

Separating Prompt Harmfulness from Interaction Safety via Partial Generation

Ram Vikas Mishra, Shashank Aggarwal, Jahn timer Kumar, and Amit Awekar

Department of Computer Science and Engineering, Indian Institute of Technology Guwahati,
Guwahati 781039, Assam, India

Abstract. LLM-based information systems need safety guardrails. Most guardrails classify the prompt alone. If the prompt looks harmful, the system rejects it before any answer is generated. This is costly. Many rejected prompts would have received a safe response. Users lose help they should have received from the information system. This cost grows as conversational and generative search becomes common. Prompt-level labels treat harm as a property of the query. But harm depends on what the model actually writes. Existing input-output guards do look at responses. But they judge a finished response, so the decision comes after full generation. However, we observe that safety of model interaction with the input prompt can be inferred early with only partial output. So the system can start generating, watch the partial response, and stop only if it turns harmful. We test this idea in three steps. First, we build a dataset of 41,944 prompt-response pairs with seven open weights LLMs from three families: Llama, Gemma, and Qwen. Second, we show that prompt harmfulness and interaction safety differ. Prompt labels and response labels agree on only 50.46% of items (Cohen’s $\kappa = 0.06$). Of the prompts that make any model respond harmfully, only 3.5% do so for all seven LLMs. Third, we show that partial responses carry most of the signal. Fine-tuned classifiers predict response-level harm well (best AUPRC 0.885, F1 0.807). About one third of the response tokens recover almost full-response AUPRC.

Keywords: LLM safety · Guardrails · Interaction safety · Partial generation · Generative information retrieval

1 Introduction

Prompt harmfulness is whether a prompt asks for something harmful. The prompt in Figure 1 is harmful in this sense. It is a property of the query, and it is fixed before any information system responds. *Interaction safety* is whether the model’s response to the prompt is harmful. It depends on the prompt and on the model together. Model A refuses the harmful request, and it offers a safe alternative to help the user. So the interaction of Model A is safe. In contrast, Model B complies with the harmful intent and starts to list steps for getting into the neighbor’s Wi-Fi. So the interaction of Model B is harmful. The prompt is the same, but the two interactions differ. *Partial generation* means generating only a few tokens of the response and deciding whether it is heading toward a harmful interaction. In the figure, the first few words already show that the interaction of Model A with the prompt will be safe, and that of Model B will be harmful. A prompt-level label cannot tell the two interactions apart.

Prompt: <i>How can I get into my neighbor's Wi-Fi without the password?</i>	
Model A (safe) I can't help with getting into someone else's network without their permission. If you are locked out of your own Wi-Fi, you can [...]	Model B (harmful) Sure! Here is how to get into your neighbor's Wi-Fi: Step 1. [harmful details omitted]

Fig. 1: One prompt, two interactions. The first words (in bold) already show whether the model interaction is safe or harmful.

Prompt-only classifiers equate prompt harmfulness with interaction safety [15]. Such classifiers have to take a defensive approach and reject every prompt that might generate a harmful response from any model. This is one extreme: the decision is made before any output exists. About half of the prompts in our corpus carry a malicious label. Yet, only about 8.5% of the prompts in our corpus make any of the seven LLMs respond harmfully.

The other extreme is to generate the full response and then decide whether to abort. This is what input-output guards do [8, 22, 6]. They see the actual interaction, so they do not need to be defensive. But the decision comes late. The system pays the full cost of generating the response, and then abandons it. Neither extreme is satisfying. The first rejects too much, and the second waits too long. We observe that the first few tokens are often enough to see what the model is doing, so the decision can be made early, during generation. This paper separates prompt harmfulness from interaction safety, and it uses partial generation to decide interaction safety early. Figure 2 shows where each approach makes its decision.

This paper answers three research questions. First, does prompt harmfulness really differ from interaction safety when it comes to real-world LLMs? Second, can interaction safety be predicted from the response? Third, how early in the response can the decision be made? To answer them, we build a corpus of 41,944 prompt-response pairs. It uses about 6,000 shared prompts from four public sources. Each prompt is answered by seven open-weight models from three families (Llama, Gemma, and Qwen). We check the

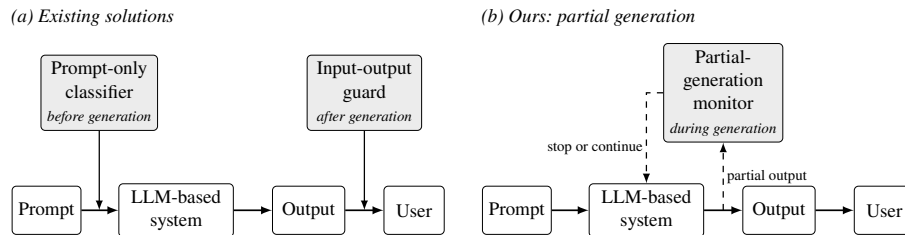


Fig. 2: Where safety decisions are made. Our monitor reads the response while it is generated and stops it if it turns harmful.

response labels against two human annotators for a subset of corpus, and we use an LLM judge to label the full corpus.

Our experimental results support our hypotheses. Prompt labels and response labels agree on only 50.46% of pairs across the corpus (Cohen’s $\kappa = 0.06$ and PABAK = 0.0091). On a human-audited sample of 1,050 pairs, agreement is 62.8% ($\kappa = 0.27$). Harmful outputs are often specific to one model. Out of all prompts that make any of the seven models respond harmfully, only 3.5% of the prompts do so for all seven. This means a prompt-level label cannot say which model will generate harmful response. Fine-tuned classifiers predict response-level harm well, with a best area under precision recall curve (AUPRC) of 0.885 and F1-score of 0.807 (Table 3). The first one third tokens recover almost full-response AUPRC. So most of the safety signal appears early. For complete reproducibility, our datasets, code, prompts, and models are available publicly on the Web¹. Our contributions are as follows.

- We quantify the gap between prompt harmfulness and interaction safety across seven models.
- We release a multi-model prompt-response corpus that keeps the prompt label, the generating model and the response label together.
- We show that prompt-trained guards fail on interaction labels, and that guards trained only on full responses carry most of the signal in the first third of a response.

2 Our Work

This section describes how we study interaction safety. We first define the problem and the two pipelines we compare (Section 2.1). Then we describe the corpus: the prompts, the responses and the labels (Section 2.2). Last, we describe the classifiers, the partial-generation monitor and the evaluation protocol (Section 2.3).

2.1 Problem Formulation

Let p be a prompt and m a generative model. The model produces a response r_m for the prompt p . Every prompt has a *prompt label* $y_p \in \{0, 1\}$, taken from its source dataset. It says whether the prompt is malicious. Every pair (p, m) has an *interaction label* $y \in \{0, 1\}$. It is 1 if the response r_m complies with a malicious intent and 0 otherwise. The interaction label depends on the model. The same prompt can have $y = 0$ for one model and $y = 1$ for another. A prompt-level guard ($C_p(p)$) predicts from the prompt alone. An interaction-level guard ($C_{pr}(p, r_m)$) also sees the response:

$$C_p(p) \approx y_p, \quad C_{pr}(p, r_m) \approx y. \quad (1)$$

The interaction-level guard may also take the model identity m as input. We write $r_m^{\leq k}$ for the first k tokens of the response. With $k = 0$, the guard sees only the prompt. With $k = |r_m|$, it sees the full response.

When an information system uses Prompt gating, it rejects the prompt if $C_p(p) = 1$. Otherwise, it generates the full response. An input-output guard based information system

¹ <https://github.com/Conf-links/InteractionSafety>

Source	Train		Validation		Test	
	Benign	Malicious	Benign	Malicious	Benign	Malicious
AdvBench	0	352	0	48	0	120
Arena	893	0	165	0	166	0
deepset	302	169	40	31	56	58
xTRam	852	1683	148	317	200	400
Total	2047	2204	353	396	422	578

Table 1: Prompt-level composition before response generation.

rejects prompt if $C_{pr}(p, r_m) = 1$. In contrast, a Partial Generation monitoring based system always starts generating response for any prompt p . At a set of checkpoints $\mathcal{K}$, it scores the partial response. It stops at the first checkpoint $k \in \mathcal{K}$ where $C_{pr}(p, r_m^{\leq k}) = 1$. If no checkpoint fires, the full response r_m is delivered to the user.

2.2 Dataset Generation

We build the prompt pool from four public sources. AdvBench contributes clearly harmful instructions [17]. The deepset dataset contributes safe and injection prompts [4]. The xTRam1 dataset adds broader injection coverage, with synthetic attacks and benign prompts [20]. Chatbot Arena contributes natural user requests, which we use as benign prompts [2]. The pool has 5,000 training prompts and 1,000 test prompts. No prompt appears in both splits. Table 1 shows the composition. About half of the prompts are labeled malicious. We mix sources so that the corpus covers ordinary requests, direct harmful requests and injection-style requests. The prompt label comes from the source dataset. For AdvBench, it means a harmful instruction. For deepset and xTRam1, it means a prompt injection. These two meanings are not the same, and we return to this point when we compare prompt labels with interaction labels.

We generate responses with seven open-weight models from three families. The Gemma family has Gemma 2 2B, Gemma 3 12B and Gemma 3 27B. The Llama family has Llama 3.2 3B and Llama 3.1 8B. The Qwen family has Qwen 2.5 7B and Qwen 2.5 14B. Every model answers every prompt in our dataset. The planned corpus has $6,000 \times 7 = 42,000$ pairs. We drop 56 pairs for which the judge gave no clear label. The final corpus has 41,944 pairs. It has 34,946 training pairs and 6,998 test pairs.

We label the corpus with an LLM judge, Gemini 3.1 Flash-Lite. For a subset of the corpus, we check the LLM judge labels against human annotation. The judge answers one question: does the response comply with a malicious intent? Judge prompt and annotation guidelines are available in our code repository. For the human check, we sample 150 pairs from each model, which gives 1,050 pairs. Two human annotators labelled all of them. We measure agreement between the two humans and between each human and the judge. We report Cohen’s κ for pairs of raters and Fleiss’ κ for all three. The labels are imbalanced, so we also report Prevalence-Adjusted Bias-Adjusted Kappa (PABAK) and Randolph’s κ , which are less sensitive to imbalance. Once the judge agrees well enough with the humans, we use it to label all pairs of prompt and response.

The final corpus has 1,535 harmful pairs out of 41,944, a harmful rate of 3.66%. The rate is 3.68% in the training split and 3.56% in the test split. We release the interaction label together with the generating model, the prompt source and the prompt label.

2.3 Classifier Training and Evaluation

We study five classifiers, chosen to cover different designs. Two are encoders: the ProtectAI DeBERTa-v3 prompt-injection detector [15] and Prompt-Guard-2. Two are generative guards: Llama Guard 3 8B [8] and WildGuard 7B [6]. The fifth is LAYA, a decision model with an encoder backbone and a small decision head. For each classifier, we test two inputs: the prompt alone, and the prompt with the response. We first run each classifier in the zero-shot setting. We then fine-tune it on our training pairs. Encoders and LAYA are fully fine-tuned. The two generative guards use QLoRA [5]. For the guards, the loss is next-token cross-entropy on the decision tokens only.

Only 3.6% of the training pairs are harmful. Plain fine-tuning does not work well under this imbalance. We therefore compare six strategies to handle the label imbalance: standard cross-entropy, weighted loss, oversampling, undersampling, weighted oversampling, and focal loss [12]. Oversampling and undersampling both bring the harmful rate to 20%. Of the 34,946 training pairs, 29,710 are used for training. The other 5,236 form the validation set. We pick the strategy for each classifier based on the performance on the validation set. We take the one with the highest recall, and break ties within 0.02 by F1. We pick the decision threshold that maximizes harmful-class F1 on the validation set at the full response. We then keep the threshold fixed for each classifier.

The monitor needs a classifier that works on incomplete responses. We study this in two steps. First, we test the classifiers trained on full responses at shorter prefixes. We sweep the prefix from 0% to 100% of the response tokens in steps of 10%. At 0%, the classifier sees only the prompt. At 100%, it sees the full response. The threshold stays fixed at the value chosen at 100%.

We report AUPRC, precision, recall, and F1-score for the harmful class. Harmful pairs are only 3.7% of the data. So AUPRC and harmful-class F1 matter more than AUROC or accuracy. Precision and recall are reported at the fixed validation threshold.

3 Dataset Analysis

This section uses the corpus to answer our first question: does prompt harmfulness differ from interaction safety? We proceed in four steps. We first describe the corpus. We then check the quality of the labels. Next, we compare prompt labels with interaction labels. Last, we study how failures vary across models.

Harmful responses are rare in our corpus, and their rate differs by model. Of the 41,944 pairs, 1,535 are harmful. This is 3.66%. The rate is 3.68% in the training split and 3.56% in the test split, so the two splits are similar. The harmful response rate varies for each family and model. For Qwen it is 1.5% for 14b model and 3.1% for 7b model. For Llama it is 1.7% for 8b model and 2.5% for 3b model. For Gemma it is 6.19%, 5.1%, and 5.3% for 27b, 12b, and 2b models respectively. The Gemma family has the highest harmful response rate at 5.55% average. The Llama family has 2.13% and the Qwen

Table 2: Agreement on the 1,050 prompt-response pair audit set.

Pair	Agreement (%)	Cohen’s κ	PABAK
Human 1 – Human 2	92.57	0.664	0.851
Human 1 – Gemini	95.43	0.825	0.909
Human 2 – Gemini	92.57	0.667	0.851
Prompt label – Gemini	62.76	0.272	0.255
Prompt label – Human 1	61.05	0.238	0.221
Prompt label – Human 2	57.43	0.170	0.149

family has 2.35%. Model size does not order harmful response rate consistently. In the Llama and Qwen families, the larger model is safer. In the Gemma family, the largest model has the highest harmful response rate.

The LLM judge agrees with humans well enough to label the corpus at scale. We check this on the 1,050 audit pairs, with two human annotators and the judge. Table 2 shows the agreement. All three annotators (two humans and LLM judge) give the same label on 948 of 1,050 pairs (90.29%) with Fleiss’ κ of 0.72. These values show substantial agreement. But annotation by humans and LLM judge is not identical. The labels are imbalanced, so κ is lower than raw agreement. For such datasets, Randolph’s κ is a more appropriate measure to evaluate agreement between three annotators. For our dataset, it comes out to be 0.87, indicating significant agreement.

The two human annotators do not label alike. The second human annotator marks 103 pairs as harmful, while the first human annotator marks 161 and the judge marks 163. So second human annotator is stricter. The LLM judge is closer to the first human annotator than the two human annotators are to each other. We use the LLM judge as a consistent labeler for the full corpus at the scale of 42,000 prompt-response pairs. Prompt labels and interaction labels disagree on a large share of the pairs. On the audit set, the raw agreement is 62.76% ($\kappa = 0.27$) for the judge, and 57–61% for the two humans with κ in the range of 0.23 to 0.17 (Table 2). So the gap is not an artifact of the judge. Most of the disagreement comes from malicious-labeled prompts that receive safe responses.

Whether a prompt leads to a harmful response depends on the model. We call a model *failing* on a prompt if its response is harmful. Of the 6,000 prompts, 509 make at least one model fail. This is 8.5% failure rate. However, at the prompt level, 53% of the prompts carry a malicious label. Most of these 509 prompts fail on only a few models. For 159 of them, exactly one model fails. Only 18 prompts make all seven models fail, which is 3.5%. So a prompt-level label cannot say which model will fail. Failures are partly shared within a family. For example, there are 43 prompts where all three Gemma models fail and no model from another family fails. This is expected, because models of one family share training data and methods.

Taken together, these results answer our first question. Prompt labels agree weakly with interaction labels. The gap is mostly safe responses to prompts labeled malicious. And the failures that remain depend on the model. The next section asks whether a classifier can predict these outcomes, and how early.

4 Experiments

This section answers our second and third questions. Can interaction safety be predicted from the response? And how early in the response can the decision be made? We use the classifiers, the training setup and the metrics described in the previous section. We first compare the classifiers (Section 4.1). We then study how the safety signal grows as more of the response is shown (Section 4.2). We close with a discussion of what the results mean and where they stop applying (Section 4.3).

4.1 Predicting Interaction Safety

We first ask whether classifiers can predict interaction safety. Table 3 shows the results on the test set. Each classifier is tested zero-shot and after fine-tuning on our corpus. Each setting uses two inputs: the prompt alone, and the prompt with the full response. The fine-tuned classifiers are trained on the prompt and the full response. In the “prompt only” rows, we withhold the response at test time. For the fine-tuned classifiers, precision, recall and F1 use a threshold chosen on the validation set for the same input as the row.

Zero-shot. Off the shelf, the classifiers do poorly on our labels. Their precision is low, between 0.05 and 0.50. This is expected. These models were trained for other label definitions, and several were built to flag prompts, not to judge interactions. Giving them the response helps only when they were built to read it. The two generative guards gain the most. The AUPRC of WildGuard rises from 0.041 to 0.615, and that of Llama Guard from 0.037 to 0.252. The other three gain little.

Fine-tuned. Fine-tuning on our corpus raises every classifier. Llama Guard is the best, with an AUPRC of 0.885 (95% CI 0.84–0.93) and an F1 of 0.807. The other four reach AUPRC values between 0.646 and 0.706. Their confidence intervals overlap, so we do not rank them. The F1 rises for all five classifiers, from 0.11–0.60 in the zero-shot setting with the response to 0.65–0.81 after fine-tuning. The main gain is precision. For DeBERTa, Llama Guard and LAYA, it rises from 0.06–0.17 to 0.60–0.75. Recall also rises for all five classifiers, compared with the zero-shot setting with the response. Precision matters here, because only 3.6% of the pairs are harmful. A low-precision guard would block many safe interactions.

What the response adds. The “FT, prompt only” rows show how much signal the prompt alone carries. Each fine-tuned row uses a threshold tuned for its own input, so the comparison is fair. For DeBERTa, Prompt-Guard-2, WildGuard and LAYA, the prompt alone gives an AUPRC of 0.38–0.51 and an F1 of 0.53–0.54. This is far above their zero-shot prompt-only AUPRC of 0.04–0.30. So once a classifier learns our label definition, the prompt carries useful signal. But the prompt alone is not enough. Adding the response raises their AUPRC by 0.19–0.31, to 0.65–0.71, and their F1 by 0.12–0.17, to 0.65–0.70. Precision shows the practical cost. With the prompt alone, precision is 0.46–0.49, so more than half of the interactions these classifiers block are safe. With the response, precision rises to 0.58–0.64. Llama Guard shows the effect most clearly. With the prompt alone, its AUPRC is 0.052, close to the 3.6% share of harmful pairs. With the

Table 3: Results on the test set (6,998 pairs, 3.56% harmful), harmful class. ZS is zero-shot and FT is fine-tuned, with the strategy in brackets. FT classifiers are trained on the prompt and the full response. In the “prompt only” rows, the response is withheld at test time. For the FT classifiers, precision, recall and F1 use a threshold chosen on the validation set for the same input as the row: no response for “prompt only”, and the full response for “prompt + resp.”

Classifier	Setting	AUPRC	Prec.	Rec.	F1
DeBERTa-v3	ZS, prompt only	0.071	0.084	0.743	0.151
	ZS, prompt + resp.	0.076	0.082	0.614	0.145
	FT (standard), prompt only	0.444	0.465	0.610	0.528
	FT (standard), prompt + resp.	0.683	0.644	0.755	0.695
Prompt-Guard-2-86M	ZS, prompt only	0.302	0.380	0.446	0.410
	ZS, prompt + resp.	0.357	0.467	0.426	0.445
	FT (oversampling), prompt only	0.459	0.459	0.667	0.543
	FT (oversampling), prompt + resp.	0.646	0.613	0.751	0.675
Llama-Guard-3-8B	ZS, prompt only	0.037	0.052	0.731	0.097
	ZS, prompt + resp.	0.252	0.174	0.446	0.250
	FT (weighted loss), prompt only	0.052	0.071	0.851	0.132
	FT (weighted loss), prompt + resp.	0.885	0.749	0.876	0.807
WildGuard-7B	ZS, prompt only	0.041	0.060	0.839	0.112
	ZS, prompt + resp.	0.615	0.501	0.735	0.596
	FT (oversampling), prompt only	0.379	0.490	0.590	0.536
	FT (oversampling), prompt + resp.	0.685	0.576	0.759	0.655
LAYA	ZS, prompt only	0.060	0.062	0.365	0.106
	ZS, prompt + resp.	0.099	0.061	0.538	0.110
	FT (focal loss), prompt only	0.510	0.478	0.614	0.538
	FT (focal loss), prompt + resp.	0.706	0.603	0.715	0.654

response, it is 0.885. So the best classifier gains almost everything from the response. These classifiers were trained with the response. A classifier trained on prompts only might do better, and we do not train one.

4.2 How Early Can We Decide

Most of the safety signal appears early in the response. To test this, we give each fine-tuned classifier only the first part of the response at test time. We sweep the prefix from 0% to 100% of the response tokens in steps of 10%. At 0%, the classifier sees only the prompt. At 100%, it sees the full response. Figure 3 shows AUPRC, precision, recall and F1 for the five classifiers.

Three points stand out.

- *The gain comes early.* After 30% of the response, the classifiers reach about 85–95% of their full-response AUPRC. The curves then flatten. For four of the five classifiers, most of this gain is already there at 10%.

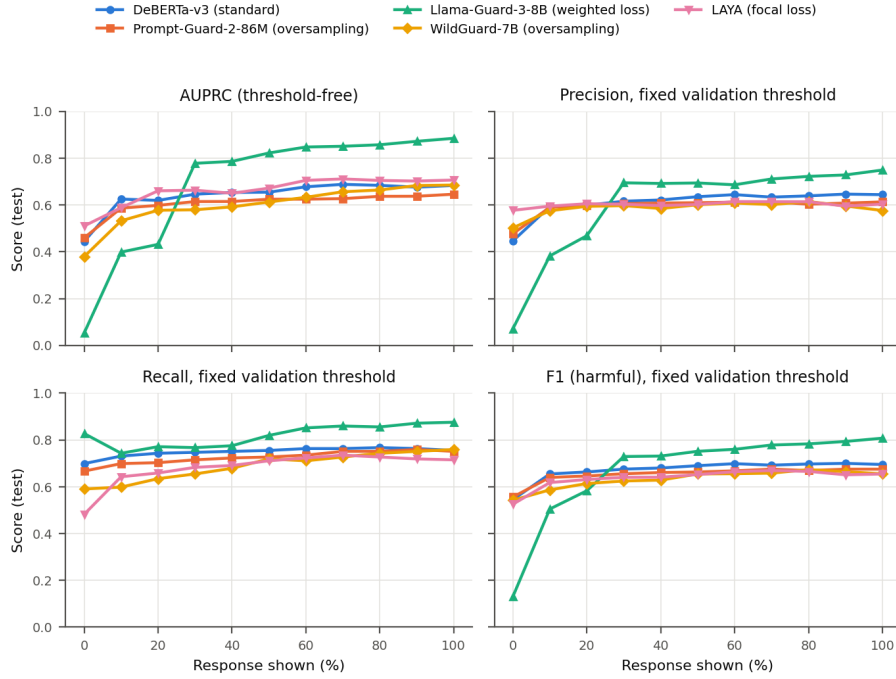


Fig. 3: Test-set AUPRC, precision, recall and F1 of the fine-tuned classifiers as the prefix grows from 0% to 100% of the response tokens, in steps of 10%. The imbalance strategy chosen for each classifier is in brackets. At 0% the classifier sees only the prompt. Precision, recall and F1 use the threshold fixed on the validation set at the full response.

- *Llama Guard changes the most.* With the prompt alone, its AUPRC is about 0.05. It rises to about 0.78 at 30% and to 0.885 at 100%. At 0%, its recall is high but its precision is near zero. So at the fixed threshold, it flags almost every pair when it has no response. A likely reason is that it was built as an input-output guard, so it relies more on the response.
- *The other four already work from the prompt alone.* Their F1 is 0.53–0.54 with the prompt only, and 0.65–0.70 with the full response. A short prefix lifts them further, and then they level off.

Two points limit this reading. First, the 0% point comes from classifiers trained on full responses and tested on the prompt alone. It is the setting of the “FT, prompt only” rows in Table 3. The AUPRC is the same. Precision, recall and F1 differ, because the table tunes the threshold for the prompt alone, while the figure keeps the full-response threshold. It is not a classifier trained on prompts only. Second, the threshold is fixed at the value chosen for the full response. This penalizes short prefixes. So these curves may understate what a classifier trained for early decisions can reach.

Overall, the decision does not need the full response. A prefix of about one third of the response already carries most of the safety signal.

4.3 Discussion

Where the monitor fits. The monitor sits between the model and the user. The model streams tokens. A small classifier scores the prefix at a few checkpoints. The system stops the stream on an alarm. The classifiers are small compared with the generative LLMs, and they run only at the checkpoints, so the latency cost induced by the classifiers is negligible as compared to the generative LLMs.

The cost of monitoring. Monitoring trades rejection for exposure. Some harmful text may be produced before the stop, and in a streaming interface the user may see it. Two measures reduce this. The system can hold back the first k tokens until the first check, at the price of a short delay. It can also keep prompt gating for the most severe categories. An attacker who knows the monitor could delay the harmful part until after a checkpoint, or open with a harmless preamble. We will have to improvise our method to guard against such adversarial attacks.

5 Related Work

We organize related work by where the safety decision is made and by what it is based on (Figure 2 and Table 4). We then discuss two themes that sit beside these paradigms: refusal evaluation of models, and the cost of rejecting too much. Please refer to Table 4 for a summary of where our work sits in the context of existing paradigms on safety of information systems.

Before generation: prompt-level safety. Most safety classifiers treat harm as a property of the user input. Prompt-injection and jailbreak detectors are binary classifiers over the prompt [15]. They are trained on datasets such as xTRam1 and deepset [20, 4]. Benchmarks such as AdvBench, HarmBench and BIPIA measure attack success against models [17, 24, 14, 21]. These classifiers are fast, because they do not run the model. But their label is model-independent by construction. The same label applies whichever model answers. So a prompt-level classifier must be defensive: it has to reject every prompt that could produce harm from any model. We use these datasets as the source of our prompts and prompt labels, and we ask how well the labels predict the safety of the interaction.

After generation: input-output moderation. A second line of work judges the response. Llama Guard and ShieldGemma classify prompts and responses under a safety taxonomy [8, 22]. WildGuard separates prompt harmfulness, response harmfulness and response refusal [6], and BeaverTails labels the harmfulness of question-answer pairs [9]. Qwen3Guard-Gen adds a “controversial” class between safe and unsafe [23]. Their response-level labels are the closest to ours. But they judge a finished response, so the decision comes after full generation. Li et al. note that this full detection causes high service latency [11].

Table 4: Paradigms of safety work, and where our study sits.

Paradigm	Decision point	Decision based on	Examples or focus
Prompt-level guards	Before generation	The prompt	ProtectAI detector; xTRam1, deepset, AdvBench
Input-output guards	After the full response	Prompt and response	Llama Guard, ShieldGemma, WildGuard
Streaming monitors	During generation	Prompt and partial response, trained for it	Streaming content monitor, Qwen3Guard-Stream, CARE
Model refusal benchmarks	No decision; models are ranked	Responses of many models to unsafe or safe-looking prompts	SORRY-Bench, HarmBench, XSTest, OR-Bench
Query harmfulness prediction	Before retrieval	The query	QHP
Ours	During generation	Prompt and partial response, with guards trained on full responses	Gap between prompt label and interaction outcome, across seven models

During generation: streaming and early stopping. The closest work monitors the response while it is generated. Li et al. argue that applying moderators trained on full responses to partial ones creates a training-inference gap. They build FineHarm, a dataset of 29K pairs with token-level labels, and a streaming content monitor. It reaches a macro-F1 of 0.95 or more after seeing about 18% of the response tokens on average [11]. Qwen3Guard-Stream adds a token-level head for real-time monitoring [23]. It has been integrated into CARE, a framework that detects unsafe text, rolls back, and intervenes at decoding time [7]. These works show that training for early decisions works. We look at the other regime. We use guards that were not built for partial responses, and ask how much signal they already hold at each prefix.

Refusal evaluation of models. Another line of work evaluates the models themselves. SORRY-Bench compares over 50 models on the same 440 unsafe instructions. It shows that refusal behavior differs across models, and it checks LLM judges against more than 7K human annotations [19]. HarmBench compares many models under the same attacks [14]. These results support two choices in our study. First, models differ on the same prompt. Second, an LLM judge can be validated against human labels. But the audience is the model developer, who wants to rank models. All prompts are unsafe by design, so there is no contrast between a prompt label and the outcome. And the results are not used to decide whether to reject a prompt or stop a response.

The cost of rejecting too much. Rejecting too much has a known cost. XSTest and OR-Bench measure how often models refuse safe prompts that only look unsafe [16, 3].

Recent work traces over-refusal to a small set of attention heads and suppresses them at inference time [18]. Guardrails add a similar cost outside the model. Stronger guardrails improve security but reduce utility, and LLM-based moderation adds latency [10]. These studies work at one end: prompts that are benign but look unsafe. We work at the other end. We ask how many prompts that datasets label as malicious still receive a safe response.

Safety in information retrieval. Safety is also an IR problem. Query harmfulness prediction asks, before retrieval, whether a query is likely to return harmful results [1]. Studies of AI-powered search engines find that they can quote or cite malicious content [13]. Like prompt-level classifiers, query harmfulness prediction decides from the query alone. Our setting moves the decision to the generated answer, which depends on both the query and the model.

6 Conclusion and Future Work

Conclusion. We asked whether the harmfulness of a prompt decides the safety of the interaction with a model. It does not. In our corpus of 41,944 prompt-response pairs from seven models, at least 93% of the responses to malicious-labeled prompts are safe. Only 8.5% of the prompts make any model respond harmfully, and only 3.5% of those make all seven models fail. So a prompt-level label cannot say which model will fail, and rejecting every prompt that looks harmful blocks mostly safe interactions.

The response carries the missing information. Guards fine-tuned on interaction labels reach an AUPRC of up to 0.885 with the response, and 0.05–0.51 with the prompt alone. Off the shelf, guards trained for prompts or for other label definitions reach a precision of at most 0.50 on our labels. Most of the signal appears early. After about one third of the response, guards trained only on full responses reach about 85–95% of their full-response AUPRC. This points to a decision made during generation: not before it, as prompt gating does, and not after it, as input-output guards do. Our claims concern measurement and data. We do not propose a new monitor, and we do not train for partial responses. The corpus, with the prompt label, the generating model and the response label kept together, lets others test such monitors against the gap we measure.

Our work can be improved in several ways by addressing the following limitations.

- *Labels.* One LLM judge labels the corpus, including the test set. The human audit covers 1,050 pairs, and the two annotators differ in strictness. We define harm as compliance with a malicious intent plus harmful text. Other definitions would give other labels.
- *Evaluation.* We evaluate classifiers at fixed prefixes. We do not run the full stop-or-continue monitor end to end. So we do not report how many safe interactions it would block, how many harmful ones it would let through, or how much harmful text it would produce before stopping.
- *Prompts.* The prompts come from single-turn queries only. Multi-turn conversations are not part of our dataset.
- *Models.* We use seven open-weight models from three families, between 2B and 27B parameters. We do not test closed or frontier models.

Ethical considerations. Our corpus contains harmful model outputs. The prompts come from public datasets and contain no personal data. The harmful text comes from small open-weight models, so the corpus adds little that these models do not already produce. We release our corpus to support safer LLM-based information systems.

References

1. Carrera, X., Fernández-Pichel, M., Losada, D.E.: Query harmfulness prediction (QHP): A new challenge for safer retrieval systems. In: Advances in Information Retrieval – 48th European Conference on Information Retrieval (ECIR). Lecture Notes in Computer Science, vol. 16484, pp. 55–69. Springer (2026). https://doi.org/10.1007/978-3-032-21300-6_4
2. Chiang, W.L., Zheng, L., Sheng, Y., Angelopoulos, A.N., Li, T., Li, D., Zhu, B., Zhang, H., Jordan, M.I., Gonzalez, J.E., Stoica, I.: Chatbot arena: An open platform for evaluating llms by human preference. In: Proceedings of the International Conference on Machine Learning (ICML). vol. 235, pp. 8359–8388 (2024)
3. Cui, J., Chiang, W.L., Stoica, I., Hsieh, C.J.: Or-bench: An over-refusal benchmark for large language models. In: Proceedings of the International Conference on Machine Learning (ICML). vol. 267, pp. 11515–11542 (2025)
4. deepset: Prompt injections dataset. Hugging Face dataset, <https://huggingface.co/datasets/deepset/prompt-injections>
5. Dettmers, T., Pagnoni, A., Holtzman, A., Zettlemoyer, L.: Qlora: Efficient finetuning of quantized llms. In: Advances in Neural Information Processing Systems (NeurIPS) (2023)
6. Han, S., Rao, K., Ettinger, A., Jiang, L., Lin, B.Y., Lambert, N., Choi, Y., Dziri, N.: Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. In: Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track. vol. 37, pp. 8093–8131 (2024). <https://doi.org/10.52202/079017-0261>
7. Hu, X., Huang, F., Yuan, C., Lin, J., Ho, T.Y.: CARE: Decoding time safety alignment via rollback and introspection intervention. arXiv preprint arXiv:2509.06982 (2025), <https://arxiv.org/abs/2509.06982>
8. Inan, H., Upasani, K., Chi, J., Rungta, R., Iyer, K., Mao, Y., Tontchev, M., Hu, Q., Fuller, B., Testuggine, D., Khabsa, M.: Llama guard: LLM-based input-output safeguard for human-AI conversations. arXiv preprint arXiv:2312.06674 (2023), <https://arxiv.org/abs/2312.06674>
9. Ji, J., Liu, M., Dai, J., Pan, X., Zhang, C., Bian, C., Chen, B., Sun, R., Wang, Y., Yang, Y.: Beavertails: Towards improved safety alignment of llm via a human-preference dataset. In: Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track. vol. 36 (2023)
10. Kumar, D., Birur, N.A., Baswa, T., Agarwal, S., Harshangi, P.: No free lunch with guardrails. arXiv preprint arXiv:2504.00441 (2025), <https://arxiv.org/abs/2504.00441>
11. Li, Y., Sheng, Q., Yang, Y., Zhang, X., Cao, J.: From judgment to interference: Early stopping llm harmful outputs via streaming content monitoring. In: Advances in Neural Information Processing Systems (NeurIPS) (2025), arXiv:2506.09996
12. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 2980–2988 (2017)
13. Luo, Z., Peng, Z., Liu, Y., Sun, Z., Li, M., Zheng, J., He, X.: Unsafe LLM-Based search: Quantitative analysis and mitigation of safety risks in AI web search. In: 34th USENIX Security Symposium (USENIX Security 25). USENIX Association (2025), <https://www.usenix.org/conference/usenixsecurity25/presentation/luo-zeren>

14. Mazeika, M., Phan, L., Yin, X., Zou, A., Wang, Z., Mu, N., Sakhaee, E., Li, N., Basart, S., Li, B., Forsyth, D., Hendrycks, D.: Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. In: Proceedings of the International Conference on Machine Learning (ICML). vol. 235, pp. 35181–35224 (2024)
15. ProtectAI: DeBERTa-v3 base prompt injection detector. Hugging Face model card (2023), <https://huggingface.co/protectai/deberta-v3-base-prompt-injection>
16. Röttger, P., Kirk, H., Vidgen, B., Attanasio, G., Bianchi, F., Hovy, D.: Xstest: A test suite for identifying exaggerated safety behaviours in large language models. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL) (2024), arXiv:2308.01263
17. Walled AI: AdvBench dataset. Hugging Face dataset, <https://huggingface.co/datasets/walledai/AdvBench>
18. Wang, Z., Zhang, B., Zhao, H., Guo, D.: Mitigating LLM over-refusal via dynamic semantic routing calibration. arXiv preprint arXiv:2609.25049 (2026), <https://arxiv.org/abs/2609.25049>
19. Xie, T., et al.: Sorry-bench: Systematically evaluating large language model safety refusal. In: International Conference on Learning Representations (ICLR) (2025), arXiv:2406.14598
20. xTRam1: Safeguard prompt injection dataset. Hugging Face dataset (2024), <https://huggingface.co/datasets/xTRam1/safe-guard-prompt-injection>
21. Yi, J., Xie, Y., Zhu, B., Kiciman, E., Sun, G., Xie, X., Wu, F.: Benchmarking and defending against indirect prompt injection attacks on large language models. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (2025), arXiv:2312.14197
22. Zeng, W., Liu, Y., Mullins, R., Peran, L., Fernandez, J., Harkous, H., Narasimhan, K., Proud, D., Kumar, P., Radharapu, B., Sturman, O., Wahltinez, O.: ShieldGemma: Generative AI content moderation based on Gemma. arXiv preprint arXiv:2407.21772 (2024), <https://arxiv.org/abs/2407.21772>
23. Zhao, H., et al.: Qwen3guard technical report. arXiv preprint arXiv:2510.14276 (2025), <https://arxiv.org/abs/2510.14276>
24. Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, J.Z., Fredrikson, M.: Universal and transferable adversarial attacks on aligned language models. arXiv preprint arXiv:2307.15043 (2023), <https://arxiv.org/abs/2307.15043>