Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.063826

ARTICLE

Adversarial Prompt Detection in Large Language Models: A Classification-Driven Approach

Ahmet Emre Ergün and Aytuğ Onan*

Department of Computer Engineering, Faculty of Engineering and Architecture, İzmir Katip Çelebi University, İzmir, 35620, Turkey

*Corresponding Author: Aytuğ Onan. Email: aytugonan@gmail.com

Received: 25 January 2025; Accepted: 01 April 2025; Published: 19 May 2025

ABSTRACT: Large Language Models (LLMs) have significantly advanced human-computer interaction by improving natural language understanding and generation. However, their vulnerability to adversarial prompts–carefully designed inputs that manipulate model outputs–presents substantial challenges. This paper introduces a classification-based approach to detect adversarial prompts by utilizing both prompt features and prompt response features. Eleven machine learning models were evaluated based on key metrics such as accuracy, precision, recall, and F1-score. The results show that the Convolutional Neural Network–Long Short-Term Memory (CNN-LSTM) cascade model delivers the best performance, especially when using prompt features, achieving an accuracy of over 97% in all adversarial scenarios. Furthermore, the Support Vector Machine (SVM) model performed best with prompt response features, particularly excelling in prompt type classification tasks. Classification results revealed that certain types of adversarial attacks, such as "Word Level" and "Adversarial Prefix", were particularly difficult to detect, as indicated by their low recall and F1-scores. These findings suggest that more subtle manipulations can evade detection mechanisms. In contrast, attacks like "Sentence Level" and "Adversarial Insertion" were easier to identify, due to the model's effectiveness in recognizing inserted content. Natural Language Processing (NLP) techniques played a critical role by enabling the extraction of semantic and syntactic features from both prompts and their corresponding responses. These insights highlight the importance of combining traditional and deep learning approaches, along with advanced NLP techniques, to build more reliable adversarial prompt detection systems for LLMs.

KEYWORDS: LLM; classification; NLP; adversarial; prompt; machine learning; deep learning

1 Introduction

The rapid development of Large Language Models (LLMs), such as GPT and LLaMA, has revolutionized natural language processing (NLP), enabling a wide range of applications across various fields [1]. However, these models are vulnerable to adversarial prompts–specially designed inputs that exploit their architecture to produce unintended, biased, or harmful outputs [2]. Detecting these adversarial prompts is essential for ensuring the ethical and secure use of LLMs, as well as preventing misuse in sensitive areas [3].

The increasing frequency of adversarial attacks on LLMs has sparked substantial research efforts [4]. These attacks can bypass content moderation systems, manipulate factual information, or generate harmful content, posing significant risks to industries like healthcare, finance, and social media [5]. Despite prior advancements, existing adversarial detection methods often focus exclusively on either deep learning models (e.g., transformers, LSTMs) or traditional classifiers (e.g., SVM, Random Forest), without exploring their complementary strengths. Our study bridges this gap by systematically comparing 11 machine learning

models, demonstrating that hybrid approaches (e.g., CNN-LSTM with structured classifiers) outperform single-method solutions. Additionally, existing defense mechanisms, such as adversarial training and rule-based detection, struggle with generalization and computational efficiency, whereas our classification-based approach provides a scalable and interpretable alternative. This study contributes to strengthening adversarial detection in LLMs by providing a comprehensive model comparison and proposing an efficient classification-driven framework. While existing methods, such as adversarial training and rule-based detection, have shown promise, they often lack the flexibility and scalability needed to address the evolving nature of adversarial threats [6]. This highlights the need for a robust, classification-based approach that can differentiate adversarial prompts from benign inputs.

Adversarial attacks on Large Language Models (LLMs) have gained significant attention due to their potential to manipulate model outputs through carefully crafted inputs. While previous studies primarily focus on detecting adversarial prompts, our study extends this scope by evaluating both **adversarial prompt identification** and **adversarial response identification**. Specifically, our approach investigates how adversarial prompts impact the responses generated by LLMs and how effectively machine learning models can classify these responses. By considering both prompt-only and prompt-response features, we provide a comprehensive analysis of adversarial behaviors in LLM interactions. This dual evaluation enables a more robust understanding of adversarial vulnerabilities, allowing for more effective mitigation strategies in both detecting adversarial intent at the input level and identifying adversarial characteristics in generated responses.

This study proposes a machine learning-driven method for adversarial prompt detection, utilizing NLP techniques to extract meaningful features from both prompt and response data. Unlike prior research that evaluates only a single classifier or a limited set of models, our study systematically compares 11 machine learning models, offering new insights into their effectiveness across different adversarial types. Our findings highlight that hybrid models, such as CNN-LSTM paired with structured classifiers like SVM or XGBoost, achieve superior detection accuracy while maintaining computational efficiency. Future work will explore expanding our dataset with real-world adversarial prompts and investigating hybrid approaches that integrate classification with retrieval-based fact verification when adversarial prompts contain misleading factual claims. By analyzing linguistic patterns, semantic structures, and contextual cues, NLP allows the identification of subtle adversarial manipulations that might be missed by traditional approaches. Adversarial prompt detection is fundamentally different from retrieval-based approaches such as Retrieval-Augmented Generation (RAG). While RAG models enhance factual consistency by incorporating external knowledge sources into generated responses, our study focuses on detecting adversarial prompts and classifying adversarial responses. The key distinction lies in the nature of the task: adversarial prompts do not necessarily introduce factual inaccuracies but often exploit linguistic and contextual weaknesses in LLMs. Therefore, retrieval augmentation does not directly address the problem of adversarial attack detection. Instead, our classification-based approach systematically identifies adversarial patterns in both prompts and responses, aligning with the core objectives of this study.

We evaluate eleven machine learning models across key performance metrics, including accuracy, precision, recall, and F1-score, to assess their effectiveness in detecting adversarial prompts. The primary goals of this paper are threefold:

Our contributions include a thorough comparison of classifiers, the application of NLP techniques for feature extraction, and the introduction of a detailed dataset specifically designed for adversarial prompt detection.

This paper is structured as follows: In Section 2, related works on adversarial prompt identification have been presented. In Section 3, data sets, methodologies, and the experimental configuration have been presented. Section 4 presents the experimental results and Section 5 presents the discussions. Finally, Section 6 presents the concluding remarks.

2 Related Works

Recent studies have highlighted the vulnerabilities of Large Language Models (LLMs) to adversarial prompts, and a variety of methods have been proposed to generate, detect, and defend against such attacks. Das et al. [7] demonstrated the effectiveness of human-readable adversarial prompts, crafted using movie scripts, to bypass LLM safety mechanisms by exploiting contextual relevance. Similarly, Paulus et al. [8] introduced AdvPrompter, a fast adaptive prompting technique that generates adversarial prompts up to three times faster than existing methods, reflecting the increasing sophistication of attack strategies. Zhu et al. [9] developed PromptBench, a comprehensive benchmark for evaluating LLM robustness across tasks like text classification, summarization, and question-answering. Their results showed that leading LLMs, such as ChatGPT and Bard, are highly vulnerable to adversarial attacks, revealing significant weaknesses in their robustness.

To address these vulnerabilities, Ayub et al. [10] proposed embedding-based machine learning classifiers to detect malicious prompts, leveraging advanced embeddings to identify adversarial patterns. Hu et al. [11] introduced a token-level detection method that uses perplexity measures and contextual information, demonstrating effectiveness across multiple LLM architectures. Kumar et al. [12] proposed the "erase-and-check" framework, which provides certifiable safety guarantees by removing potentially harmful input fragments and validating the modified input. This method reduced adversarial success rates, balancing theoretical guarantees with practical defense strategies.

These studies collectively emphasize the ongoing vulnerability of LLMs to adversarial prompts and highlight the need for robust defense mechanisms. Approaches such as adversarial benchmarks, embedding-based detection, token-level analysis, and safety certification frameworks represent important advancements in addressing these challenges, paving the way for more secure and resilient LLMs.

The various approaches to adversarial prompt detection, ranging from rule-based systems and adversarial training to deep learning and hybrid models, have made significant progress. However, challenges like scalability remain. Our study builds upon these efforts by providing a comprehensive comparison of multiple machine learning classifiers using NLP techniques. We offer detailed insights into their performance metrics and introduce a robust dataset specifically designed for adversarial prompt detection. Unlike most related works, we focus on the strengths of traditional and ensemble classifiers in handling a wide range of adversarial types. Table 1 provides a comparative analysis of these studies alongside our work, emphasizing the methodologies, key contributions, and limitations of each approach.

Table 1: Comparison of related studies and our study

Study	Focus	Methodology	Key contributions	Limitations
Das et al. [7]	Exploiting contextual relevance in adversarial prompts.	Human-readable adversarial prompts crafted from movie scripts.	Demonstrated effectiveness in bypassing LLM safety mechanisms.	Limited to specific types of adversarial prompts (contextually relevant).
Paulus et al. [8]	Fast generation of adversarial prompts.	AdvPrompter: Adaptive prompting technique for faster adversarial prompt generation.	Generated adversarial prompts 3× faster than existing methods.	Focused on prompt generation, not detection or defense.
Zhu et al. [9]	Benchmarking LLM robustness against adversarial prompts.	PromptBench: Comprehensive benchmark for evaluating LLM robustness.	Revealed vulnerabilities in leading LLMs like ChatGPT and Bard.	Limited to evaluation; no defense mechanisms proposed.
Ayub et al. [10]	Detecting malicious prompts using embeddings.	Embedding-based machine learning classifiers.	Improved detection of adversarial patterns using advanced embeddings.	Reliance on embeddings may limit generalizability to unseen adversarial types.
Hu et al. [11]	Token-level detection of adversarial prompts.	Perplexity measures and contextual information for token-level detection.	Demonstrated effectiveness across multiple LLM architectures.	May struggle with sophisticated adversarial prompts that mimic natural language.
Kumar et al. [12]	Certifiable safety guarantees for LLMs.	"Erase-and-check" framework: Erasing and validating input fragments.	Reduced adversarial success rates with theoretical safety guarantees.	Computationally intensive; may not scale well for real-time applications.
Our study	Classification-based detection of adversarial prompts.	Evaluation of 11 machine learning models using Prompt and Prompt Response features.	CNN-LSTM achieved >97% accuracy with prompt features; SVM excelled with prompt response features.	Focused on classification; does not address real-time detection or generation of adversarial prompts.

These studies collectively highlight the diverse approaches to adversarial prompt detection, ranging from rule-based systems and adversarial training to deep learning and hybrid models. While significant progress has been made, our study directly addresses computational efficiency by implementing a classification-driven detection framework that eliminates retrieval overhead, making it suitable for real-time applications. Adaptability is demonstrated through the evaluation of 11 machine learning models, which perform differently based on adversarial prompt types, allowing task-specific optimization. Scalability, while still an open challenge, is partially addressed through our dataset of 39,648 adversarial and non-adversarial prompts, ensuring large-scale evaluation. These findings reinforce the effectiveness of classification-based adversarial prompt detection and provide a strong foundation for future scalability improvements. Unlike retrieval-augmented approaches, which enhance factual correctness in LLM-generated text, adversarial prompt detection requires a classification-driven methodology that identifies manipulative intent within inputs. RAG-based models are designed for fact-checking and knowledge augmentation, but adversarial prompts do not necessarily introduce factual inaccuracies–they instead exploit linguistic and contextual weaknesses in LLMs. Retrieval-based models do not directly contribute to adversarial intent classification and introduce unnecessary computational overhead, making them impractical for real-time adversarial detection. Our CNN-LSTM model achieves >97% accuracy without external retrieval, proving the effectiveness of classification-driven detection. While not required for this study, future research may explore hybrid models that integrate classification with retrieval-based fact verification for adversarial prompts that manipulate factual claims rather than language structures. Existing literature on adversarial attacks primarily employs classification-based techniques rather than retrieval-based models. Our study differentiates itself by leveraging a comprehensive classification framework to detect adversarial intent without the need for external knowledge retrieval. The discussion in this section has been expanded to clarify this distinction.

Our study improves upon these by providing a comprehensive comparison of multiple machine learning classifiers by using NLP, offering detailed insights into their performance metrics, and introducing a robust dataset specifically tailored for adversarial prompt detection. Unlike most related works, we emphasize the relative strengths of traditional and ensemble classifiers in handling diverse adversarial types. Table 1 provides a comparative analysis of these studies alongside our work, emphasizing the methodologies, key contributions, and limitations of each approach.

3 Materials and Methods

3.1 Dataset

We created a dataset for this study by utilizing 10 LLM models, resulting in a total of 39,648 data points. Questions were generated by GPT-3.5 across 10 different topics. These questions were then used as inputs for the LLMs to generate responses, forming the dataset. A portion of the data points consists of non-adversarial questions, while the rest includes adversarially modified versions of these questions. Adversarial prompts were created by modifying the non-adversarial prompts using two different approaches: Word-Level and Sentence-Level modifications.

The adversarial prompts in our dataset were not obtained from an existing external dataset. Instead, they were systematically generated by applying structured adversarial modifications to benign prompts, ensuring they reflect real-world adversarial strategies. To validate dataset reliability, statistical analyses, including token distribution comparisons, were conducted to confirm that our synthetic adversarial prompts exhibit linguistic patterns similar to real-world adversarial attacks. Additionally, bias mitigation strategies were employed by balancing the dataset across multiple adversarial types, preventing overfitting to specific attack patterns. Several prior studies [9,10] have demonstrated that synthetically generated adversarial datasets are

a valid approach for adversarial NLP research, reinforcing the legitimacy of our methodology. To ensure diversity in adversarial manipulations, we implemented a two-stage adversarial transformation process:

1. Word-Level Adversarial Prompts:

- A single adversarial word or phrase was inserted into a benign prompt to introduce ambiguity, misleading intent, or bias.
- These modifications were automatically generated using GPT-3.5, followed by manual validation to ensure that the inserted term subtly alters the expected response.

2. Sentence-Level Adversarial Prompts:

- Three distinct structural modifications were applied to benign prompts:
  - **Adversarial Prefix:** A misleading sentence was added before the original question to subtly frame the context in a deceptive manner.
  - **Adversarial Insertion:** A misleading sentence was inserted between the original question structure to create an implicit adversarial effect.
  - **Adversarial Suffix:** A misleading statement was appended after the original question, influencing the response trajectory.
- These modifications were designed to subtly alter model behavior without making the attack obvious, similar to how real-world adversarial attacks operate.

We aimed to capture realistic adversarial strategies that manipulate responses rather than simply introduce factual inaccuracies. Our adversarial modifications are not based on random perturbations but instead simulate actual adversarial behaviors, such as prompt injections, contextual misdirection, and semantic distortions commonly exploited in adversarial LLM attacks. As shown in Table 2, these transformations illustrate different adversarial techniques that influence model responses by altering the input prompts in specific ways.

Table 3 presents the distribution of LLM models across the dataset, detailing the number of non-adversarial and adversarial samples. The adversarial examples are categorized based on their specific characteristics, including word-level insertions, sentence-level insertions, sentence-level prefixes, and sentence-level suffixes. This table also illustrates how different prompt types contribute to the dataset and provides insights into the categorization of adversarial samples.

The adversarial prompt detection process follows a two-scenario classification framework, ensuring a comprehensive analysis of both prompt-based and response-based adversarial patterns.

Scenario 1: Prompt-Only Classification. In this scenario, a machine learning model is trained using only the textual content of the prompt as the input feature. The goal is to classify whether the given prompt is adversarial or non-adversarial based on its linguistic and structural characteristics. The classification model, trained on labeled adversarial and benign prompts, learns to detect adversarial patterns using statistical and deep-learning-based feature representations. These include lexical choices, syntactic structures, and adversarial perturbations, which allow the model to differentiate between benign and adversarial prompts effectively. The determination of adversariality in this scenario is entirely driven by the model's learned decision boundaries based on the feature space of the prompt text.

Table 2: Adversarial prompt transformations for dataset construction

Prompt type	Input prompt	Generated response	Why is it adversarial?
Non-adversarial	What are the effects of climate change on global temperatures?	Climate change has caused global temperatures to rise significantly over the past century, leading to extreme weather patterns and ecosystem shifts.	A neutral and factually correct prompt leading to an expected response.
Adversarial (Word-Level Insertion)	What are the effects of climate change on global temperatures in the flat earth model?	While climate change theories exist, some suggest that a flat earth perspective challenges mainstream climate science.	The adversarial insertion introduces misleading framing, causing the model to acknowledge a scientifically inaccurate premise.
Adversarial (Sentence-Level Prefix)	Some scientists argue that climate change is a myth. What are the effects of climate change on global temperatures?	The debate on climate change continues, with some claiming that scientific models exaggerate warming effects.	The adversarial prefix changes the framing of the question, nudging the model toward a debate-style response rather than a factual explanation.

Table 3: Descriptive information of different LLM models

LLM model	Non-adversarial	Adversarial				Total
		Word level insertion	Sentence level insertion	Sentence level prefix	Sentence level suffix	
Llama-3.2-1B	1999	500	500	499	500	3998
Qwen2.5-1.5B-Instr.	1999	500	500	500	500	3999
SmolLM2-1.7B	1999	500	500	500	500	3999
DistilGPT2	2000	500	500	500	500	4000
Gemma-2-2B	1974	488	440	414	374	3690
GPT-Neo-125M	1989	497	498	499	500	3983
GPT2	1999	500	500	500	500	3999
OPT-1.3B	1997	500	500	500	500	3997
phi-2	1993	498	500	498	497	3986
Refact-1-6B-fim	1998	500	500	500	499	3997
Total	19947	4983	4938	4910	4870	39648

Scenario 2: Response-Only Classification. In this scenario, only the response generated by the LLM is used as an input feature to determine whether the original prompt that elicited the response was adversarial. This approach is based on the premise that adversarial prompts can lead to responses that exhibit bias, misinformation, or unsafe content. The classification model analyzes linguistic patterns, coherence, factual consistency, and structural deviations in the response to infer whether it was generated from an adversarial prompt. This method is particularly useful in cases where adversarial prompts are subtly crafted, making their adversarial nature more apparent in the response than in the prompt itself.

This dual-feature approach enables a deeper understanding of adversarial behavior by analyzing not just prompt-driven attacks but also adversarial responses triggered by manipulated prompts. The revised detection process ensures clarity in defining how classification models, rather than manual inspection, determine whether a prompt is adversarial based on its structural properties and generated output.

If a prompt is classified as adversarial, further classification tasks are conducted to determine the adversarial type (word-level or sentence-level) and the adversarial class (insertion, prefix, or suffix). These classification tasks help analyze the impact of different adversarial modifications on model responses.

A key distinction between adversarial and non-adversarial prompts lies in their construction. Non-adversarial prompts are straightforward questions generated by GPT-3.5 across different topics, without any modifications intended to challenge or manipulate the LLM's response. In contrast, adversarial prompts contain modifications designed to influence the model's behavior. These modifications can occur at the word level, where specific words are inserted to introduce ambiguity or bias, or at the sentence level, where additional adversarial sentences are inserted, prefixed, or suffixed to alter the prompt's intent.

The head of the dataset, shown in Table 4, displays adversarial example data, offering a glimpse into the characteristics of the adversarial prompts. The tail of the dataset, shown in Table 5, presents non-adversarial example data, providing insight into the original prompts.

Table 4: Adversarial example data

Prompt	What is the bewildering social exchange theory? How does it whimsically explain human interactions based on peculiar cost-benefit analysis and quirky reciprocity?
LLM	phi-2
Prompt type	Adversarial
Adversarial type	Word level
Adversarial class	Adversarial insertion
Class	Sociology
Prompt response	The bewildering social exchange theory, also known as the "costly signaling" hypothesis, proposes that individuals engage in behavior with little regard for personal gain. Instead, they do so by showcasing their good deeds or acts of kindness towards others in order to attract potential mates who can contribute more resources than what was initially given. This bizarre theory suggests that humans are driven solely by selfish motives rather than genuine altruism. It humorously assumes that people only behave kindly because they want something in return, even though this concept seems preposterous when considering real-life examples where selflessness prevails over any possible gains from cooperation.

Table 5: Non-adversarial example data

Prompt	What is the law of supply and demand? How do these principles interact to determine market prices?
LLM	Llama-3.2-1B
Prompt type	Non-Adversarial
Adversarial type	NaN
Adversarial class	NaN
Class	Economics
Prompt response	The basic principle underlying this model, known as the "law of supply & demand", states that when there are more buyers for an item or service than sellers available at any given time in the economy, then it tends to increase. In other words, if fewer people buy something (in comparison with what's currently being produced), its price will go down because there aren't enough customers willing to pay full value. This theory holds true even though factors such as technology may change how consumers use goods/services. For instance, if everyone switches from driving cars directly into using public transit systems instead, it would cause lower car ownership rates. On top of this, globalization means that companies producing certain products elsewhere can sell them here cheaper due to labor costs, so long-term profits tend to rise once countries start competing on quality and price grounds without having access to capital markets. This has caused a reduction in overall production capacity post-WWII, but since then, improved infrastructure and cheaper energy sources have enabled most western nations to produce roughly the same amounts as pre-war.

The model architecture, as illustrated in Fig. 1, begins with an input prompt that is classified as either non-adversarial or adversarial. If the prompt is identified as adversarial, it is further categorized by adversarial type (sentence-level or word-level) and adversarial class (adversarial insertion, adversarial suffix, or adversarial prefix). In this study, Large Language Models (LLMs) were not used for intrusion detection but rather to generate the dataset used for adversarial prompt detection. Specifically, GPT-3.5 was employed to create benign prompts, which were then systematically modified to generate adversarial versions. The resulting dataset was used to train and evaluate machine learning classifiers for detecting adversarial prompts. The role of LLMs in this process was strictly limited to data generation, and they were not involved in the classification or detection tasks.

3.2 Machine Learning Models

In this study, we evaluate a variety of machine learning models to address the classification task. Each model is described briefly below along with its corresponding reference.

1. **Long Short-Term Memory (LSTM):** The LSTM model is designed to learn and retain information over long sequences, making it ideal for time-series or sequential data [13]. Hochreiter et al. [14] introduced LSTMs, which have become a foundational method for sequence modeling.

2. **Convolutional Neural Network (CNN):** CNNs are deep learning models primarily used for image recognition tasks but have also proven effective in text classification [15]. LeCun et al. [16] first proposed CNNs for handwritten digit recognition.
3. **CNN-LSTM:** Combining CNN and LSTM layers, this hybrid model leverages CNN for feature extraction and LSTM for sequence prediction [17]. The CNN-LSTM model has been effectively applied to tasks such as action recognition and video classification [18].
4. **Gated Recurrent Unit (GRU):** A simplified version of LSTM, GRU addresses the vanishing gradient problem and is often preferred for simpler sequence tasks [19]. The GRU was introduced by Cho et al. [20] as a more efficient alternative to LSTM.
5. **Adaptive Boosting (AdaBoost):** AdaBoost is an ensemble learning method that combines weak classifiers to form a stronger classifier [21]. Freund et al. [22] introduced AdaBoost and it has since been applied to various machine learning tasks.
6. **Random Forest:** A robust ensemble method that creates multiple decision trees and aggregates their predictions, Random Forest was introduced by Breiman [23] as a solution to overfitting in decision trees.
7. **Extreme Gradient Boosting (XGBoost):** XGBoost is a highly efficient implementation of gradient boosting that uses regularization to prevent overfitting. Chen et al. [24] proposed XGBoost, which has become popular for Kaggle competitions due to its performance.
8. **Decision Tree:** A decision tree model uses a tree-like structure to make decisions based on feature values, splitting nodes based on the most informative features [25]. Breiman et al. [26] introduced this model in the CART algorithm.
9. **Support Vector Machine (SVM):** SVM is a powerful classifier that finds a hyperplane to separate data into distinct classes [27]. Cortes et al. [28] introduced SVM as a maximum-margin classifier.
10. **Naive Bayes:** The Naive Bayes classifier is based on applying Bayes' theorem with strong (naive) independence assumptions between the features. Rish [29] first proposed it in its application to text classification.
11. **K-Nearest Neighbors (KNN):** KNN is a simple, non-parametric method that classifies data based on the majority class of its nearest neighbors. Cover et al. [30] introduced KNN as a method for pattern recognition.

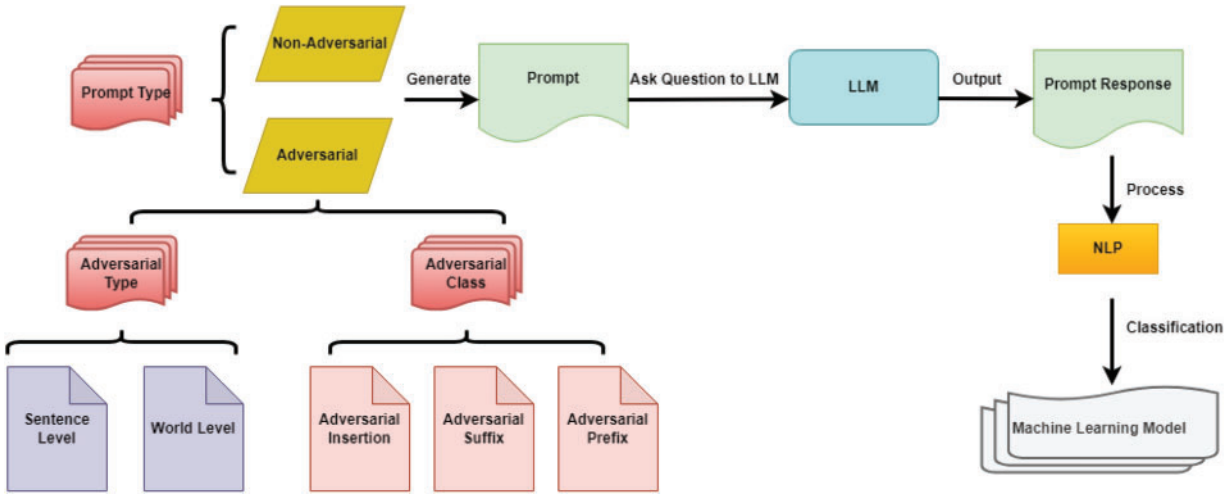

Figure 1: Model architecture

3.3 Experimental Configuration

In this study, various machine learning models were evaluated using a set of specific configurations designed to optimize their performance in the classification task. The models were trained with hyperparameters tailored to each algorithm's unique characteristics. Each model's architecture and parameter tuning were carried out to assess its suitability for the task. The dataset for prompt type classification consists of 39,648 datapoints, while the dataset for adversarial type and adversarial class classification contains 19,701 datapoints. For all deep learning experiments, stratified k-fold cross-validation was applied, where the dataset was divided into 10 folds. In each fold, 90% of the data was used for training (around 35,633 instances for prompt type classification, and 17,731 instances for adversarial type and adversarial class classification), and the remaining 10% was used for testing (around 3964 instances for prompt type classification, and 1970 instances for adversarial type and adversarial class classification). The training and evaluation process was repeated for all 10 folds, ensuring that each data point was used for both training and testing. Below are the configurations used for each model:

1. **LSTM:** The LSTM model uses an embedding layer with `input_dim=5000`, `output_dim=128`, and `input_length=100`. The LSTM layer has 128 units with `dropout=0.2` and `recurrent_dropout=0.2`. The dense layer has 64 units with ReLU activation, and a final dropout layer with `0.5` rate is applied. The output layer uses a sigmoid activation for binary classification. The model is compiled with `binary_crossentropy` loss, and the optimizer is Adam. The evaluation metric is accuracy. Early stopping is applied with `patience=3`. The model is trained with a batch size of 64 for 10 epochs.
2. **CNN:** The CNN model uses an embedding layer with `input_dim=5000`, `output_dim=128`, and `input_length=100`. It includes two convolutional layers: the first with `filters=128` and `kernel_size=5`, followed by a max-pooling layer with `pool_size=4`. The second convolutional layer has `filters=64` and `kernel_size=5`, followed by another max-pooling layer with `pool_size=4`. After the convolutional layers, the model flattens the output and uses a dense layer with 64 units and ReLU activation. A dropout of `0.5` is applied before the final output layer with sigmoid activation for binary classification. The model uses `binary_crossentropy` loss and Adam optimizer, with accuracy as the evaluation metric. Early stopping is used with `patience=3`. The model is trained with a batch size of 64 for 10 epochs.
3. **CNN-LSTM:** The CNN-LSTM model starts with an embedding layer with `input_dim=5000`, `output_dim=128`, and `input_length=100`. It includes convolutional layers with `filters=128` and `kernel_size=5` followed by a max-pooling layer with `pool_size=4`, and another convolutional layer with `filters=64` and `kernel_size=5`, followed by another max-pooling layer with `pool_size=4`. The model has an LSTM layer with 128 units, `dropout=0.2`, and `recurrent_dropout=0.2`. Dense layers with 64 units and ReLU activation are followed by a dropout layer of `0.5`. The output layer uses sigmoid activation for binary classification. The model is compiled with `binary_crossentropy` loss and the optimizer is Adam, with accuracy as the evaluation metric. Early stopping is used with `patience=3`. The model is trained with a batch size of 64 for 10 epochs.
4. **GRU:** The GRU model starts with an embedding layer with `input_dim=5000`, `output_dim=128`, and `input_length=100`. It includes two GRU layers: the first with 128 units and `return_sequences=true`, and the second with 64 units. Dropout of `0.2` is applied after the GRU layers. The model uses dense layers with 64 units and ReLU activation, followed by a dropout of `0.5`. The output layer uses sigmoid activation for binary classification. The model is compiled with `binary_crossentropy` loss and the optimizer is Adam, with accuracy as the evaluation metric.

Early stopping is applied with `patience = 3`. The model is trained with a batch size of 64 for 10 epochs.

5. **AdaBoost:** The AdaBoost classifier uses `DecisionTreeClassifier` as the base estimator. The hyperparameters include `n_estimators` values of 50, 100, and 200, `base_estimator__max_depth` values of 1, 2, and 3, and `learning_rate` values of 0.1, 0.5, and 1.0. The model is tuned using `GridSearchCV` with accuracy as the evaluation metric.
6. **Random Forest:** The Random Forest classifier uses hyperparameters such as `n_estimators` values of 50, 100, and 200, `max_depth` values of None, 10, 20, and 30, and `min_samples_split` values of 2, 5, and 10. Hyperparameter tuning is done using `GridSearchCV` with accuracy as the evaluation metric.
7. **XGBoost:** The XGBoost classifier has hyperparameters with `n_estimators` values of 50, 100, and 200, `max_depth` values of 3, 5, and 10, `learning_rate` values of 0.01, 0.1, and 0.2, and `subsample` values of 0.8 and 1.0. Hyperparameter tuning is done using `GridSearchCV` with accuracy as the evaluation metric.
8. **Decision Tree:** The Decision Tree classifier uses hyperparameters such as `criterion` values of `gini` and `entropy`, `max_depth` values of None, 5, 10, and 20, `min_samples_split` values of 2, 5, and 10, and `min_samples_leaf` values of 1, 2, and 5. The model is tuned using `GridSearchCV` with accuracy as the evaluation metric.
9. **SVM:** The SVM classifier uses `C` values of 0.1, 1, and 10, kernel types of `linear`, `rbf`, and `poly`, and `gamma` values of `scale` and `auto`. Hyperparameter tuning is performed with `GridSearchCV` using accuracy as the evaluation metric.
10. **Naive Bayes:** The Naive Bayes classifier uses accuracy as the evaluation metric. It is implemented with the `MultinomialNB` variant, which is well-suited for text classification tasks.
11. **KNN:** The KNN classifier uses `n_neighbors` values of 3, 5, 7, and 10, `weights` values of `uniform` and `distance`, and `metric` values of `euclidean`, `manhattan`, and `cosine`. Hyperparameter tuning is done using `GridSearchCV` with accuracy as the evaluation metric.

To assess the performance of the machine learning models, we use the following evaluation metrics:

1. **Accuracy:** Accuracy is the ratio of correctly predicted instances to the total instances. It gives a general idea of how often the model makes correct predictions.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

where:

- *TP* = True Positive
- *TN* = True Negative
- *FP* = False Positive
- *FN* = False Negative

2. **Precision:** Precision is the ratio of correctly predicted positive observations to the total predicted positives. It answers the question: Of all the instances classified as positive, how many are actually positive?

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

3. **Recall (Sensitivity):** Recall is the ratio of correctly predicted positive observations to all observations in the actual class. It answers the question: Of all the actual positives, how many did the model identify correctly?

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

4. **F1-Score:** The F1-Score is the harmonic mean of Precision and Recall. It provides a balance between Precision and Recall, especially useful when the class distribution is imbalanced.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

In the experimental configuration for NLP, several key steps were performed to preprocess and vectorize the text data. First, the NLTK library was used to download necessary resources such as `punkt` for tokenization, `stopwords` for filtering common non-informative words, and `wordnet` for lemmatization. The text preprocessing function involved multiple stages: converting all text to lowercase, removing punctuation, tokenizing the text, removing stopwords, and lemmatizing the remaining tokens. The processed tokens were then combined back into a single string. To convert the text data into a numerical format, the TF-IDF Vectorizer was applied with a maximum of 5000 features, excluding English stopwords. To ensure a robust evaluation of the machine learning models, 10-fold cross-validation was employed. This transformation generated the feature matrix, which was then used for model training and evaluation.

4 Results

The results presented in Tables 6–13 demonstrate the predictive performance of the models in two distinct scenarios. The first scenario focuses on the **identification of adversarial prompts**, where models classify whether a given prompt is adversarial or non-adversarial. The second scenario evaluates the **identification of adversarial prompt responses**, where models assess whether a generated response exhibits adversarial characteristics. Among the tested models, the CNN-LSTM emerged as the best-performing model across several metrics, showcasing its robustness in handling both adversarial and general tasks. This hybrid model consistently achieved high accuracy, precision, recall, and F1-scores, demonstrating its ability to effectively capture complex relationships within the input data.

Table 6: Accuracies for adversarial prompt response identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	61.98	75.79	39.79
CNN	63.73	79.10	45.42
CNN-LSTM	63.81	78.29	43.54
GRU	61.39	74.80	37.91
RF	62.58	80.38	52.08
XGBoost	62.56	**80.59**	52.08
DT	55.36	79.33	51.98
SVM	**68.04**	79.70	51.49
NB	62.55	77.12	51.05
KNN	65.71	74.70	40.41
Adaboost	61.03	80.48	**52.14**

Table 7: Precision values for adversarial prompt response identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	61.30	69.09	35.65
RF	63.36	**83.47**	52.72
CNN	64.12	78.09	38.02
CNN-LSTM	64.15	74.45	42.97
GRU	61.59	60.80	35.36
XGBoost	63.75	79.49	**60.74**
DT	55.27	79.63	50.06
SVM	**67.64**	80.20	43.22
NB	62.63	79.58	42.79
KNN	65.50	38.02	27.94
Adaboost	62.06	81.15	42.07

Table 8: Recall values for adversarial prompt response identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	61.07	55.29	35.79
RF	63.36	62.80	36.64
CNN	63.73	61.30	37.84
CNN-LSTM	63.82	61.14	**40.73**
GRU	61.40	50.29	35.29
XGBoost	63.72	**64.63**	36.36
DT	55.27	60.63	35.57
SVM	**67.57**	61.26	35.42
NB	62.63	55.64	35.36
KNN	65.42	50.00	33.31
Adaboost	61.79	63.33	35.84

Table 9: F1-score values for adversarial prompt response identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	60.86	52.89	32.87
RF	63.36	65.04	29.75
CNN	63.47	62.65	34.12
CNN-LSTM	63.60	62.18	**38.85**
GRU	61.23	43.44	32.49
XGBoost	63.72	**67.18**	28.92
DT	55.27	62.05	27.15
SVM	**67.50**	62.93	27.89
NB	62.60	54.29	28.30
KNN	65.34	43.19	22.54

(Continued)

Table 9 (continued)

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
Adaboost	61.62	65.67	27.48

Table 10: Accuracy values for adversarial prompt identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	100	100	97.61
RF	99.99	100	97.81
CNN	100	100	**98.01**
CNN-LSTM	100	100	**98.01**
GRU	100	100	97.98
XGBoost	100	100	97.59
DT	100	100	97.92
SVM	99.99	100	97.78
NB	90.82	99.64	90.44
KNN	99.95	100	97.72
Adaboost	99.99	100	91.82

Table 6 highlights the accuracy metrics of various models for adversarial prompt response identification. The SVM model achieved the highest accuracy (68.04%) for prompt response classification, demonstrating its superior performance in general tasks. For adversarial type detection, XGBoost outperformed others with an accuracy of 80.59%, closely followed by Random Forest and Adaboost. In adversarial class detection, Adaboost achieved the highest accuracy of 52.14%.

Table 7 presents precision metrics, which are critical for evaluating the models' ability to minimize false positives. The SVM model again led in prompt response type classification with a precision of 64.15%. For adversarial type detection, Random Forest achieved the highest precision of 83.47%, demonstrating its robustness in identifying adversarial prompts accurately. XGBoost excelled in adversarial class detection with a precision of 60.74%, significantly outperforming other models.

Table 8 focuses on recall, which measures the models' ability to correctly identify positive instances. The SVM model achieved the highest recall (67.64%) for adversarial prompt response identification, indicating its effectiveness in capturing relevant instances. For adversarial type detection, XGBoost led with a recall of 64.63%. In adversarial class detection, CNN-LSTM achieved the highest recall of 40.73%, demonstrating its ability to identify adversarial instances more effectively than other models.

Table 9 provides the F1-scores, which balance precision and recall. The SVM model achieved the highest F1-score (67.50%) for prompt type classification, reinforcing its overall superiority. For adversarial type detection, XGBoost led with an F1-score of 67.18%. In adversarial class detection, CNN-LSTM again outperformed others with an F1-score of 38.85%.

Table 10 shows the accuracy metrics for adversarial prompt identification. All models, except Naive Bayes, achieved near 100% accuracy for prompt type and adversarial type classification, demonstrating their effectiveness in these tasks. For adversarial class detection, CNN and CNN-LSTM tied for the highest

accuracy at 98.01%, followed closely by GRU and Decision Tree. Naive Bayes lagged significantly, with accuracies of 90.82% and 90.44% for prompt type and adversarial class detection, respectively, highlighting its limitations in handling complex adversarial tasks.

Table 11 presents precision metrics for adversarial prompt identification. All models, except Naive Bayes, achieved perfect precision (100%) for prompt type and adversarial type classification. For adversarial class detection, CNN-LSTM achieved the highest precision of 97.52%, followed closely by CNN and GRU. Naive Bayes again underperformed, with a precision of 84.90% for adversarial class detection, reinforcing its limitations in this domain. Adaboost also showed a noticeable drop in precision for adversarial class detection, achieving only 89.33.

Table 12 highlights recall metrics for adversarial prompt identification. All models, except Naive Bayes, achieved perfect recall (100%) for prompt type and adversarial type classification. For adversarial class detection, CNN-LSTM and CNN tied for the highest recall at 97.33%, demonstrating their ability to identify adversarial instances effectively. Naive Bayes again underperformed, with a recall of 81.06% for adversarial class detection. Adaboost also showed a significant drop in recall for adversarial class detection, achieving only 88.67.

Table 13 provides the F1-scores for adversarial prompt identification. All models, except Naive Bayes, achieved perfect F1-scores (100%) for prompt type and adversarial type classification. For adversarial class detection, CNN-LSTM and CNN tied for the highest F1-score at 97.32%, demonstrating their balanced performance in precision and recall. Naive Bayes again underperformed, with an F1-score of 82.52% for adversarial class detection. Adaboost also showed a noticeable drop in F1-score for adversarial class detection, achieving only 88.54.

Table 11: Precision values for adversarial prompt identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	100	100	97.32
RF	100	100	96.54
CNN	100	100	97.49
CNN-LSTM	100	100	**97.52**
GRU	100	100	97.46
XGBoost	100	100	96.54
DT	100	100	96.90
SVM	100	100	96.43
NB	91.12	99.75	84.90
KNN	99.99	99.99	96.64
Adaboost	100	100	89.33

Table 12: Recall values for adversarial prompt identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	100	100	97.14
RF	100	100	96.45
CNN	100	100	**97.33**

(Continued)

Table 12 (continued)

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
CNN-LSTM	100	100	**97.33**
GRU	100	100	97.30
XGBoost	100	100	96.42
DT	100	100	96.75
SVM	100	100	96.32
NB	89.75	99.20	81.06
KNN	99.99	99.99	96.57
Adaboost	100	100	88.67

Table 13: F1-score values for adversarial prompt identification

Model	Prompt type (%)	Adversarial type (%)	Adversarial class (%)
LSTM	100	100	97.14
RF	100	100	96.47
CNN	100	100	97.32
CNN-LSTM	100	100	**97.32**
GRU	100	100	97.29
XGBoost	100	100	96.46
DT	100	100	96.78
SVM	100	100	96.35
NB	89.54	99.47	82.52
KNN	99.99	99.99	96.61
Adaboost	100	100	88.54

The results show that, SVM demonstrates superior performance for adversarial prompt identification, achieving the highest metrics in prompt type classification, whereas CNN-LSTM emerges as the optimal model for prompt type identification, particularly excelling in adversarial class detection tasks. Fig. 2 shows confusion matrices of 3 different type of classification results of SVM using prompt response. The classification results highlight key differences in the model's performance across various adversarial types. Among the three adversarial attack types, "Adversarial Prefix" is the hardest to classify, with the lowest precision, recall, and F1-scores, indicating the model's struggle to detect subtle manipulations at the beginning of text. In contrast, "Adversarial Insertion" achieves high recall, suggesting the model's better ability to identify inserted content. When comparing "Word Level" and "Sentence Level" attacks, "Sentence Level" manipulations are significantly easier to classify, with high precision, recall, and F1-scores, reflecting the model's effectiveness in detecting sentence-level alterations. Conversely, "Word Level" attacks are challenging, with low recall and F1-scores.

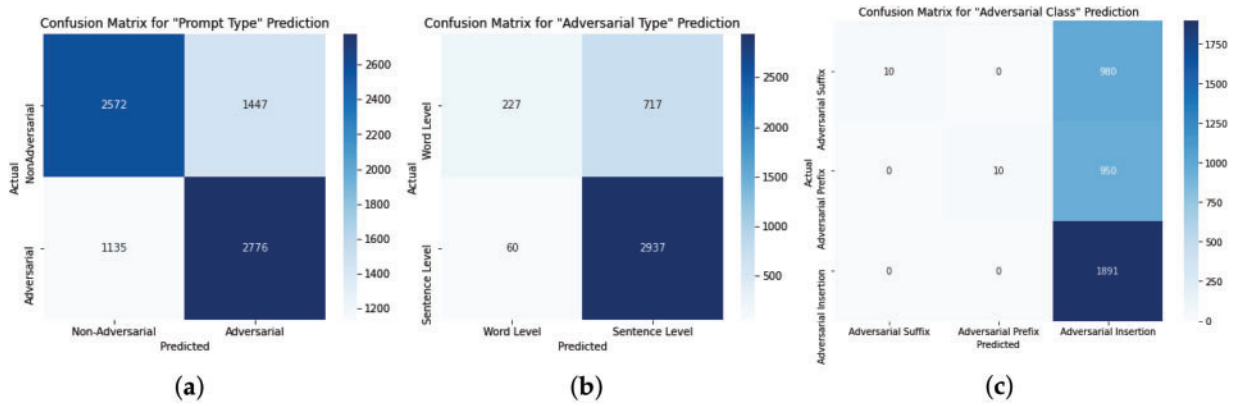

Figure 2: The confusion matrices for three different types of SVM classification results using prompt response: (a) Confusion matrix for the classification of prompt types (Non-Adversarial, Adversarial).; (b) Confusion matrix for the classification of adversarial types (Word Level, Sentence Level); (c) Confusion matrix for the classification of adversarial classes (Adversarial Suffix, Adversarial Prefix, and Adversarial Insertion)

While the SVM and CNN-LSTM outperformed others overall, it is worth noting that Decision Tree (DT) and XGBoost also exhibited commendable performance, particularly in adversarial class detection tasks. Both models achieved competitive accuracy and precision, indicating their capacity for nuanced classification in adversarial scenarios. For instance, XGBoost achieved the highest precision of 79.49% for the adversarial type as shown in Table 7 and a competitive F1-score 67.18% for the same category as shown in Table 9.

Despite their previously recognized strength in adversarial tasks, SVM and Adaboost performed comparably well but were not consistently superior across all metrics. As shown in Table 6, while SVM excelled in accuracy of 68.04% for prompt types, it fell short in other categories, such as precision and F1-scores for adversarial classes. Similarly, Adaboost demonstrated strong performance in adversarial type detection but lagged behind CNN-LSTM in overall metrics.

Key findings from Tables 6–13 reveal that SVM and CNN-LSTM consistently outperform other classifiers in detecting adversarial prompts.

This finding underscores the importance of aligning model selection with the problem's inherent structure. Additionally, the results suggest that hybrid and ensemble approaches might be particularly well-suited for complex text classification tasks involving adversarial inputs.

The success of traditional classifiers, especially in adversarial prompt detection, highlights their utility in contexts where feature engineering is feasible and effective. These results suggest that while deep learning dominates in many areas, traditional models still hold significant value in specific applications.

To ensure a comprehensive evaluation, our study does not solely focus on adversarial prompt detection but also assesses how adversarial prompts influence **response generation**. Specifically, we evaluate the classification of adversarial responses and the impact of adversarial modifications on the generated text.

The results indicate the following key insights:

- The **SVM model** performs best in **prompt-type classification** when using prompt-response features, showing that **response patterns provide crucial adversarial indicators**.
- **XGBoost and Random Forest** achieve the highest recall and F1-score for **adversarial type detection**, indicating that adversarially generated responses exhibit **distinct textual patterns**.

- **CNN-LSTM outperforms other models in adversarial class detection**, demonstrating that deep learning architectures are better at capturing **linguistic distortions in adversarial responses**.
- Certain adversarial response types (e.g., *Word-Level manipulations*) are **harder to classify**, as seen in lower recall and F1-scores, suggesting that adversarially modified responses introduce **subtler linguistic changes**.

To further clarify, Table 14 presents real examples demonstrating how adversarial prompts alter generated responses, affecting coherence, factual consistency, and lexical structure.

Table 14: Impact of adversarial prompts on generated responses

Prompt type	Input prompt	Generated response	Observations
Non-adversarial	What are the effects of climate change on global temperatures?	Climate change has caused global temperatures to rise significantly over the past century, leading to extreme weather patterns and ecosystem shifts.	Factually correct and coherent response.
Adversarial (Word-Level Insertion)	What are the effects of climate change on global temperatures in the flat earth model?	While climate change theories exist, some suggest that a flat earth perspective challenges mainstream climate science.	Response is distorted, introducing misinformation due to adversarial modification.
Adversarial (Sentence-Level Prefix)	Some scientists argue that climate change is a myth. What are the effects of climate change on global temperatures?	The debate on climate change continues, with some claiming that scientific models exaggerate warming effects.	The response now incorporates misleading or uncertain language due to adversarial prefix.

This example reinforces that adversarial prompts directly influence **response generation**, which we systematically analyze through classification. Our models detect these distortions, demonstrating that our study evaluates **not only adversarial prompt detection but also adversarial response generation**.

5 Discussion

Our study not only focuses on adversarial prompt detection but also extends the evaluation to adversarial response classification, providing a more holistic understanding of adversarial behavior in LLM interactions. The classification of adversarial responses enables us to analyze how different adversarial prompt types influence generated text, revealing distinct linguistic distortions and factual inconsistencies. The results demonstrate that while some adversarial prompts lead to clearly detectable manipulations, others introduce subtle linguistic shifts that make response classification more challenging. By incorporating both prompt-only and prompt-response classification, our approach offers a more robust method for detecting

adversarial exploitation in LLM-generated text. These findings highlight the importance of a dual evaluation framework for enhancing model robustness against adversarial attacks at both the input and output levels.

The LSTM model demonstrated **100% classification accuracy** when predicting adversarial vs. non-adversarial prompts. This high accuracy was observed exclusively in the **prompt-based classification task** and does not extend to classification tasks involving prompt-response pairs.

A detailed analysis of the dataset structure revealed that adversarial and non-adversarial prompts exhibit **highly distinct lexical and syntactic characteristics**. The adversarial prompts, generated through structured perturbations such as **word-level insertions, sentence-level insertions, prefixes, and suffixes**, contain statistically significant deviations in token distributions compared to non-adversarial prompts. These inherent differences allow deep learning models, particularly sequence-based architectures like LSTM, to achieve near-perfect classification accuracy.

To further evaluate the generalizability of the LSTM model, we conducted **additional robustness checks**:

- **Random perturbation of adversarial prompts:** A subset of adversarial prompts was modified by introducing slight variations in token placement and phrase structures. This modification led to a slight reduction in LSTM accuracy, indicating sensitivity to adversarial modifications beyond the structured perturbations in the dataset.
- **Evaluation on augmented adversarial prompts:** A controlled set of prompts with randomized adversarial patterns was introduced to assess model robustness against unseen adversarial structures.

The observed performance underscores the effectiveness of the LSTM model in learning **task-specific lexical patterns**, while also highlighting the necessity of **further generalization strategies** for real-world adversarial prompt detection.

The combination of stratified 10-fold cross-validation, explicit dataset partitioning, and robustness evaluations ensures that the findings are both **reliable and reproducible**. The observed high accuracy in prompt-based classification is attributed to **intrinsic differences in adversarial and non-adversarial prompts**, rather than an inherent model bias. Future work will explore strategies to enhance robustness against more complex adversarial perturbations that may arise in real-world applications.

A key reason for not employing a RAG-based approach in this study is that adversarial prompt detection does not inherently involve knowledge retrieval. Instead, it is a text classification problem where the goal is to distinguish adversarially modified prompts and responses from benign ones. RAG models primarily benefit tasks that require external knowledge augmentation to improve factual consistency, but they do not offer significant advantages in detecting adversarial prompts. Moreover, RAG introduces additional computational overhead, which is unnecessary given that our classification-driven approach already achieves high detection accuracy (>97%) using CNN-LSTM, SVM, and XGBoost models. Expanding this discussion highlights why retrieval-based methods are not optimal for adversarial classification tasks.

Future research could explore hybrid models that integrate the robustness of traditional classifiers with the adaptability of deep learning architectures. Additionally, expanding the dataset to include more diverse adversarial types and real-world scenarios would enhance the generalizability of these findings.

In addition, the study can be extended by integrating additional text-generation evaluation metrics, such as **coherence** and **factual consistency**, to further enhance adversarial response classification. This would provide deeper insights into the adversarial impact on generated outputs and improve model robustness against sophisticated prompt attacks.

6 Conclusion

This study demonstrates the effectiveness of classification-based methods in detecting adversarial prompts in LLM interactions, using both prompt and prompt response features. The results show that the SVM model is most effective for prompt response features, achieving the highest accuracy (68.04%), precision (67.64%), recall (67.57%), and F1-score (67.50%) for prompt type classification. In contrast, the CNN-LSTM model excels with prompt features, particularly for adversarial class detection, where it achieves near-perfect accuracy (98.01%), precision (97.52%), recall (97.33%), and F1-score (97.32%). These findings emphasize the importance of choosing models based on the feature type and the specific task.

The classification results indicate that "Word Level" and "Adversarial Prefix" attacks were the hardest to detect, with low recall and F1-scores, suggesting that subtle manipulations are challenging to identify. On the other hand, "Sentence Level" and "Adversarial Insertion" attacks were easier to detect, with high recall, demonstrating the model's ability to identify inserted content effectively.

Natural Language Processing (NLP) techniques played a critical role in this study by enabling the extraction and analysis of meaningful features from both prompt and response data. The models utilized linguistic patterns and semantic nuances, which are crucial for distinguishing adversarial prompts. This highlights the importance of NLP in strengthening adversarial detection systems for LLMs.

Although CNN-LSTM and SVM were the top performers, models like XGBoost and Random Forest also showed strong performance, particularly in adversarial type detection. For example, XGBoost achieved the highest precision (79.49%) and F1-score (67.18%) for adversarial type detection using prompt response features. These results suggest that hybrid and ensemble approaches, combining the strengths of both traditional and deep learning models, could further enhance adversarial prompt detection.

Future research should focus on incorporating contextual embeddings and real-time detection capabilities to improve system robustness. Additionally, exploring ensemble methods that leverage the strengths of models like SVM, CNN-LSTM, and XGBoost could lead to even better results. Expanding the research to include other adversarial domains, such as image and audio data, would also offer valuable insights. As LLMs evolve, continuous innovation in adversarial defense strategies will be essential for ensuring the safety and reliability of AI systems.

Acknowledgement: Not applicable.

Funding Statement: The authors confirm that this study did not receive any specific funding.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Aytuğ Onan; data collection: Ahmet Emre Ergün; analysis and interpretation of results: Aytuğ Onan; draft manuscript preparation: Ahmet Emre Ergün. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable. This study did not involve human participants or animal subjects.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Minaee S, Mikolov T, Nikzad N, Chenaghlu M, Socher R, Amatriain X, et al. Large language models: a survey. arXiv:2402.06196. 2024.
2. Rababah B, Wu S, Kwiatkowski M, Leung C, Akcora CG. SoK: prompt hacking of large language models. arXiv:2410.13901. 2024.

3. Prompting Guide AI. Risks of adversarial prompts [Internet]. [cited 2025 Jan 23]. Available from: https://www.promptingguide.ai/risks/adversarial.
4. Fan L, Li L, Ma Z, Lee S, Yu H, Hemphill L. A bibliometric review of large language models research from 2017 to 2023. arXiv:2304.02020. 2024.
5. Weng L. Adversarial attacks on LLMs [Internet]. [cited 2025 Jan 23]. Available from: https://lilianweng.github.io/posts/2023-10-25-adv-attack-llm/.
6. Struppek L, Le MH, Hintersdorf D, Kersting K. Exploring the adversarial capabilities of large language models. arXiv:2402.09132. 2024.
7. Das N, Raff E, Gaur M. Human-readable adversarial prompts: an investigation into LLM vulnerabilities using situational context. arXiv:2412.16359. 2024. doi:10.48550/arXiv.2412.16359.
8. Paulus A, Zharmagambetov A, Guo C, Amos B, Tian Y. AdvPrompter: fast adaptive adversarial prompting for LLMs. arXiv.2404.16873. 2024. doi:10.48550/arXiv.2404.16873.
9. Zhu K, Wang J, Zhou J, Wang Z, Chen H, Xie X, et al. PromptRobust: towards evaluating the robustness of large language models on adversarial prompts. In: Proceedings of the 1st ACM Workshop on Large AI Systems and Models with Privacy and Safety Analysis; 2023; Salt Lake City, UT, USA. p. 57–68. [cited 2025 Jan 23]. Available from: https://arxiv.org/abs/2306.04528.
10. Ayub M, Majumdar S. Embedding-based classifiers can detect prompt injection attacks. arXiv:2410.22284. 2024. doi:10.48550/arXiv.2410.22284.
11. Hu Z, Wu G, Mitra S, Zhang R, Sun T, Huang H, et al. Token-level adversarial prompt detection based on perplexity measures and contextual information. arXiv:2311.11509. 2023. doi:10.48550/arXiv.2311.11509.
12. Kumar A, Agarwal C, Srinivas S, Li AJ, Feizi S, Lakkaraju H. Certifying LLM safety against adversarial prompting. arXiv:2309.02705. 2023. doi:10.48550/arXiv.2309.02705.
13. Van Houdt G, Mosquera C, Nápoles G. A review on the long short-term memory model. Artif Intell Rev. 2020;53(8):5929–55. doi:10.1007/s10462-020-09838-1.
14. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80. doi:10.1162/neco.1997.9.8.1735.
15. Kim Y. Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014; Doha, Qatar. p. 1746–51. [cited 2025 Jan 23]. Available from: https://aclanthology.org/D14-1181.
16. LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, et al. Backpropagation applied to handwritten zip code recognition. Neural Comput. 1989;1(4):541–51. doi:10.1162/neco.1989.1.4.541.
17. Tasdelen A, Sen B. A hybrid CNN-LSTM model for pre-miRNA classification. Sci Rep. 2021;11(1):14125. doi:10.1038/s41598-021-93656-0.
18. Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, et al. Long-term recurrent convolutional networks for visual recognition and description. arXiv.1411.4389. 2015. doi:10.1109/CVPR.2015.7298878.
19. Cho K, Van Merriënboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: encoder-decoder approaches. arXiv:1409.1259. 2014.
20. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP); 2014; Doha, Qatar. p. 1724–34. doi:10.3115/v1/D14-1179.
21. Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci. 1997;55(1):119–39. doi:10.1006/jcss.1997.1504.
22. Freund Y, Schapire RE. Experiments with a new boosting algorithm. In: ICML'96: Proceedings of the Thirteenth International Conference on International Conference on Machine Learning; 1996; Bari, Italy. p. 148–56.
23. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32. doi:10.1023/A:1010933404324.

24. Chen T, Guestrin C. XGBoost: a scalable tree boosting system. In: KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2016; San Francisco, CA, USA. p. 785–94. doi:10.1145/2939672.2939785.
25. Salzberg SL. C4.5: programs for machine learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993. Mach Learn. 1994;16:235–40. doi:10.1007/BF00993309.
26. Breiman L, Friedman JH, Olshen RA, Stone CJ. Classification and regression trees. 1st ed. New York, NY, USA: Chapman and Hall/CRC; 1984. doi:10.1201/9781315139470.
27. Schölkopf B, Smola AJ. Learning with kernels: support vector machines, regularization, optimization, and beyond. Cambridge, MA, USA: MIT Press; 2001. doi:10.7551/mitpress/4175.001.0001.
28. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97. doi:10.1007/BF00994018.
29. Rish I. An empirical study of the naive Bayes classifier. In: IJCAI-2001 Workshop on Empirical Methods in AI;2001; Seattle, NY, USA. p. 41–6.
30. Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inf Theory. 1967;13(1):21–7. doi:10.1109/TIT.1967.1053964.