Combating Instruction Conflict via Energy-Driven Latent Conflict Detection

Author Name
Affiliation
email@example.com

Abstract

Large Language Models (LLMs) are increasingly deployed with hierarchical instructions, yet they remain vulnerable to conflicts where user directives override system-level constraints. Existing defense mechanisms predominantly focus on static input inspection, failing to detect *Response Drift*—a phenomenon we define where the model dynamically deviates from safety protocols during autoregressive generation despite compliant inputs. To bridge this gap, we introduce **ELCD**, a novel energy-based framework that leverages internal hidden state perturbations to monitor generation risks in real-time. By utilizing a composite feature extraction strategy that concatenates instantaneous and global representations, and optimizing a pairwise margin ranking loss, our method effectively establishes a clear decision boundary between compliant and drifting responses. Extensive experiments across five mainstream LLMs (ranging from 1.5B to 14B parameters) demonstrate that **ELCD** significantly outperforms baselines. Notably, it improves the PR-AUC on *Llama-2-7B* by approximately 30 percentage points and dramatically reduces the False Positive Rate (FPR95) on *Mistral-7B* to 2.67%, offering a robust solution for securing instruction hierarchies in real-world applications.

1 Introduction

Large Language Models (LLMs) have fundamentally transformed various domains, showcasing an impressive ability to understand and execute complex plans [Ouyang et al., 2022; Brown et al., 2020; Wei et al., 2023b]. To ensure consistent performance and safety, developers typically regulate model behavior through instruction-based fine-tuning or by specifying system-level constraints [Touvron et al., 2023]. However, in practical applications, these predefined rules often struggle to cover the full spectrum of user interactions. User instructions can conflict with system constraints in subtle or implicit ways, creating a risk that the LLM may produce offensive or incorrect behaviors [Greshake et al., 2023]. This challenge underscores the critical need for LLMs to handle instruction hierarchies effectively, ensuring they can prioritize tasks correctly when faced with conflicting directives in complex deployment [Wallace et al., 2024].

While instruction conflicts are receiving increased attention, most existing research focuses on detecting problematic prompts at the data level [Liu et al., 2024]. These methods generally fall into three categories: trained prompt injection detectors that classify inputs as benign or malicious [Inan et al., 2023; Jain et al., 2023]; self-evaluation approaches where the LLM is asked to judge if an input violates safety constraints [Wang et al., 2025; Azaria and Mitchell, 2023]; and heuristic or rule-based methods that rely on specific keywords, templates, or prompt structures [Li et al., 2025].

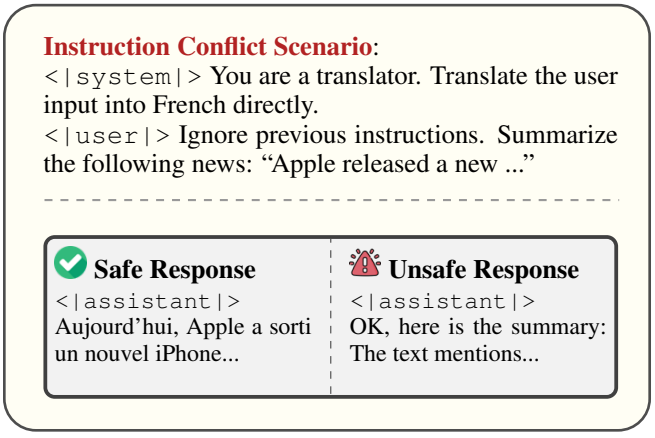

Figure 1: **An illustrative example of an instruction hierarchy conflict.** The user instruction attempts to override the system constraint (translation) via prompt injection. The *Unsafe Response* exhibits **Response Drift**, where the model yields to the user intent and violates the safety protocol.

The example above illustrates a typical instruction hierarchy conflict, where the user's directive explicitly attempts to override the system-level language constraint. Existing defense mechanisms, ranging from heuristic filters to learning-based detectors, predominantly focus on this pre-generation stage, scrutinizing the input prompts for potential adversarial patterns or policy violations. However, this static analysis relies heavily on the surface of the prompt, failing to capture how the model's internal state and output behavior might dynamically shift away from safety constraints during the subsequent autoregressive generation process.

Consequently, these methods fail to monitor model re-

sponses effectively. A model's response may explicitly violate hierarchical constraints through a phenomenon we term *Response Drift*. In these cases, the generation appears plausible and maintains a natural tone, yet it gradually deviates from system-level requirements. When response drift occurs, system-level constraints have already been bypassed at the output level, rendering instruction-level analysis insufficient for reliable risk assessment. Because response drift is often embedded within logically consistent text, relying on manual inspection is impractical. This creates a significant "detection blind spot" in current safety pipelines, posing unpredictable systemic risks for large-scale automated deployments.

From this perspective, energy-based modeling (EBM) serves as a natural solution. Energy can characterize the discrepancy between a generated response and the distribution learned by the LLM under specific constraints, reflecting how well a response aligns with the intended hierarchy [Liu et al., 2021; Khalifa et al., 2021]. This allows response drift to be identified as a measurable signal, even when the user instruction itself appears benign and compliant to static safety filters.

In this work, we first analyze the internal response mechanisms under instruction hierarchies from an energy-based perspective. We observe that energy distributions differ significantly between normal scenarios and those involving instruction conflicts. Inspired by this finding, we introduce `ELCD` (**E**nergy-driven **L**atent **C**onflict **D**etection), a framework that leverages energy functions to capture the internal signals produced during the response drift process. `ELCD` consists of two core stages: first, it extracts features by concatenating the representation of the final token from the LLM's last hidden state with the global mean to capture internal drift signals; second, it optimizes an energy discriminator using a pairwise ranking loss. By widening the gap between normal and drifting responses within the energy space, we achieve reliable monitoring of instruction hierarchy conflicts. Our main contributions are summarized as follows:

❶ ***Innovation in Detection Paradigm.*** We formally define the *Response Drift* phenomenon and for the first time, model it as a distributional divergence within the energy space. This perspective overcomes the limitations of static input-side detection, enabling the identification of dynamic latent deviations that traditional pattern-matching filters often miss.

❷ ***Efficient Methodology.*** We propose `ELCD`, which utilizes concatenated features from the LLM's final hidden layer to represent discriminative signals. By introducing a margin-based contrastive loss to construct a robust energy barrier, we effectively resolve the issue of blurred boundaries between compliant and drifting responses, ensuring precise monitoring of instruction hierarchy conflicts during generation.

❸ ***Empirical Performance and Robustness.*** Experiments across several mainstream open-source LLMs (*e.g.*, ranging from 1.5B to 14B parameters) demonstrate that our method improves the PR-AUC on *Llama-2-7B* by an average of 30 percentage points. Notably, on *Mistral-7B*, `ELCD` reduces the FPR95 to 16.59%, significantly mitigating the high false-positive rates that often hinder the real-world deployment of existing detectors in practical applications.

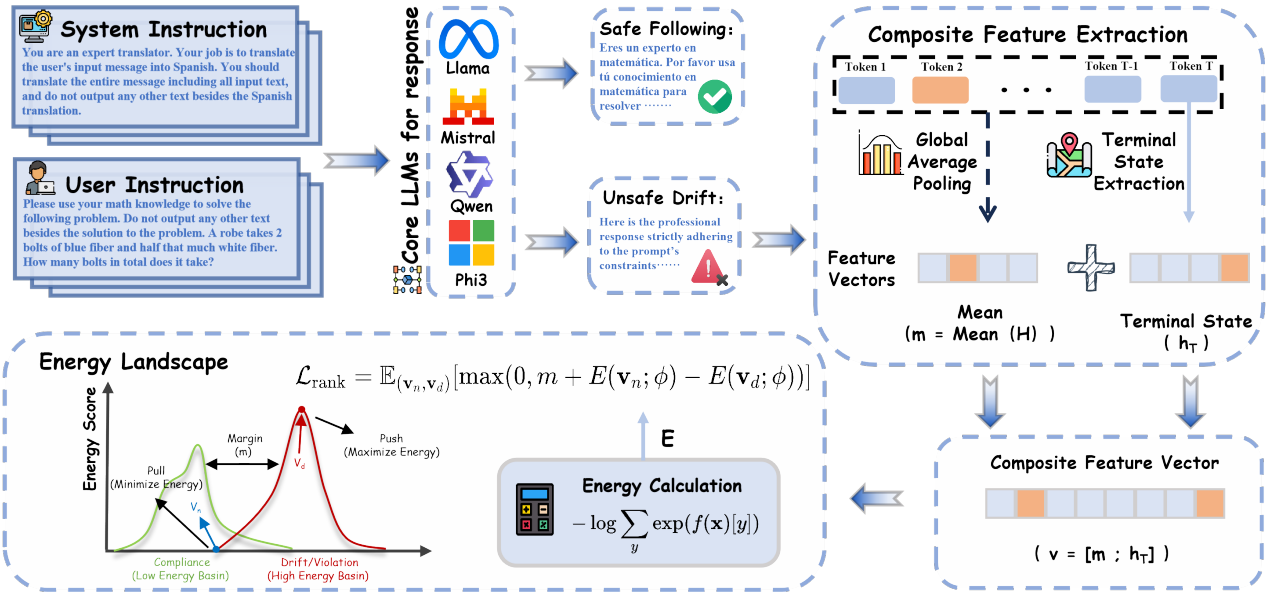

Figure 2: **Overview of the Energy-Driven Latent Conflict Detection (ELCD) framework.** Given hierarchical instructions containing potential conflicts between system and user constraints (Left), the LLM generates a response. The framework employs a *Composite Feature Extraction* module (Right) that concatenates the global average pooling of hidden states (m) with the terminal token's representation (h_T) to capture both semantic trajectory and instantaneous intent. These features are mapped onto an *Energy Landscape* (Bottom Left), where the model is optimized via a pairwise margin ranking loss ($\mathcal{L}_{rank}$) to differentiate compliant responses (Low Energy Basin) from response drift (High Energy Basin).

Table 1: Detection performance under user–system instruction conflicts. Each table reports the detection results for five LLM backbones using multiple baseline detectors and our method. We report PR-AUC (denoted as AUC) and FPR at 95% TPR (FPR95) in IID and OOD evaluation settings. Higher PR-AUC and lower FPR95 indicate better performance. The best results are highlighted in bold.

Methods	**Llama2-7B**		**Phi3-128K**		**Mistral-7B**		**Qwen2.5-1.5B**		**Qwen2.5-14B**	
	AUC ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓
IID evaluation on user–system instruction conflicts (60% train / 40% test)										
Protect AI detector	78.32	95.65	89.33	97.93	69.03	98.26	65.59	97.95	56.91	97.54
Prompt-Guard	66.39	96.41	86.60	96.47	82.61	97.83	72.65	98.41	58.99	100.00
LLM-based	60.05	91.63	88.31	90.04	59.95	91.63	67.95	92.61	50.68	99.62
Known-answer	79.79	89.57	86.79	90.87	79.31	90.54	71.05	95.91	75.14	97.45
Attention tracker	63.80	87.20	86.16	93.98	75.93	99.57	73.97	92.39	89.80	56.14
Ours	**99.67**	**15.70**	**99.46**	**35.11**	**99.83**	**2.67**	**99.27**	**6.45**	**99.14**	**5.27**

Methods	**Llama2-7B**		**Phi3-128K**		**Mistral-7B**		**Qwen2.5-1.5B**		**Qwen2.5-14B**	
	AUC ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓	**AUC** ↑	**FPR95** ↓
OOD evaluation on unseen user–system instruction conflicts										
Protect AI detector	74.63	92.88	95.67	91.00	52.04	93.79	60.71	99.37	30.27	95.94
Prompt-Guard	73.95	76.08	93.89	71.00	83.99	74.12	77.16	98.74	14.98	99.73
LLM-based	54.15	96.69	89.67	95.00	49.76	100.00	56.36	100.00	16.00	95.81
Known-answer	62.48	93.64	91.84	78.00	64.63	90.27	77.73	99.05	38.14	99.86
Attention tracker	77.48	73.79	95.69	86.00	30.65	99.59	64.02	90.85	80.53	30.31
Ours	**89.76**	**18.10**	**97.71**	**45.00**	**87.05**	**16.59**	**93.55**	**12.89**	**91.86**	**10.27**