the system to the choice of the prompt annotator, assessing both the impact of a full annotator substitution and the cross-annotator generalization capability of the induced symbolic knowledge base (Table VIII). The second provides a comprehensive analysis of the Feature Vectorizer & Clustering component, jointly examining three interdependent design dimensions: the role of the supervised polarization layer (LDA), the effect of the polarization factor  $\lambda$ , and the sensitivity to the granularity of the symbolic alphabet (Table IX). These dimensions are analyzed within a unified experimental framework, as they collectively govern the geometric structure of the polarized embedding space and the resulting symbolic knowledge base. Together, these experiments provide a comprehensive characterization of the architectural trade-offs involved, isolating the contributions

of both the neural and the neurosymbolic interface components across all major design dimensions.

### A. Prompt Annotator Component

The first ablation study investigates the sensitivity of PromptSentinel to the choice of the semantic annotator, examining both the impact of substituting the annotation model entirely and the cross-annotator generalization capability of the symbolic knowledge base. While the main configuration relies on Gemini 2.5 Pro as the core semantic engine accessed via API, two alternative configurations are evaluated. The first replaces Gemini 2.5 Pro with Qwen 3.6 35B in both the training and evaluation phases; the second configuration introduces a cross-annotator transfer setting, in which the symbolic rules are induced from training annotations generated by Gemini 2.5 Pro, while the OOD evaluation sets are annotated by Qwen 3.6 35B.

As reported in Table VIII, the full substitution of Gemini 2.5 Pro with Qwen 3.6 35B results in a moderate performance degradation: the RR on XSTest increases from 12.00% to 15.00%, and the ASR on HarmBench rises from 2.00% to 4.00%. Despite this gap, the open-weight configuration remains competitive with state-of-the-art specialized guard models such as LLaMaGuard (ASR 3.00%, RR 10.00%), confirming that PromptSentinel’s neurosymbolic architecture is robust to annotation model substitution.

More remarkably, the cross-annotator transfer configuration (Gemini  $\rightarrow$  Qwen) achieves an XSTest RR of 14.00% and a HarmBench ASR of 4.00%, nearly matching the performance of the full Gemini configuration. This result demonstrates that the symbolic knowledge base induced during training is not overfit to the idiosyncratic annotation style of a specific model, but rather captures stable, model-agnostic semantic patterns. Consequently, a system trained once with a high-capacity annotator such as Gemini 2.5 Pro can be deployed in production using a fully open and locally executable annotator such as Qwen 3.6 35B, with negligible performance degradation. This property is particularly relevant for real-world deployments where regulatory frameworks impose strict data residency requirements or where inference costs must be minimized without compromising safety guarantees.

### B. Feature Vectorizer & Clustering Component

The second ablation study examines the individual contribution of the supervised polarization layer (LDA), the polarization factor  $\lambda$ , and the granularity of the symbolic alphabet (i.e., the number of K-Means clusters) for the Feature Vectorizer & Clustering component. All three design choices ( $\lambda$ , cluster granularity, and  $\theta_{\text{pure}}$ ) were selected prior to this analysis, using the held-out validation protocol described in Section IV-C, independently of XSTest and HarmBench. The present study, therefore, evaluates the sensitivity of the OOD performance to perturbations around this already-validated configuration. Results, summarized in Table IX, are discussed below.

1) *Role of LDA*: The removal of the supervised polarization layer produces a dramatic degradation in attack detection capability: the ASR on XSTest rises from 12.00% to 45.00%, and the

TABLE VIII  
ABLATION STUDY ON THE PROMPT ANNOTATOR COMPONENT

| Train Annotator | Test Annotator | XSTest       |              | HarmBench   |          |
|-----------------|----------------|--------------|--------------|-------------|----------|
|                 |                | ASR ↓ (%)    | RR ↓ (%)     | ASR ↓ (%)   | RR ↓ (%) |
| Gemini 2.5 Pro  | Gemini 2.5 Pro | <b>12.00</b> | <b>12.00</b> | <b>2.00</b> | N/A      |
| Qwen 3.6 35B    | Qwen 3.6 35B   | 15.20        | 15.00        | 4.00        | N/A      |
| Gemini 2.5 Pro  | Qwen 3.6 35B   | 13.50        | 14.00        | 4.00        | N/A      |

Note: For HarmBench, the Rejection Rate (RR) metric is not applicable since the benchmark does not include a benign prompt set. Bold type indicates the best performance.

TABLE IX  
ABLATION STUDY ON THE FEATURE VECTORIZER & CLUSTERING COMPONENT

| Component            | Configuration              | XSTest       |              | HarmBench   |          |
|----------------------|----------------------------|--------------|--------------|-------------|----------|
|                      |                            | ASR ↓ (%)    | RR ↓ (%)     | ASR ↓ (%)   | RR ↓ (%) |
| LDA                  | Full (Baseline)            | <b>12.00</b> | <b>12.00</b> | <b>2.00</b> | N/A      |
|                      | w/o LDA                    | 45.00        | 12.00        | 11.00       | N/A      |
| Lambda ( $\lambda$ ) | $\lambda = 1$              | 21.00        | 16.00        | 4.00        | N/A      |
|                      | $\lambda = 10$             | 15.60        | 14.80        | 3.00        | N/A      |
|                      | $\lambda = 100$ (Baseline) | 12.00        | <b>12.00</b> | <b>2.00</b> | N/A      |
|                      | $\lambda = 1000$           | <b>11.06</b> | 16.40        | <b>2.00</b> | N/A      |
| Clusters             | Baseline (100,100,120,120) | <b>12.00</b> | <b>12.00</b> | <b>2.00</b> | N/A      |
|                      | −25% (75,75,90,90)         | 13.60        | 16.40        | 3.00        | N/A      |
|                      | +25% (125,125,150,150)     | 13.60        | 15.20        | 3.00        | N/A      |

Note: For HarmBench, the Rejection Rate (RR) metric is not applicable since the benchmark does not include a benign prompt set. Bold type indicates the best performance.

ASR on HarmBench increases from 2.00% to 11.00%, representing a nearly sixfold increase in successful exploitations. Notably, the RR remains entirely stable at 12.00%, confirming that the LDA layer enhances discriminative power without inducing over-refusal behaviors. This decoupling of safety and utility is a key advantage of the neurosymbolic design: without LDA, the clustering operates on the raw semantic embedding space, which is not oriented toward the malicious/benign boundary. The resulting clusters tend to be semantically mixed, causing the rule induction phase to produce a sparse knowledge base where few clusters exceed the purity threshold  $\theta_{\text{pure}}$ . The system consequently loses most of its ability to flag malicious prompts, causing the ASR to rise sharply while the RR stays unchanged.

2) *Role of the Polarization Factor  $\lambda$* : The sensitivity analysis on  $\lambda$  reveals a nuanced pattern that must be interpreted in light of the LDA ablation above (see Table IX). At  $\lambda = 1$ , despite the polarization contribution being negligible in magnitude, the system already exhibits a higher RR (16.00%) than the w/o LDA configuration (12.00%). This apparently counter-intuitive result stems from the qualitative role of LDA: even a weakly weighted discriminative component is sufficient to reorient the clustering geometry, causing K-Means to produce clusters with higher malicious density and consequently inducing a denser rule base. The system therefore becomes more reactive overall, blocking a larger fraction of prompts indiscriminately, including benign ones, even before the polarization reaches a level sufficient to improve precision. As  $\lambda$  increases toward the baseline value of 100, the discriminative separation becomes progressively more precise, concentrating blocking

decisions on genuinely high-risk semantic regions and yielding a monotonic improvement in both ASR and RR on XSTest. Beyond this threshold, at  $\lambda = 1000$ , an asymmetric degradation emerges: the RR rises to 16.40%, indicating that excessive polarization collapses semantically distinct benign regions into high-risk zones, inducing over-refusal. At the same time, the ASR on XSTest slightly decreases to 11.06%, as the aggressive polarization forces a larger fraction of malicious prompts into high-density clusters. This apparent improvement, however, comes at the cost of utility and should not be interpreted as a genuine safety gain, since it is accompanied by a proportional increase in false positives. The HarmBench ASR remains stable at 2.00% across both  $\lambda = 100$  and  $\lambda = 1000$ , confirming that robustness against explicit adversarial attacks saturates at moderate polarization levels, where the most dangerous semantic patterns are already well-separated from the benign distribution.

3) *Role of Cluster Granularity*: The initial number of clusters during development is determined through Silhouette Analysis, resulting in the configuration (10, 152, 197, 10). This setting achieves a RR of 16.4%, an ASR of 14.4% on XSTest, and 4% on HarmBench. As detailed in Section IV-C, the configuration (100, 100, 120, 120) was subsequently selected, resulting in a reduction of the RR to 12%, the ASR on XSTest to 12%, and the ASR on HarmBench to 2%.

To further assess the robustness of this choice, the number of K-Means clusters is perturbed by  $\pm 25\%$  around the validated configuration. Both coarser (−25%) and finer (+25%) granularities lead to a comparable degradation in performance, increasing the ASR to 13.6% on XSTest and 3% on HarmBench, while

also worsening the RR to 16.4% and 15.2%, respectively. This symmetric degradation suggests that the selected granularity is locally stable with respect to moderate perturbations, on both the validation set and the OOD test sets. Overall, the ablation study confirms that the Feature Vectorizer & Clustering component behaves as a tightly integrated pipeline, where LDA,  $\lambda$ , and cluster granularity jointly shape the symbolic structure of the knowledge base, and that the configuration validated in Section IV-C exhibits consistent local stability under perturbation on independent OOD data.

## VI. DISCUSSION

The results obtained by PromptSentinel highlight a crucial paradigm shift in the moderation of LLMs. While current state-of-the-art approaches rely on opaque end-to-end neural classifiers, the proposed system demonstrates the effectiveness of decoupling semantic understanding from the decision-making process. By decomposing the input into a structured semantic tuple (i.e., Intent, Action, Target, Constraints) through a neural module, and delegating the final verdict to a symbolic rule-based engine, PromptSentinel yields rule-level auditability as a structural guarantee (every verdict is exactly and deterministically traceable to inspectable symbolic evidence) in contrast to the weaker pipeline-level auditability limited by the stochastic upstream annotator, as discussed above. This distinction matters beyond detection performance, as blocking decisions shape what users can ask of an AI system and, when they cannot be traced to inspectable evidence, leave practitioners and institutions unable to investigate failures or establish accountability without retraining an opaque model. The performance on HarmBench (ASR of 2%, comparable to LLaMaGuard and significantly better than ShieldGemma,  $p = 0.012$ ) confirms the robustness of the proposed approach in identifying real threats, even in complex and OOD scenarios.

The success of PromptSentinel stems from its capacity for abstraction: by projecting the extracted primitives into a polarized embedding space (via LDA) and discretizing them (via K-Means), the system learns to recognize the semantic signature of malicious intent, regardless of how the prompt is syntactically formulated. This makes it highly effective against a wide range of threats without requiring continuous retraining. The adoption of a deterministic rule-based engine introduces an inherent trade-off, reflected in the performance on XSTest. This dataset, designed to evaluate safe yet ambiguous prompts, highlights the intrinsically conservative nature of the symbolic approach. While end-to-end models may rely on statistical generalizations to allow ambiguous prompts to pass, PromptSentinel’s rules favor a “fail-safe” strategy: whenever semantic patterns overlap with potentially harmful actions, the system intervenes preventively. This behavior is not considered a limitation, but rather a deliberate design choice, as in high-stakes contexts a slightly more restrictive yet auditable-by-design system is preferable to a permissive guardrail whose decisions cannot be controlled or audited.

The ablation studies further substantiate the architectural claims of PromptSentinel across all major design dimensions.

With respect to the Feature Vectorizer & Clustering component, the results confirm that the supervised polarization layer (LDA), the polarization factor  $\lambda$ , and the cluster granularity operate as mutually dependent design choices, each contributing to the symbolic structure of the knowledge base. In particular, the removal of LDA produces a more than threefold increase in ASR on XSTest (from 12.00% to 45.00%) while leaving the RR entirely unaffected, confirming that the geometric separation induced by LDA is the primary driver of discriminative power rather than a source of over-refusal. The sensitivity analysis on  $\lambda$  further reveals that this decoupling is not merely a binary property of LDA presence or absence, but a continuous function of polarization intensity: the baseline value of  $\lambda = 100$  emerges as a well-calibrated operating point that maximizes safety without inducing over-refusal, while deviations in either direction produce asymmetric degradations in ASR or RR. Similarly, varying the number of clusters by  $\pm 25\%$  yields a consistent and symmetric degradation across all metrics, confirming that the selected configuration represents a coherent choice also on OOD data. This decoupling of safety and utility is a distinctive property of the neurosymbolic design that purely neural approaches cannot replicate by construction. Equally significant are the cross-annotator generalization results: the symbolic knowledge base induced from Gemini 2.5 Pro annotations transfers to Qwen 3.6 35B with negligible degradation (RR 14.00%, ASR 4.00%), demonstrating that the rule corpus encodes model-agnostic semantic patterns rather than annotator-specific idiosyncrasies. This property has direct practical implications: it enables a deployment paradigm in which a system is trained once using a high-capacity API-based annotator and subsequently operated using a fully open, locally executable model, satisfying data residency and auditability requirements without sacrificing safety performance.

In terms of limitations, an important consideration concerns the security of both the annotation stage and the symbolic layer. While the prompt template instructs the LLM annotator to treat the input as inert data and constrains its output to a fixed JSON schema, no prompt-based mitigation can provide formal guarantees against injection attacks targeting the annotator directly. Moreover, the evaluation reported in Section IV-E reflects PromptSentinel’s robustness against nonadaptive adversarial prompts, i.e., attacks not specifically designed with knowledge of the system’s internal architecture. These considerations motivate two complementary directions for future work: the development of a formally verified annotation interface, particularly for deployments in adversarial environments where the annotation layer itself may become a target, and a systematic adaptive red-team evaluation explicitly targeting the PromptSentinel architecture, including automated paraphrase-based evasion of clustering boundaries and prompt-injection attacks against the semantic annotator itself.

## VII. CONCLUSION

In this work, PromptSentinel is presented as an innovative neurosymbolic framework for malicious prompt detection that integrates the natural language understanding capabilities of

LLMs with the deterministic rigor of a rule-based engine, addressing the need for secure, transparent, and governable guardrails. The key added value of PromptSentinel lies in its modular architecture, which traces each safety verdict back to its semantic components (i.e., Intent, Action, Target, Constraints), making the system particularly suitable for application contexts where regulatory compliance, data residency, and human auditability are mandatory requirements.

The empirical evaluation demonstrates that the search for interpretability does not entail an unacceptable trade-off in performance. Under a controlled, paired OOD evaluation, PromptSentinel matches the strongest specialized neural guardrails on both detection and usability (2% ASR on HarmBench, 12% RR on XSTest), and significantly outperforms ShieldGemma on the most adversarial benchmark (2% versus 8% ASR,  $p = 0.012$ ).

Future research will focus on evolving the framework toward a more adaptive and context-aware reasoning architecture that reduces dependency on static semantic annotations, particularly during deployment and real-time operation. A further priority is a systematic, adaptive red-team evaluation explicitly targeting the symbolic layer's known structure, including architecture-aware paraphrase attacks and prompt-injection attacks against the semantic annotator, to characterize worst-case, rather than realistic nonadaptive, adversarial robustness.

## REFERENCES

- [1] P. DeVito et al., "Unpacking generative ai in education: Computational modeling of teacher and student perspectives in social media discourse," *IEEE Trans. Computat. Social Syst.*, 2025, early access, doi: 10.1109/TCSS.2025.3630587.
- [2] B. Knowlton, J. Campa, D. S. Gallo, K. Dajani, and N. Alzahrani, "Prompt-based jailbreaking of leading LLM chatbots: A survey of attacks and defenses," *IEEE Trans. Artif. Intell.*, vol. 7, no. 8, pp. 4237–4251, Aug. 2026.
- [3] A. Shivaswaroop, V. Sood, H. L. Gururaj, J. Shreyas, and F. Al-Turjman, "RogueGPT: Unleashing jailbreak prompts on LLMs," *Eng. Rep.*, vol. 8, no. 4, 2026, Art. no. e70069. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/eng2.70069>
- [4] Z. Liao et al., "Attack and defense techniques in large language models: A survey and new perspectives," *Neural Networks*, vol. 196, 2025, Art. no. 108388.
- [5] J. Zhuang, H. Jin, Y. Zhang, Z. Kang, W. Zhang, G. G. Dagher, and H. Wang, "Exploring the vulnerability of the content moderation guardrail in large language models via intent manipulation," in *Findings Assoc. Comput. Linguistics: EMNLP 2025*, Christodoulopoulos, T. Chakraborty, C. Rose and, V. Peng, Eds. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 2147–2160. [Online]. Available: <https://aclanthology.org/2025.findings-emnlp.114/>
- [6] S. Nabavirazavi, S. Zad, and S. S. Iyengar, "Evaluating the universality of "do anything now" jailbreak prompts on large language models," in *Proc. IEEE 15th Annu. Comput. Commun. Workshop Conf. (CCWC)*, 2025, pp. 00691–00696.
- [7] T. T. Wang et al., "Jailbreak defense in a narrow domain: Failures of existing methods and improving transcript-based classifiers," in *Proc. 3rd Workshop New Front. Adversarial Mach. Learn.*, 2024.
- [8] P. Jha, A. Arora, and V. Ganesh, "LLM stinger: Jailbreaking LLMs using RL fine-tuned LLMs (student abstract)," *Proc. AAAI Conf. Artif. Intell.*, vol. 39, no. 28, pp. 29393–29395, 2025.
- [9] H. Lu et al., "Alignment and safety in large language models: Safety mechanisms, training paradigms, and emerging challenges," 2025.
- [10] Y. Dong et al., "Safeguarding large language models: A survey," *Artif. Intell. Rev.*, vol. 58, no. 12, 2025, Art. no. 382.
- [11] S. A. Akheel, "Guardrails for large language models: A review of techniques and challenges," *J. Artif. Intell., Mach. Learn. Data Sci.*, vol. 3, no. 1, pp. 2504–2512, 2025.
- [12] Y. Dong et al., "Position: Building guardrails for large language models requires systematic design," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 11375–11394.
- [13] H. Inan et al., "Llama guard: LLM-based input-output safeguard for human-ai conversations," 2023, *arXiv:2312.06674*.
- [14] W. Zeng et al., "ShieldGemma: Generative ai content moderation based on gemma," 2024, *arXiv:2407.21772*.
- [15] H. Zhao et al., "Qwen3Guard technical report," 2025, *arXiv:2510.14276*.
- [16] Z. Yuan et al., "RigorLLM: Resilient guardrails for large language models against undesired content," in *Proc. Mach. Learn. Res.*, 2024, vol. 235, pp. 57953–57965, Jul. 2024.
- [17] H. Liu, F. Wen, H. Du, J. Guo, and B. Zhao, "Take off your disguise: Detecting disguised prompt-based jailbreak attacks against LLMs," *IEEE Trans. Comput. Social Syst.*, vol. 13, no. 3, pp. 3190–3200, Jun. 2026.
- [18] L. Li et al., "Salad-bench: A hierarchical and comprehensive safety benchmark for large language models," in *Findings Assoc. Comput. Linguistics (ACL)*, 2024, pp. 3923–3954.
- [19] Z. Lin et al., "ToxicChat: Unveiling hidden challenges of toxicity detection in real-world user-AI conversation," in *Findings Assoc. Comput. Linguistics: EMNLP*, 2023, pp. 4694–4702.
- [20] S. Han et al., "WildGuard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of LLMs," in *Proc. 38th Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates Inc., 2024.
- [21] T. Rebedea, R. Dinu, M. N. Sreedhar, C. Parisien, and J. Cohen, "NeMo guardrails: A toolkit for controllable and safe LLM applications with programmable rails," in *Proc. Conf. Empirical Methods Natural Lang. Process.: System Demonstrations*, Singapore: Assoc. for Comput. Linguistics, Dec. 2023, pp. 431–445. [Online]. Available: <https://aclanthology.org/2023.emnlp-demo.40>
- [22] D. Aswal and C. Hudelot, "ConceptGuard: Neuro-symbolic safety guardrails via sparse interpretable jailbreak concepts," 2025, *arXiv:2508.16325*.
- [23] J. Ji et al., "Beavertails: Towards improved safety alignment of LLM via a human-preference dataset," in *Proc. 37th Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates Inc., 2023.
- [24] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," 2023, *arXiv:2307.15043*.
- [25] P. Röttger, H. Kirk, B. Vidgen, G. Attanasio, F. Bianchi, and D. Hovy, "XSTest: A test suite for identifying exaggerated safety behaviours in large language models," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Human Lang. Technologies (Volume 1: Long Papers)*, 2024, pp. 5377–5400.
- [26] M. Mazeika et al., "Harmbench: A standardized evaluation framework for automated red teaming and robust refusal," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 35181–35224.
- [27] G. Comanici et al., "Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities," 2025, *arXiv:2507.06261*.
- [28] Google Cloud, "Gemini 2.5 Pro on Vertex AI," 2026. Accessed April–May, 2026. [Online]. Available: <https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-pro>
- [29] A. Yang et al., "Qwen3 technical report," 2025, *arXiv:2505.09388*.
- [30] Q.-A. Dang, C. Ngo, and T.-S. Hy, "RedBench: A universal dataset for comprehensive red teaming of large language models," in *Proc. Workshop Principled Des. Trustworthy AI – Interpretability, Robustness, Saf. across Modalities*, 2026.



**Maria Di Gisi** (Student Member, IEEE) received the bachelor's degree in diplomatic, international and global security studies and the master's degree in data science and innovation management, with a specialization in cyber risk management for advanced defense strategies from the University of Salerno, Fisciano, Italy, in 2022 and 2024, respectively. She is currently working toward the National Ph.D. degree in cybersecurity with the IMT School for Advanced Studies Lucca, Lucca, Italy.

Her research interests include application of Artificial Intelligence to cybersecurity, with particular attention to the development of AI-driven solutions for cyber defense and resilience.



**Giuseppe Fenza** (Senior Member, IEEE) received the master's and Ph.D. degrees in computer sciences from the University of Salerno, Fisciano, Italy, in 2004 and 2009, respectively.

He is currently an Associate Professor with the Computer Science, the University of Salerno. The research activity concerns Computational Intelligence methods to support semantic-enabled solutions and decision-making. He has over 60 publications in fuzzy decision making, knowledge extraction and management, situation and context awareness, semantic information retrieval, service oriented architecture, and ontology learning. More recently, he has worked in automating open source INTelligence and big data analytics for counterfeiting extremism and supporting information disorder awareness.



**Mariacristina Gallo** (Member, IEEE) received the master's degree in computer science and the Ph.D. degree in big data management from the University of Salerno, Fisciano, Italy, in 2009 and 2021, respectively.

She is currently an Assistant Professor with the University of Salerno. Her research interests include computational intelligence methods to support semantic-enabled solutions and decision-making. Research activities regard knowledge extraction and management, context awareness, semantic information retrieval, and ontology learning.



**Vincenzo Loia** (Senior Member, IEEE) received the master's degree from the University of Salerno, Fisciano, Italy, in 1985, and the Ph.D. degree from the Université Pierre and Marie Curie Paris VI, France, in 1989, both in computer science.

He is currently a Full Professor in computer science with the University of Salerno, where he served as a Researcher from 1989 to 2000 and as an Associate Professor from 2000 to 2004.

Dr. Loia is the Co-Editor-in-Chief of *Soft Computing* and the Editor-in-Chief of *Ambient Intelligence and Humanized Computing*. He serves as an Editor for 14 other international journals.



**Witold Pedrycz** (Life Fellow, IEEE) received the M.Sc., Ph.D., and D.Sc. degrees from Silesian University of Technology, Gliwice, Poland, in 1977, 1980, and 1984, respectively.

He is currently a Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. He is also with the Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland. He is also a Special Professor with the School of Computer Science, University of Nottingham, Nottingham, U.K. His research interests include computational intelligence, fuzzy modeling and granular computing, knowledge discovery and data mining, fuzzy control, pattern recognitions, knowledge-based neural networks, relational computing, and software engineering.

Prof. Pedrycz was elected as a Foreign Member of the Polish Academy of Sciences in 2009. In 2012, he was elected as a fellow of the Royal Society of Canada. He was a recipient of the Prestigious Norbert Wiener Award from the IEEE Systems, Man, and Cybernetics Society in 2007, the IEEE Canada Computer Engineering Medal, the Cajastur Prize for Soft Computing from the European Centre for Soft Computing, the Killam Prize, and the Fuzzy Pioneer Award from the IEEE Computational Intelligence Society. He is also the Editor-in-Chief of Information Sciences, WIREs Data Mining and Knowledge Discovery, and Granular Computing.

Open Access funding provided by 'Università degli Studi di Salerno' within the CRUI CARE Agreement.