IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

Machine Learning-Based Detection of Jailbreak and Prompt Injection Attacks in Large Language Models

Soham Sonawane[1], Anandita Kashyap[2], Jatin Kushwaha[3], Diya Sadhu[4], Vijaya Patil[5]
Students, School of Computing[1-4]
Professor, School of Computing[5]
MIT ADT University, Loni Kalbhor, Pune, India
[1] sohamsonawane2303@gmail.com, [2] ananditakashyap13@gmail.com [3] jatin.kushwaha@gmail.com,
[4] diyasadhu237@gmail.com [5] vijaya.patil@mituniversity.edu.in

Abstract: *Large Language Models (LLMs) are now built into many everyday applications, which exposes them to jailbreak and prompt injection attacks that try to override safety rules or hijack the instructions a model follows. This paper presents a machine learning framework that screens prompts and flags likely malicious ones before they reach the model. We compare Logistic Regression, a Linear Support Vector Machine (SVM) and a fine-tuned DistilBERT classifier on a combined dataset of 24,530 prompts drawn from GuardrailsAI and Neuralchemy, prepared through label normalization, duplicate checking and stratified splitting. After the decision threshold was tuned on validation data, DistilBERT reached 95.56% accuracy, 93.74% precision, 88.77% recall and a 91.18% F1-score on the held-out test set. On a separate collection of 300 JailbreakBench-associated prompts, however, accuracy fell to about 82% and 55 of 200 attack prompts went undetected, which shows how much harder diverse adversarial inputs are than in-distribution data. The results suggest that transformer-based screening can serve as a useful additional security layer for LLM applications, provided it is validated on external benchmarks and combined with other defences.*

Keywords: Large Language Models, Jailbreak Detection, Prompt Injection, Machine Learning, DistilBERT, Adversarial Robustness, Natural Language Processing.

I. INTRODUCTION

Large Language Models (LLMs) now sit behind chat assistants, coding tools, search products and a growing number of automated decision systems. As they are wired into software that handles private or business-critical data, the question of how they can be misused has moved from a research curiosity to a practical engineering concern.

Two families of attack dominate that discussion. A jailbreak tries to talk a model out of its safety training so that it produces content it would normally refuse. A prompt injection instead tampers with the instructions the model is following, either through text typed by the user or through material the model reads on its own, such as web pages, documents and retrieved passages. The two overlap in practice, but they differ in goal and in where the malicious text enters the system, so we treat them as related rather than identical threats.

Keyword lists and hand-written filters catch only part of this traffic. Attackers rewrite requests in other languages, wrap them in role-play, spread them across indirect instructions or hide them inside otherwise ordinary text. A defence therefore has to judge what a prompt is trying to do, not only which words it contains.

Three recent studies motivate our work. HarmBench [1] offers a standardized way to compare automated red-teaming attacks and refusal defences, and found that no single attack or defence performs best everywhere. Dzhaliuk et al. [2] compared classical classifiers, fine-tuned transformers, specialized detectors and general-purpose LLMs for prompt injection detection, and drew attention to the trade-off between accuracy and computational cost. Gao [3] showed that

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

even the automated judges used to score jailbreak attacks can be unreliable and can be manipulated. Together, these studies argue that a detector should be assessed on more than a conventional test split.

Against this background we build and evaluate a classifier that screens a prompt before it reaches an LLM. We train Logistic Regression, a Linear Support Vector Machine (SVM) and a fine-tuned DistilBERT model [6] on a combined dataset assembled from GuardrailsAI and Neuralchemy [5], tune the decision threshold on validation data, and then test the resulting detector on 300 prompts from a separate JailbreakBench-associated collection [4]. We pay particular attention to missed attacks (false negatives), because these are the prompts that slip through.

The main contributions of this work are as follows:

1. Unified dataset. A binary prompt dataset of 24,530 records built from two public sources, with label normalization and duplicate checks.
2. Like-for-like comparison. An evaluation of Logistic Regression, Linear SVM and fine-tuned DistilBERT under the same data splits and metrics.
3. Threshold study. A validation-based analysis of how the decision threshold shifts the balance between precision and recall.
4. External evaluation. A test on JailbreakBench-associated prompts that quantifies the drop in recall outside the training distribution.
5. Deployment view. A description of how the detector can act as an input-screening layer alongside other safeguards.

We do not present prompt classification as a complete defence. Our aim is to measure how far it goes, where it fails, and what that implies for its role in a layered security design. The remainder of the paper is organized as follows. Section II reviews related work, Section III outlines the methodology, Section IV describes the dataset, Section V presents the models, Section VI details the experimental setup and metrics, Section VII reports and discusses the results, and Section VIII concludes with directions for future work.

II. LITERATURE REVIEW

A. Standardized Evaluation of Jailbreak Attacks

Comparing jailbreak attacks and defences has long been difficult because every paper used its own prompts, target models and success criteria. HarmBench [1] was designed to remove that obstacle. It evaluates 18 automated red-teaming methods against 33 target models and defences under a single protocol, so that results can be compared across studies. One of its central findings is that no attack and no defence came out on top in every setting. A safety mechanism tested against only a handful of attacks can therefore give a misleadingly comfortable picture.

We take the lesson from this work rather than its purpose. HarmBench measures how robust a model is, whereas our classifier decides whether an incoming prompt is malicious. Those are different questions, so we use prompts from an outside source to probe our detector but interpret the outcome strictly as detection performance.

B. Machine Learning for Prompt Injection Detection

Dzhaliuk et al. [2] compared four groups of defences on a benchmark of more than 300,000 labelled prompts: classical classifiers (including Naive Bayes, Random Forest, LightGBM and a multilayer perceptron), fine-tuned encoder transformers (DistilBERT, RoBERTa and DeBERTa), specialized pre-trained injection detectors, and general-purpose LLMs used as classifiers. Fine-tuned transformers reached the highest detection rates on balanced data, classical classifiers stayed competitive at a much lower computational cost, specialized detectors showed lower recall, and the quality of LLM-based classifiers depended strongly on the model chosen.

Their results show that selecting a detector means weighing accuracy against cost and deployment constraints. We follow the same spirit by comparing classical and transformer models, but our data, model set and training conditions differ, so the figures in the two studies should not be compared directly.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

C. Reliability and Adversarial Robustness

Gao [3] asked whether the automated judges that score jailbreak attacks can themselves be trusted. Using 596 human-labelled completions from the HarmBench classifier validation set, the study compared a dedicated safety classifier with three LLM judges. The classifier flagged too much, with very high recall but lower precision, whereas the LLM judges were precise but erratic in recall. Wrapping harmful text in benign framing, without removing the harm, flipped the verdicts of the LLM judges, and a white-box greedy coordinate gradient (GCG) attack [7] reversed 21 of 30 confident detections of the dedicated classifier.

Gao studies judges rather than input filters, but the lesson carries over: a high score on ordinary test data says little about behaviour under deliberate manipulation. This is the reason we add an external evaluation to the standard held-out test.

D. Research Gap

Taken together, these studies point to a gap. Prompt classifiers are usually reported on in-distribution test splits, while evidence on how they behave on prompts from a different source is thinner. We address a deliberately narrow version of this gap by asking how well classical and transformer classifiers separate benign from malicious prompts, how much the decision threshold matters, and how large the drop is when the detector meets a separate benchmark. We make no claim of introducing a new attack or of solving prompt injection.

III. PROPOSED METHODOLOGY

A. Overview

Fig. 1 summarizes the pipeline. Prompts from two public datasets are cleaned, mapped to a common binary label and merged into a single dataset of 24,530 records. The data are split 80/10/10 into training, validation and test sets. Logistic Regression, Linear SVM and DistilBERT are trained on the training split, the validation split is used to compare them and to choose a decision threshold for each, and the held-out test split provides the primary results. A separate 300-prompt collection is reserved for external evaluation. Finally, the trained model is placed in front of an LLM application as a screening layer.

B. Data and Preprocessing

Section IV gives the dataset details. In brief, GuardrailsAI (18,256 records) and Neuralchemy (6,274 records) were converted to a prompt-plus-label format, with 0 for benign and 1 for malicious, screened for incomplete and duplicate entries, and divided with stratification so that each split keeps the original class ratio. Since only about 23% of the records are malicious, we report precision, recall and F1-score alongside accuracy.

C. Models and Threshold Selection

Logistic Regression and Linear SVM serve as classical baselines, and DistilBERT, fine-tuned for three epochs, is the transformer model (Section V). A classifier produces a score that must be compared with a threshold to yield a label, and the common default of 0.5 is not necessarily the best choice for an imbalanced problem. We therefore evaluated a range of thresholds on the validation split and kept, for each model, the one with the highest F1-score: 0.80 for DistilBERT and 0.15 for Linear SVM. Choosing the threshold on validation data keeps the test split untouched until the final evaluation.

D. External Benchmark Evaluation

To check generalization, we scored 300 prompts from a separate JailbreakBench-associated collection [4]. It contains 200 attack prompts, generated with PAIR [8], GCG [7] and manual adversarial writing, and 100 benign prompts taken from XSTest [9]. This collection took no part in training, validation or threshold selection.

E. Detection Workflow

In deployment, the classifier sits between the user and the LLM. Each incoming prompt is scored and compared with the chosen threshold. A prompt judged benign is passed on, still subject to the application's other safeguards; a prompt judged malicious is flagged or blocked according to the application's security policy. The detector is meant to add a layer of protection. It cannot promise to catch every attack and should not replace access control, input validation, output monitoring or sound system design.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

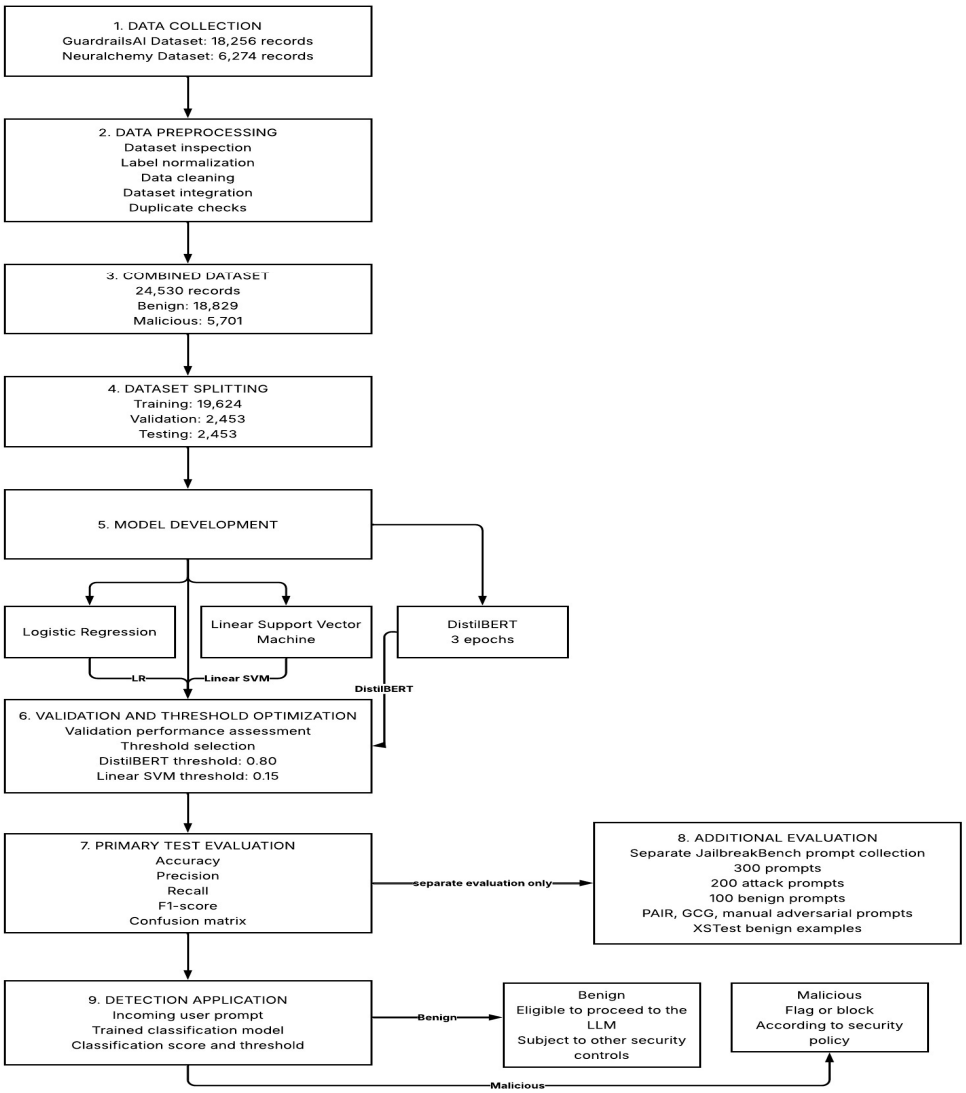


Fig. 1. Overall architecture of the proposed prompt-screening framework, from data collection to deployment.

IV. DATASET PREPARATION

A detector is only as good as the examples it learns from, and jailbreak and prompt injection attempts take many forms. To widen the range of examples, we combined two public prompt datasets into a single resource.

A. Dataset Sources

The GuardrailsAI dataset provides prompts labelled by their security characteristics, such as jailbreak attempts and benign inputs, and was our main source with 18,256 records. The Neuralchemy dataset added 6,274 records with different prompt styles and attack-related label categories. Table I lists the contribution of each source.

TABLE I: Dataset Sources

Dataset Source	Number of Records
GuardrailsAI	18,256
Neuralchemy	6,274
Total	24,530

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

B. Label Normalization

The two sources do not label their data in the same way, so their labels were mapped to a single binary scheme:

Label 0 (benign): prompts not identified as malicious or as attempts to bypass an LLM's intended instructions.

Label 1 (malicious): prompts identified as jailbreak attempts, prompt injection attempts or other malicious patterns present in the source data.

The merged dataset holds 18,829 benign and 5,701 malicious prompts (Table II). Benign prompts therefore make up about three quarters of the data, and this imbalance is the reason we rely on precision, recall and F1-score rather than accuracy alone.

TABLE II: Binary Class Distribution

Class	Label	Number of Records	Percentage
Benign	0	18,829	76.76%
Malicious	1	5,701	23.24%
Total	—	24,530	100%

C. Integration and Preprocessing

Preparing the data involved five steps:

1. Structure inspection: identifying the prompt and label fields in each source.
2. Label normalization: mapping source labels to the benign and malicious classes.
3. Cleaning: removing incomplete or unusable entries.
4. Integration: merging the prepared records into one dataset.
5. Duplicate checking: looking for repeated prompts that could distort evaluation.

D. Training, Validation and Test Splits

The dataset was divided 80/10/10 using stratified sampling, so that every split keeps the original class proportions (Table III). The training split is used to fit the models, the validation split to compare them and select thresholds, and the test split only for the final evaluation, which lets us estimate performance on prompts the models have never seen.

TABLE III: Dataset Split Distribution

Dataset Split	Total Records	Benign (0)	Malicious (1)
Training	19,624	15,063	4,561
Validation	2,453	1,883	570
Testing	2,453	1,883	570
Total	24,530	18,829	5,701

E. Duplicate and Leakage Checks

After splitting, we checked whether identical prompts appeared in more than one subset and found no exact overlap between the training and validation sets, the training and test sets, or the validation and test sets. This lowers the risk of inflated scores from repeated prompts. It does not rule out near-duplicates or semantically similar prompts, which would require a separate similarity analysis.

F. External JailbreakBench-Associated Collection

Apart from the 24,530-record dataset, we assembled a separate collection of 300 prompts for external evaluation (Table IV). Its 200 attack prompts were produced with PAIR, GCG and manual adversarial writing, and its 100 benign prompts come from XSTest. Because this collection comes from a different source and style than the training data, it is a more demanding test of how the detector copes with unfamiliar prompt patterns.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

TABLE IV: External Evaluation Collection

Category	Number of Prompts
Attack prompts	200
Benign prompts	100
Total	300

V. MODEL DEVELOPMENT

We selected three models that represent two broad approaches to text classification: classical machine learning (Logistic Regression and Linear SVM) and transformer-based deep learning (DistilBERT). Comparing them shows what a more complex language model buys in detection quality and at what computational price.

A. Logistic Regression

Logistic Regression is a standard supervised method for binary classification. It estimates the probability that a prompt belongs to the malicious class, and a threshold on that probability decides the final label. Its simplicity makes it a natural baseline against which the other two models can be judged.

B. Linear Support Vector Machine

A Linear SVM looks for the boundary that separates benign and malicious prompts with the widest possible margin, and classifies new prompts by which side of the boundary they fall on. Linear SVMs often work well on high-dimensional text features, which is why we included one next to Logistic Regression. We also varied its decision threshold to see whether moving the boundary improves the balance between precision and recall.

C. DistilBERT

DistilBERT [6] is a compact transformer obtained by distilling BERT: it has a smaller architecture but keeps much of the original model's language understanding. Classical models typically rely on separately engineered text features, whereas a transformer builds contextual representations in which the meaning of a word depends on its surrounding words. This can help with prompts whose intent is spread across several sentences or hidden in role-play.

We fine-tuned DistilBERT for binary classification on the training split for three epochs. It was chosen to test whether such contextual representations improve the detection of complex jailbreak and prompt injection patterns compared with the classical alternatives.

D. Rationale for the Model Set

The three models span different levels of complexity. If the transformer shows a measurable advantage over the two classical baselines on the same data, that supports its use in a screening layer; if not, the cheaper models may be the more practical choice. All models were assessed with accuracy, precision, recall and F1-score, and with threshold optimization.

VI. EXPERIMENTAL SETUP AND EVALUATION METRICS

A. Environment

All experiments ran in Python on a Windows system. DistilBERT training and evaluation used CUDA on an NVIDIA GeForce RTX 4050 Laptop GPU (Table V).

TABLE V: Experimental Environment

Component	Configuration
Operating System	Windows
Programming Language	Python
GPU	NVIDIA GeForce RTX 4050 Laptop GPU
Deep Learning Model	DistilBERT
Traditional ML Models	Logistic Regression, Linear SVM

Copyright to IJARSCT
www.ijarsct.co.in
DOI: 10.48175/IJARSCT-40870

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

Component	Configuration
GPU Acceleration	CUDA

B. Training Procedure

The 19,624-record training split was used to fit all three models, with DistilBERT fine-tuned for three epochs. The 2,453-record validation split was used to compare models and select thresholds, and the 2,453-record test split was held back for final evaluation

C. Evaluation Metrics

We treat the malicious class as positive. A true positive (TP) is a malicious prompt correctly flagged, a true negative (TN) a benign prompt correctly passed, a false positive (FP) a benign prompt wrongly flagged and a false negative (FN) a malicious prompt wrongly passed. Accuracy is the share of all prompts classified correctly:

$$\text{Accuracy} = (TP+TN) / (TP+TN+FP+FN) \quad (1)$$

Precision is the share of flagged prompts that are truly malicious:

$$\text{Precision} = TP / (TP+FP) \quad (2)$$

Recall is the share of malicious prompts that the detector catches:

$$\text{Recall} = TP / (TP+FN) \quad (3)$$

The F1-score is the harmonic mean of precision and recall:

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall}) \quad (4)$$

We also inspect confusion matrices. False negatives deserve special attention in this study, since each one is a malicious prompt that would reach the LLM unchallenged.

D. Threshold Optimization

The threshold decides how high a model's malicious-class score must be before a prompt is flagged. We evaluated a range of values on the validation split and used the F1-score to pick the operating point. The best thresholds were 0.15 for Linear SVM (validation F1-score 0.8694) and 0.80 for DistilBERT (validation F1-score 0.8853), and these were then applied unchanged to the test split.

E. External Evaluation Procedure

The 300-prompt external collection was scored with the same threshold-optimized detector and analyzed with the same metrics and confusion matrix. It is reported separately from the main test results because its purpose is different: to examine behaviour beyond the distribution of the primary dataset.

VII. RESULTS AND DISCUSSION

A. Performance on the Primary Test Set

Table VI reports the threshold-optimized models on the 2,453-prompt test split.

TABLE VI: Test Performance After Threshold Optimization

Evaluation Metric	Logistic Regression	Linear SVM	DistilBERT
Accuracy	93.60%	94.86%	95.56%
Precision	84.82%	90.81%	93.74%
Recall	88.25%	86.67%	88.77%
F1-Score	86.50%	88.69%	91.18%

DistilBERT scored highest on every metric, with 95.56% accuracy and a 91.18% F1-score. Its precision of 93.74% means that most prompts it flagged were truly malicious, and its recall of 88.77% means that it caught most, though not all, of the malicious prompts. Linear SVM was close behind at 94.86% accuracy and an F1-score of 88.69%, so a much simpler model is already a serious contender.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

The gap between the two models, about 2.5 F1 points, suggests that contextual language representations help on this task. It would be premature to read it as proof that transformers will outperform classical models on every type of jailbreak or injection attack, particularly given the external results below.

B. Effect of the Decision Threshold

Tuning the threshold on validation data transferred well to the test split: Linear SVM moved from a validation F1-score of 0.8694 to a test F1-score of 88.69%, and DistilBERT from 0.8853 to 91.18%. The decision threshold therefore acts as a tunable part of the detector, not a fixed detail, and the default of 0.5 should not be assumed to be appropriate. A security team that values catching attacks over avoiding false alarms could deliberately choose a lower threshold.

C. Evaluation on the External Collection

Table VII gives the confusion matrix for the 300-prompt external collection, and Table VIII the derived metrics.

TABLE VII: Confusion Matrix for the External Evaluation

Actual Class	Predicted Benign	Predicted Malicious
Benign	100	0
Malicious	55	145

TABLE VIII: External Evaluation Metrics

Metric	Result
Accuracy	81.67%
Precision	100.00%
Recall	72.50%
F1-Score	84.06%
False Positive Rate	0.00%
False Negative Rate	27.50%

The detector classified all 100 benign prompts correctly but identified only 145 of the 200 attack prompts, giving an accuracy of 81.67% (about 82%) and a recall of 72.50%. More than one in four attack prompts therefore passed through the screen. The perfect precision and zero false positives should be read with care: the benign set contained only 100 prompts, and it is not clear how a different or larger benign set would behave.

The contrast with the primary test is striking. Recall fell from 88.77% to 72.50%, even though the external collection is small. High accuracy on the test split is therefore not evidence of reliable protection against every attack style.

D. Attack Categories

The external collection mixes three attack styles with one benign source. PAIR [8] is an automated red-teaming method that iteratively rewrites a prompt until the target model complies. GCG [7] searches for adversarial suffixes that push a model toward a prohibited response. The manual adversarial prompts were written by hand to test less standardized patterns. XSTest [9] supplies harmless prompts that superficially resemble unsafe ones, and is meant to expose detectors that over-block.

We only have aggregate results for the 300 prompts. A per-category breakdown would show which styles are hardest to detect and is an obvious next step.

E. Overall Findings and Limitations

Three observations stand out. First, both Linear SVM and DistilBERT performed strongly on the held-out test split, with DistilBERT ahead. Second, selecting the threshold on validation data gave a principled operating point and the choice changed the results. Third, recall dropped sharply on the external collection, which indicates that a detector trained on one data distribution can miss attacks written in other styles.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

Several limitations apply. The dataset is imbalanced, and although no exact duplicates were found across splits, near-duplicates may still exist. The external collection is small and its results are aggregate only. We report single runs without confidence intervals. And we have not tested obfuscated, encoded, multilingual or multi-turn attacks. These limitations do not undermine the comparison, but they do limit how far the results can be generalized.

VIII. CONCLUSION AND FUTURE SCOPE

A. Conclusion

We presented a machine learning framework that screens prompts for jailbreak and prompt injection attempts before they reach an LLM. A combined dataset of 24,530 prompts from GuardrailsAI and Neuralchemy was normalized to a binary labelling and split for training, validation and testing, and Logistic Regression, Linear SVM and fine-tuned DistilBERT were compared on it.

With a threshold chosen on validation data, DistilBERT reached 95.56% accuracy, 93.74% precision, 88.77% recall and a 91.18% F1-score on the held-out test split, ahead of Linear SVM at 94.86% accuracy and an F1-score of 88.69%. On a separate collection of 300 prompts, however, accuracy was 81.67% and 55 of 200 attack prompts were missed, even though no benign prompt was wrongly flagged. The result shows that in-distribution scores can overstate real-world protection.

Machine learning detectors are therefore best seen as one layer of a broader LLM security architecture, combined with access control, input validation and output monitoring, and not as a complete answer to jailbreak and prompt injection.

B. Future Scope

Several directions could strengthen this work:

1. Broader data. Adding datasets with more jailbreak strategies, injection patterns, languages and multi-turn attacks would reduce dependence on the current training distribution.
2. Adaptive attacks. Testing against encoded, obfuscated and adversarially optimized prompts would expose weaknesses that ordinary test splits do not reveal.
3. Explainability. Techniques such as SHAP and LIME could show which parts of a prompt drive a malicious prediction, although their reliability would first need to be checked.
4. Hybrid detection. Combining the classifier with rule-based checks and contextual analysis would give several independent layers instead of a single point of failure.
5. Real-time integration. Measuring latency, throughput and resource needs would show whether the detector is practical in production applications.
6. Continuous updating. Because attack techniques keep changing, a deployed system needs regular evaluation, retraining on fresh data and monitoring of false positives and false negatives.
7. Wider benchmarks. Evaluating on more benchmarks, other LLM architectures, retrieval-augmented generation and tool-using agents would test the detector in more realistic settings.

REFERENCES

1. M. Mazeika, L. Phan, X. Yin, A. Zou, Z. Wang, N. Mu, E. Sakhaee, N. Li, S. Basart, B. Li, D. Forsyth, and D. Hendrycks, "HarmBench: A standardized evaluation framework for automated red teaming and robust refusal," in Proc. 41st Int. Conf. Machine Learning (ICML), PMLR vol. 235, 2024.
2. N. Dzhaliuk, D. Sabodashko, V. Khoma, Y. Khoma, V. Kolchenko, and M. Podpora, "Comparative Evaluation of Machine Learning Methods for Protecting LLMs from Prompt Injection Attacks," International Journal of Information Security, vol. 25, article 109, 2026. doi: 10.1007/s10207-026-01264-8.
3. Y. Gao, "How reliable is your jailbreak judge? Calibration and adversarial robustness of automated ASR scoring," arXiv:2606.25487, Jun. 2026.
4. P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, V. Sehwag, E. Dobriban, N. Flammarion, G. J. Pappas, F. Tramèr, H. Hassani, and E. Wong, "JailbreakBench: An open robustness benchmark for

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 3, November 2026
Impact Factor: 8.2

jailbreaking large language models," in Proc. NeurIPS Datasets and Benchmarks Track, 2024.

5. neuralchemy, "Prompt-injection-dataset," Hugging Face Datasets, [Online]. Available: https://huggingface.co/datasets/neuralchemy/Prompt-injection-dataset. Accessed: Nov. 18, 2026.
6. V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," arXiv:1910.01108, 2019.
7. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," arXiv:2307.15043, 2023.
8. P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong, "Jailbreaking black box large language models in twenty queries," arXiv:2310.08419, 2023.
9. P. Röttger, H. R. 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. NAACL, 2024.
10. P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, V. Sehwag, E. Dobriban, N. Flammarion, G. J. Pappas, F. Tramèr, H. Hassani, and E. Wong, "JailbreakBench: An Open Robustness Benchmark for Jailbreaking Large Language Models," in Proceedings of the NeurIPS 2024 Datasets and Benchmarks Track, 2024. DOI: 10.57967/hf/2540. [Online]. Available: https://huggingface.co/datasets/JailbreakBench/JBB-Behaviors
11. S. Han, K. Rao, A. Ettinger, L. Jiang, B. Y. Lin, N. Lambert, Y. Choi, and N. Dziri, "WildGuard: Open One-Stop Moderation Tools for Safety Risks, Jailbreaks, and Refusals of LLMs," arXiv preprint arXiv:2406.18495, 2024. [Online]. Available: https://arxiv.org/abs/2406.18495. (Dataset: WildGuardMix, Hugging Face). https://huggingface.co/datasets/allenai/wildguardmix