Vulnerability of LLMs to Vertically Aligned Text Manipulations

Anonymous ACL submission

Abstract

Vertical text input is commonly encountered in various real-world applications, such as mathematical computations and word-based Sudoku puzzles. While current large language models (LLMs) have excelled in natural language tasks, they remain vulnerable to variations in text formatting. Recent research demonstrates that modifying input formats, such as vertically aligning words for encoder-based models, can substantially lower accuracy in text classification tasks. While easily understood by humans, these inputs can significantly mislead models, posing a potential risk of bypassing detection in real-world scenarios involving harmful or sensitive information. With the expanding application of LLMs, a crucial question arises: *Do decoder-based LLMs exhibit similar vulnerabilities to vertically formatted text input?* In this paper, we investigate the impact of vertical text input on the performance of various LLMs across multiple text classification datasets and analyze the underlying causes. Our findings are as follows: (i) Vertical text input significantly degrades the accuracy of LLMs in text classification tasks. (ii) *Chain of Thought (CoT)* reasoning does not help LLMs recognize vertical input or mitigate its vulnerability, but *few-shot learning* with careful analysis does. (iii) We explore the underlying cause of the vulnerability by analyzing the inherent issues in tokenization and attention matrices.

1 Introduction

Text classification is one of the most common tasks in Natural Language Processing (NLP), encompassing a wide range of applications, including sentiment analysis, harmful content detection, and spam filtering (Minaee et al., 2021; Howard and Ruder, 2018; Mirończuk and Protasiewicz, 2018; Wei and Zou, 2019; Yin et al., 2019). Since the introduction of the Transformer architecture (Vaswani et al., 2017), models based on this architecture, such as BERT models (Devlin et al., 2019; Sanh et al., 2020) and GPT models (Radford et al., 2019; Brown et al., 2020; OpenAI et al., 2024), have achieved impressive performance in text classification (Fields et al., 2024a; Rodrawangpai and Daungjaiboon, 2022; Wang and Banko, 2021). These models typically operate by either fine-tuning encoder-based Transformers (Sun et al., 2020; Prottasha et al., 2022; González-Carvajal and Garrido-Merchán, 2020; Qasim et al., 2022) or leveraging decoder-based models to generate text outputs directly (Chae and Davidson, 2023; Fields et al., 2024b; Milios et al., 2023). Their success across various text classification tasks has made them indispensable tools in NLP.

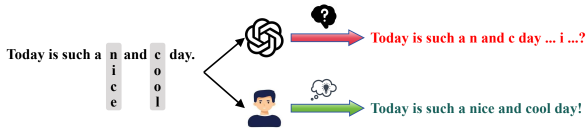


Figure 1: Humans can easily comprehend vertically formatted input by mentally processing the transformation, models often struggle to interpret it.

However, despite these advances, language models remain sensitive to certain types of input variation. Seemingly minor formatting changes, such as line breaks, punctuation marks, or word order, can significantly affect model outputs (Sclar et al., 2024; Wang et al., 2023). A particularly intriguing example of this is vertical text formatting, where words are arranged vertically rather than horizontally. Although this format poses no challenge for human readers, it can confuse language models (shown in Figure 1). Meanwhile, vertical text finds practical applications in areas such as representing tree structures in computing, bypassing social media detection, and even educational games, such as Sudoku word and acrostic poetry. Therefore, understanding how language models interpret vertical text is essential for enhancing their utility.

As the capabilities of large language models (LLMs) such as the GPT series (Radford et al., 2019; Brown et al., 2020; OpenAI et al., 2024) and LLAMA series (Dubey et al., 2024; Touvron et al., 2023) continue to expand, evaluating their ability to understand vertical input becomes increasingly important. Although some of the research on input sensitivity has focused on encoder-based models such as BERT (Devlin et al., 2019; Sanh et al., 2020), the behavior of LLMs when faced with unconventional input formats, such as vertical text, remains underexplored. Given that LLMs are increasingly being applied in critical areas such as content moderation, spam filtering, and misinformation detection, understanding their ability to handle atypical input is crucial to ensure robustness in real-world applications (Rojas-Galeano, 2024; Hu et al., 2024; Wang and Chang, 2022).

To address this gap, we systematically evaluate the impact of vertical text formatting on LLM performance across various text classification tasks. We hypothesize that, despite their advanced capabilities, LLMs still exhibit similar vulnerabilities to vertically formatted input as encoder-based models.

To explore this, we conduct experiments on various open-source and closed-source LLMs across various text classification tasks. Our results show a significant decrease in model accuracy when exposed to vertical text input, indicating that even state-of-the-art LLMs struggle with this input format. We further analyze tokenization and attention matrices to investigate the underlying mechanisms causing this degradation. Additionally, we explore mitigation strategies, finding that few-shot learning can effectively improve the models' ability to handle vertical text, while CoT reasoning does not provide similar benefits.

In summary, our contributions are as follows:

- We present the first comprehensive study on the vulnerability of LLMs to vertically formatted text, demonstrating that this formatting significantly impairs text classification performance.
- Our analysis of tokenization and attention patterns provides insight into the underlying reasons for LLM performance degradation when handling vertical input.
- We propose a few-shot learning with a manually crafted analysis as an effective strategy to mitigate the impact of vertical text formatting, while CoT reasoning proves to be less effective in this context.

2 Related Works

The introduction of the transformer architecture marked the beginning of the LLM era in natural language processing (Vaswani et al., 2017). This evolution started with early models such as GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020) and advanced to more semantically capable and safer models such as ChatGPT, GPT-4, and GPT-4o (OpenAI et al., 2024). These models have been further refined through post-training techniques, including RLHF (Ouyang et al., 2022) and DPO (Rafailov et al., 2024). Throughout their development, concerns and research on the vulnerability and susceptibility of LLMs to various attacks have remained a key focus.

For example, in recent years, researchers have developed various methods to "jailbreak" LLMs, causing them to generate harmful content (Dong et al., 2024; Perez and Ribeiro, 2022; Chao et al., 2024; Deng et al., 2024). Such attacks clearly highlight the inherent vulnerabilities of LLMs. However, these vulnerabilities extend beyond this issue. Studies have shown that LLMs can produce significantly different outputs with similar input texts depending on content order (e.g., in multiple choice questions) (Wang et al., 2023). Moreover, LLMs often show pronounced sensitivity to basic punctuation and line breaks, further illustrating their fragility (Sclar et al., 2024).

Recent research reveals that vertically formatted text can severely impair the comprehension abilities of Transformer encoder-based models, such as BERT, causing significant performance degradation (Rusert, 2024; Devlin et al., 2019). This vulnerability, where straightforward text can substantially deceive language models, raises concerns about the potential impact of such formatting on contemporary LLMs. If LLMs are similarly susceptible to vertically formatted text, this could pose a threat to real-world applications.

3 Methodology

To assess the vulnerability of LLMs to vertically formatted text, we will select a set of semantically significant words from a given text and input them vertically into the LLMs. The remaining words will be presented in the standard horizontal format from left to right. This approach mirrors real-world usage, where most text is input normally to ensure coherence and readability, while the vertically formatted keywords test the model's susceptibil-

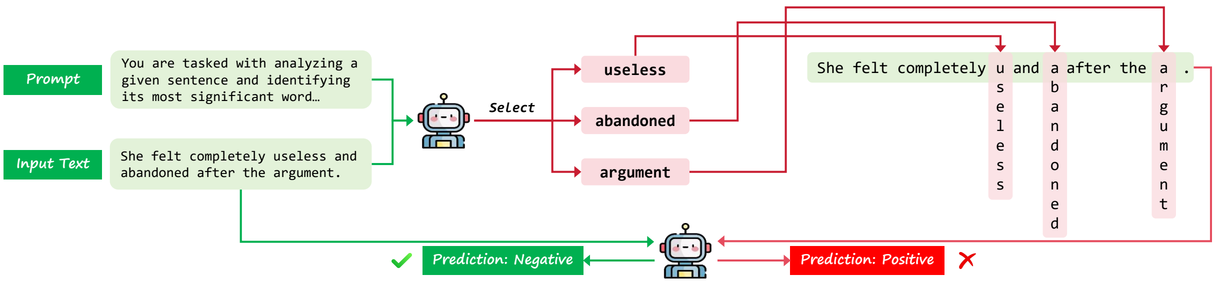


Figure 2: An example of the entire algorithm process demonstrates that vertically inputted words lead the LLM to incorrectly predict the label for text classification.

ity. The method comprises two main components: *Word Selection* and *Word Transformation*, as shown in Figure 2.

3.1 Word Selection

In previous work, researchers used a greedy approach to assess the impact of each word on the prediction probability of the model, thus determining the importance of individual words in a sentence (Rusert, 2024). However, this method is time-consuming and resource-intensive for LLMs. In this paper, we address this issue by leveraging prompt-based LLMs as evaluators to identify and extract key words from the text.

3.2 Word Transformation

The goal of word transformation is to vertically arrange specific words within a sentence. The workflow is as follows:

Decompose the Sentence. The input sentence is divided into a list of words, and the height of the vertical arrangement is determined by the longest designated word.

Initialize the Grid. We create a two-dimensional grid based on the length of the longest word, initially filled with empty spaces to hold the characters of the vertical words.

Align Vertical Characters. Each vertical word's characters are placed in corresponding grid rows, with shorter words padded to maintain alignment.

Handle Non-Vertical Words. Characters of non-vertical words are placed in their respective rows, with remaining spaces filled to ensure formatting consistency.

Generate the Final String. Each grid row is concatenated into a string, separated by newlines and adjusted for alignment, resulting in a layout that displays vertical and horizontal words as intended.

4 Experiments

4.1 Datasets

We propose that vertically formatted text may disrupt LLMs' comprehension of content, potentially posing a threat by circumventing their detection mechanisms. In this paper, we therefore select five text classification datasets, covering a range of applications including sentiment analysis, harmful statement detection, and more:

- SST-2 (Socher et al., 2013): A benchmark dataset for sentiment classification, featuring labeled movie reviews.
- CoLA (Warstadt et al., 2019): A dataset for evaluating linguistic acceptability, containing English sentences labeled as grammatically acceptable or unacceptable.
- QNLI (Wang et al., 2019): Derived from the Stanford Question Answering Dataset (SQuAD), which evaluates whether a sentence answers a specific question.
- Rotten Tomatoes (Pang and Lee, 2005): A dataset of movie reviews used for sentiment analysis, with each review classified as positive or negative.
- Jigsaw Toxicity[1] (cjadams et al., 2017): A dataset of online comments labeled for various types of toxic behavior.

4.2 Large Language Models

To more effectively assess the vulnerability of latest LLMs to vertically formatted input text, we select various well-known models for experiments.

Closed-Source Models. We utilize four different models developed by OpenAI, including the latest powerful models GPT-4O and GPT-4O-MINI.

[1] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

Model	SST-2		CoLA ♠		QNLI		Rotten Tomatoes		Jigsaw Toxicity	
	Original	Vertical	Original	Vertical	Original	Vertical	Original	Vertical	Original	Vertical
Closed-Source Models										
GPT-3.5	93.00	65.00 (↓ **28.00**)	80.00	47.00 (↓ **33.00**)	85.00	69.00 (↓ **16.00**)	92.00	57.00 (↓ **35.00**)	85.00	62.00 (↓ **23.00**)
GPT-4	96.00	67.00 (↓ **29.00**)	90.00	49.00 (↓ **41.00**)	89.00	71.00 (↓ **18.00**)	93.00	64.00 (↓ **29.00**)	89.00	58.00 (↓ **31.00**)
GPT-4o-mini	95.00	66.00 (↓ **29.00**)	89.00	50.00 (↓ **39.00**)	90.00	71.00 (↓ **19.00**)	91.00	61.00 (↓ **30.00**)	85.00	57.00 (↓ **28.00**)
GPT-4o	95.00	68.00 (↓ **27.00**)	87.00	47.00 (↓ **40.00**)	90.00	70.00 (↓ **20.00**)	90.00	65.00 (↓ **25.00**)	91.00	60.00 (↓ **31.00**)
Open-Source Models										
Llama3-8B	89.00	61.00 (↓ **28.00**)	75.00	50.00 (↓ **25.00**)	83.00	62.00 (↓ **21.00**)	86.00	42.00 (↓ **44.00**)	88.00	58.00 (↓ **30.00**)
Llama3-70B	96.00	67.00 (↓ **29.00**)	85.00	48.00 (↓ **37.00**)	86.00	63.00 (↓ **23.00**)	91.00	46.00 (↓ **45.00**)	88.00	58.00 (↓ **30.00**)
Llama3.1-8B	93.00	51.00 (↓ **42.00**)	80.00	49.00 (↓ **31.00**)	83.00	59.00 (↓ **24.00**)	89.00	54.00 (↓ **45.00**)	80.00	64.00 (↓ **16.00**)
Llama3.1-70B	96.00	66.00 (↓ **30.00**)	84.00	50.00 (↓ **34.00**)	84.00	66.00 (↓ **18.00**)	92.00	63.00 (↓ **29.00**)	87.00	62.00 (↓ **25.00**)
Gemma2-9B	88.00	60.00 (↓ **28.00**)	83.00	51.00 (↓ **32.00**)	78.00	60.00 (↓ **18.00**)	86.00	58.00 (↓ **28.00**)	85.00	53.00 (↓ **32.00**)
Gemma2-27B	94.00	58.00 (↓ **36.00**)	87.00	50.00 (↓ **37.00**)	83.00	64.00 (↓ **19.00**)	89.00	54.00 (↓ **35.00**)	88.00	55.00 (↓ **33.00**)
Qwen1.5-72B	95.00	63.00 (↓ **32.00**)	81.00	52.00 (↓ **29.00**)	85.00	66.00 (↓ **19.00**)	93.00	57.00 (↓ **36.00**)	88.00	63.00 (↓ **25.00**)
Qwen2-72B	96.00	60.00 (↓ **36.00**)	84.00	50.00 (↓ **34.00**)	88.00	62.00 (↓ **26.00**)	93.00	59.00 (↓ **34.00**)	91.00	59.00 (↓ **32.00**)

Table 1: Accuracy scores of different LLMs on five datasets using zero-shot prediction, with changes before and after shown in parentheses (indicating the decrease). Dataset marked with ♠ includes two words inputted in a vertical format, while the others contain four words vertically inputted.

Open-Source Models. We conduct our experiments on four models from the Llama series, two models from the Gemma series, and two models from the Qwen series, with parameter sizes ranging from 8 billion to 72 billion (Dubey et al., 2024; Team et al., 2024; Yang et al., 2024).

4.3 Metric

We use the most straightforward metric: **Accuracy**. We evaluated the model by comparing its predicted labels with the actual labels. The accuracy of a model is calculated as follows:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{I}(y_i = \hat{y}_i) \qquad (1)$$

where $\mathrm{I}(\cdot)$ is an indicator function that is 1 if the condition is true and 0 otherwise, y_i represents the true label, $\hat{y}_i$ is the predicted label, and N is the total number of samples.

4.4 Experiment Results

We utilize four GPT series LLMs along with eight widely recognized open-source LLMs on five text classification datasets. For the SST-2, QNLI, Rotten Tomatoes, and Jigsaw Toxicity datasets, the GPT-4o-mini model would select four words for vertical input. In contrast, for the CoLA dataset, it selects two words.

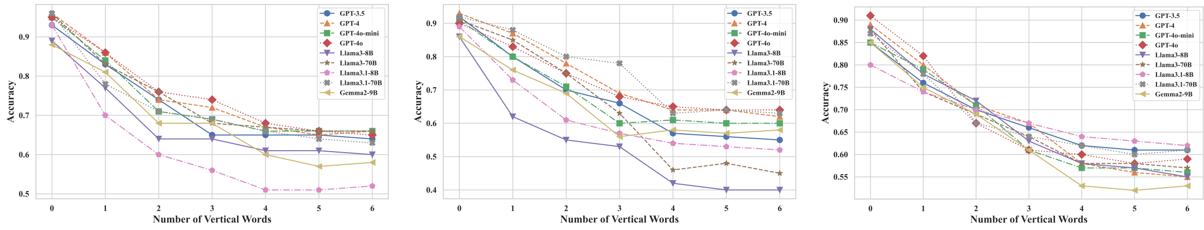
In the label prediction phase, we utilize straightforward and precise prompts for zero-shot prediction, enabling the model to generate predicted labels. We compare the classification accuracy before and after modifying the input format, with the results presented in Table 1. The detailed implementation of different LLMs is shown in Appendix B.

Vulnerability of LLMs to Vertical Input. Based on the experimental results in Table 1, it is observed that inputting a few key words from the text in a vertical format into LLM significantly disrupts their ability to perform text classification tasks.

Specifically, for the CoLA dataset, the model's classification accuracy drops by nearly 40 percentage points when two key words are input vertically. Given that the random prediction accuracy for a binary classification problem is 50%, this indicates that the model essentially loses its classification ability.

The model accuracy decreases the least on the QNLI dataset, likely due to the nature of the task. Relationships between sentences can be expressed in various ways, allowing LLMs to identify connections in the remaining content even when a few words are presented in a vertical format.

For the other three datasets, the decrease in accuracy after implementing vertical input for different models mostly ranges from 25 to 40 percentage points, which is a significant drop and almost indicates that current LLMs nearly lose their natural language understanding ability when encountering



(a) SST-2 (b) Rotten Tomatoes (c) Jigsaw Toxicity

Figure 3: The relationship between the number of vertically inputted words and the accuracy of various LLMs on text classification tasks.

Vertical

		Actual +	Actual -
Pred.	+	14	2
Pred.	-	36	48

Original

		Actual +	Actual -
Pred.	+	43	4
Pred.	-	7	46

Table 2: Comparison of confusion matrices for the Jigsaw Toxicity dataset. (+) indicates toxic, while (-) indicates non-toxic.

Vertical

		Actual +	Actual -
Pred.	+	39	41
Pred.	-	7	13

Original

		Actual +	Actual -
Pred.	+	43	5
Pred.	-	3	49

Table 3: Comparison of confusion matrices for the SST-2 dataset. (+) indicates positive and (-) indicates negative.

vertical input, even when the manipulation involves only a few key words.

Potential Threat of Mislabelling. According to Table 1, vertically formatted input text could reduce the accuracy of LLMs on sentiment and toxicity classification tasks by 30 to 40 percentage points. A more concerning issue emerges when examining the confusion matrix in Table 2 and Table 3: On the SST-2 dataset, inputting some words in a vertical format to the large language model reduces the classification accuracy for negative sentences from 91% to 24%. Similarly, on the Jigsaw Toxicity dataset, this approach decreases the model's accuracy for classifying harmful text from 86% to 28%.

This indicates that the models largely lose their ability to identify harmful content when key words are input vertically. In contrast, humans easily understand such text. Therefore, this vulnerability can have a severe negative impact on tasks like harmful content monitoring.

Open-Source via Closed-Source. In the upper part of Table 1, we evaluate four GPT series models, considered some of the most powerful LLMs today. Regardless of model size and release date, they exhibit significant confusion when encountering vertically formatted text inputs.

To ensure comprehensive experiments, we also test three high-performing open-source LLM series: LLAMA, GEMMA, and QWEN. As shown in the lower part of Table 1, these open-source models also display considerable vulnerability to vertically formatted text inputs. Moreover, on four experimental datasets, LLAMA3.1-8B shows a greater accuracy drop than LLAMA3-8B, despite being a stronger language model. This suggests that current LLMs are not trained with consideration for the impact of vertical text inputs, resulting in a high degree of vulnerability, which may pose certain risks in some areas.

Number of Vertical Words via Accuracy. In Figure 3, we choose 9 different LLMs to conduct a detailed experiment on the relationship between the number of vertically formatted words and the model's classification accuracy.

Initially, as the number of vertically input words increases, the classification accuracy of the LLMs consistently declines. This happens because words relevant to the classification are gradually input in a vertical format, confusing the model's ability to distinguish them. When the number of vertically formatted words reaches a certain point, the model's accuracy stabilizes at a fixed value. This occurs because most of the key words affecting the classification are presented in a vertical format, and the remaining words have little impact on the model's predictions.

This experiment shows that the LLMs' classification heavily relies on the recognition of relevant words, further explaining its vulnerability to vertically formatted text. In contrast, identifying keywords related to sentiment or toxicity from a text is relatively simple for humans, highlighting the significant threat this vulnerability poses to LLMs.

Model	SST-2	CoLA	QNLI	Rotten Tomatoes	Jigsaw Toxicity
Closed-Source Models					
GPT-3.5 w/ CoT	61.00 (↓ **4.00**)	50.00 (↑ **3.00**)	59.00 (↓ **10.00**)	53.00 (↓ **4.00**)	62.00 (**0.00**)
GPT-4 w/ CoT	66.00 (↓ **1.00**)	51.00 (↑ **2.00**)	68.00 (↓ **3.00**)	60.00 (↓ **4.00**)	56.00 (↓ **2.00**)
GPT-4o-mini w/ CoT	68.00 (↑ **2.00**)	50.00 (**0.00**)	76.00 (↑ **1.00**)	64.00 (↑ **3.00**)	56.00 (↑ **3.00**)
GPT-4 w/ CoT	71.00 (↑ **3.00**)	52.00 (↑ **5.00**)	74.00 (↑ **4.00**)	66.00 (↑ **1.00**)	66.00 (↑ **6.00**)
Open-Source Models					
Llama3-8B w/ CoT	59.00 (↓ **2.00**)	49.00 (↓ **1.00**)	64.00 (↑ **2.00**)	53.00 (↑ **11.00**)	55.00 (↓ **3.00**)
Llama3-70B w/ CoT	66.00 (↓ **1.00**)	46.00 (↓ **2.00**)	59.00 (↓ **4.00**)	55.00 (↑ **9.00**)	62.00 (↑ **4.00**)
Llama3.1-8B w/ CoT	53.00 (↑ **2.00**)	51.00 (↑ **2.00**)	62.00 (↑ **3.00**)	56.00 (↑ **2.00**)	63.00 (↓ **1.00**)
Llama3.1-70B w/ CoT	68.00 (↑ **2.00**)	47.00 (↓ **3.00**)	65.00 (↓ **1.00**)	62.00 (↓ **1.00**)	65.00 (↑ **3.00**)
Gemma2-9B w/ CoT	60.00 (**0.00**)	49.00 (↓ **2.00**)	64.00 (↑ **4.00**)	61.00 (↑ **3.00**)	52.00 (↓ **1.00**)
Gemma2-27B w/ CoT	61.00 (↑ **3.00**)	51.00 (↑ **1.00**)	64.00 (**0.00**)	57.00 (↑ **3.00**)	53.00 (↓ **2.00**)
Qwen1.5-72B w/ CoT	63.00 (**0.00**)	51.00 (↓ **1.00**)	63.00 (↓ **3.00**)	59.00 (↑ **2.00**)	60.00 (↓ **3.00**)
Qwen2-72B w/ CoT	62.00 (↑ **2.00**)	52.00 (↑ **2.00**)	64.00 (↑ **2.00**)	60.00 (↑ **1.00**)	61.00 (↑ **2.00**)

Table 4: Accuracy scores after applying CoT, with parentheses indicating the change in scores (compared to the non-CoT scores in Table 1). Dark green indicates an increase in accuracy, red denotes a decline, and black means no change. All other experimental settings remain consistent with those in Table 1.

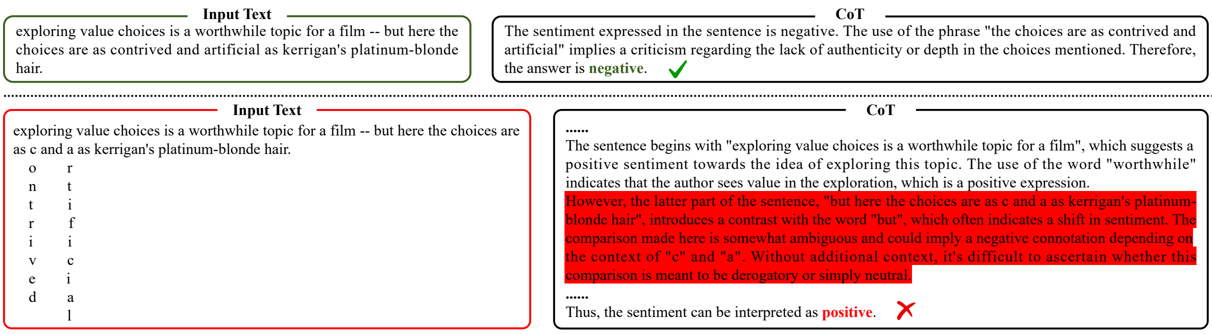


Figure 4: An example where Chain of Thought (CoT) reasoning is used during GPT-4 inference, yet the text with some vertically input words is misclassified.

5 Potential Mitigation Methods

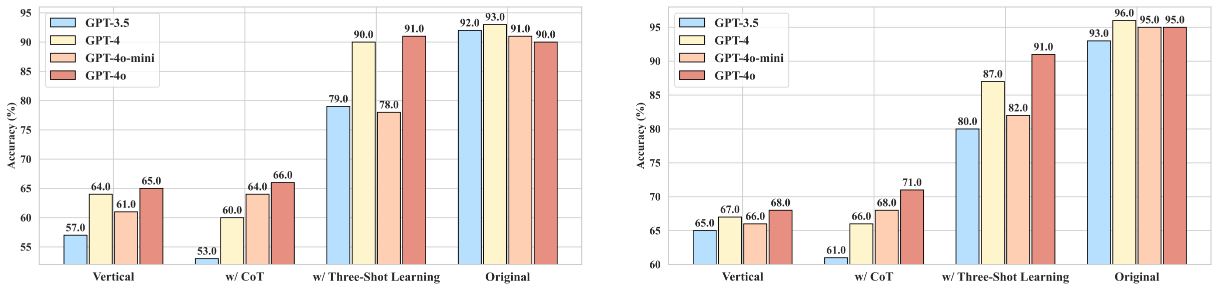
5.1 Failure of CoT

In the above experiments, we have the LLMs directly output classification results for two types of text inputs. This method somewhat bypasses the model's reasoning process, which may result in inaccuracies. To address this, we consider using a Chain of Thought (CoT) approach (Wei et al., 2023) to determine if prompting the model to output reasoning steps can mitigate its vulnerability to vertically formatted text inputs. We incorporate "Think step by step" into the prompt to encourage the LLMs to provide a step-by-step reasoning process, aiming to help the model better understand the intricacies of vertically formatted text inputs.

However, the experimental results in Table 4 show that even with the use of the CoT, the LLMs still fail to recognize that some words are input in a vertical format, despite generating a corresponding reasoning process. We can observe that after using CoT with LLMs on different classification tasks, the classification accuracy either increases or decreases compared to not using it, which differs significantly from our expectations. Even when there is an improvement, the increase in accuracy is minimal, generally less than or equal to 3 percentage points. On the Rotten Tomatoes dataset, although the two models from the Llama3 series show a noticeable improvement of around 10 percentage points after using CoT, their final classification accuracy remains extremely low, only about 55%. Unexpectedly, powerful models like GPT-3.5 and GPT-4 show reduced accuracy across four datasets despite using CoT.

Figure 4 illustrates an instance where the powerful GPT-4 model produces an incorrect answer despite utilizing the CoT method for reasoning. The model fails to associate the vertically formatted characters of each word, leading to guesswork and uncertainty, which results in the incorrect answer.

In summary, comparing the accuracy between



(a) *Rotten Tomatoes* (b) *SST-2*

Figure 5: Comparison of accuracy scores between three-shot learning and three other methods.

GPT-4 Few-Shot Generation

To determine the sentiment of the sentence, let's first reconstruct it by identifying the vertically inputted words:

1."overburdened" (o, v, e, r, b, u, r, d, e, n, e, d)

2."complicated" (c, o, m, p, l, i, c, a, t, e, d)

3."banal" (b, a, n, a, l)

Reconstructing the vertical part of the sentence: "overburdened with complicated plotting and banal dialogue".

Now, let's analyze the sentiment of the reconstructed sentence:

(1) "overburdened" suggests something is excessively loaded or stressed. (2) "complicated" implies something is overly complex. (3) "banal" means something is unoriginal or boring.

Therefore, the overall sentiment is **negative.** ✓

Figure 6: An example of few-shot learning output from the GPT-4 model, in which the model identifies vertical words to arrive at the correct answer.

the two types of inferences reveals no significant difference. This suggests that the chain of thought does not help the model reduce the interference caused by vertically formatted text inputs.

5.2 Effective Few-Shot Learning

Our findings indicate that employing CoT does not effectively help LLMs identify words presented in vertical format, resulting in incorrect answers. We believe this limitation stems from the LLM's lack of awareness regarding such text formats.

To address this, we propose using a few-shot learning approach to enhance the model's ability to recognize and respond to these special inputs. For each example, we meticulously construct a detailed analysis to facilitate the model's learning of relevant knowledge and assist in reconstructing the original sentence.

We employ a three-shot learning approach, selecting four models from the GPT series for testing and validation across two datasets. As shown in Figure 5, the experimental results indicate a significant improvement in the model's accuracy after applying the three-shot learning method, especially with the GPT-4 and GPT-4O models, which achieve prediction accuracy comparable to that of the original normal input text. This underscores the strong understanding and reasoning capabilities of these models. While the performance of GPT-3.5 and GPT-4O-MINI is lower than that of the first two models, they still demonstrate a noteworthy increase in accuracy. In Figure 6, the GPT-4 model, using a three-shot learning method, accurately detects and reconstructs vertically formatted words from the input, generating correct classification outputs. This highlights the model's enhanced performance with few-shot learning, particularly when compared to the CoT output in Figure 4.

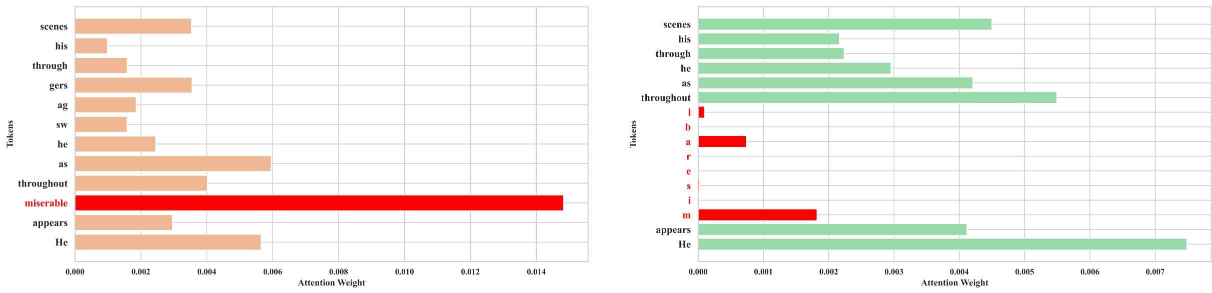
6 Cause Analysis

6.1 Disordered Tokens

Due to the natural left-to-right and top-to-bottom writing order of text, the tokenizer used by LLMs encodes input text in this sequence. For instance, the tokenizer for the LLAMA3.1-8B model represents the word "vertical" as a single token when input horizontally. However, when input vertically, it converts into a token sequence of length 15 due to the spaces and line breaks. This causes the LLM to lose its understanding of the complete word.

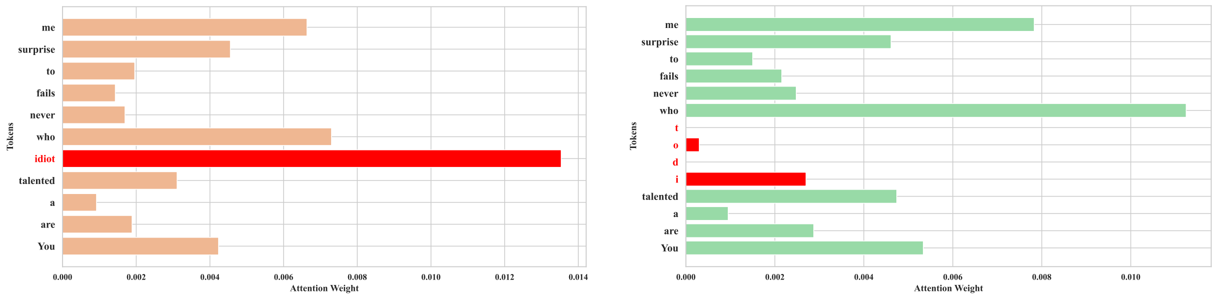
6.2 Lost Attention

The most important part of LLMs is the attention matrix, which shows the degree of association between tokens within the model. We analyze the fundamental reason for the decrease in text classification ability of LLMs caused by vertical input



(a) *Original* (b) *Vertical*

Figure 7: An example of comparing the attention weights of each token towards the "negative" token before and after vertical input of the word. The red highlights indicate the vertical word and its constituent tokens.



(a) *Original* (b) *Vertical*

Figure 8: An example of comparing the attention weights of each token towards the "toxic" token before and after vertical input of the word. The red highlights indicate the vertical word and its constituent tokens.

text by plotting the changes in attention weights for corresponding words in the attention matrices of the LLAMA3.1-8B model, both before and after some words are input in a vertical format.

In Figure 7, we use text from sentiment classification as an example, the original text is “He appears miserable throughout as he swaggers through his scenes”. We observe that when words are input into the large language model in a horizontal format, the model assigns strong attention weights to the word "miserable", which signifies a negative sentiment, in relation to "negative". This indicates that the model recognizes "miserable" as conveying a negative emotion. However, when the word "miserable" is split and input in a vertical format, its components fail to establish a strong connection with "negative", ultimately leading to incorrect classification predictions by the model.

Similarly, in Figure 8, the original toxic text is "You are a talented idiot who never fails to surprise me", when we input the harmful word "idiot" normally, the large language model connects "toxic" with "idiot" and classifies the text as harmful. However, when we input "idiot" in a vertical format, the model loses its ability to establish a strong connection between "toxic" and the new tokens ("i", "d", "o" and "t") that make up "idiot", leading it to conclude that the input is not harmful.

In summary, the tokenization conventions of LLMs and the lack of relevant pre-training data have impaired their ability to understand vertical text. In contrast, people can easily comprehend words presented in a vertical format. Consequently, the cognitive disparity between humans and LLMs leads to the models’ vulnerability to vertical text input, which could pose potential threats that humans might exploit.

7 Conclusion

In this paper, we validate the vulnerability of modern LLMs to vertically formatted inputs, testing various mainstream models to reveal defects and flaws specific to this input type, which could pose real-world threats. Moreover, we find that CoT reasoning does not aid LLMs in resolving this issue, while few-shot learning with provided analysis could help mitigate it. We also note that this limitation arises from the nature of their pre-training data and tokenization mechanisms. In the future, we aim to explore more effective strategies for addressing this issue in LLMs through either pre-training or fine-tuning.

Limitations

While we investigate the vulnerability of LLMs to vertically formatted input, this paper has several limitations: (i) While our findings suggest that few-shot learning does help mitigate this vulnerability, it necessitates the design of demonstrations for each specific task. We do not explore the potential of fine-tuning LLMs to address this issue due to limitations in available datasets and GPU resources. Future research could investigate whether fine-tuning techniques could improve model robustness against vertically formatted input. (ii) We do not assess the impact of vertically formatted input on text generation tasks. Future studies could investigate this aspect to evaluate any potential negative effects of such formatting on the generation performance of LLMs.

Ethics Statement

Ethical considerations are of utmost importance in our research endeavors. In this paper, we strictly adhere to ethical principles by exclusively utilizing open-source datasets and employing various models that are either open-source or widely recognized in the scientific community. Our findings highlight the text format vulnerabilities in large language models. We are committed to upholding ethical standards throughout the research process, prioritizing transparency, and promoting the responsible use of technology for the betterment of society. Additionally, we include a toxic example to highlight the potential severity of these vulnerabilities To minimize negative impacts, we explore and provide a method for mitigation.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, et al. 2020. Language models are few-shot learners.

Youngjin Chae and Thomas Davidson. 2023. Large language models for text classification: From zero-shot learning to fine-tuning. *Open Science Foundation*.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. 2024. Jailbreaking black box large language models in twenty queries.

cjadams, Jeffrey Sorensen, Julia Elliott, Lucas Dixon, Mark McDonald, nithum, and Will Cukierski. 2017. Toxic comment classification challenge.

Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. 2024. Masterkey: Automated jailbreaking of large language model chatbots. In *Proceedings 2024 Network and Distributed System Security Symposium*, NDSS 2024. Internet Society.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Zhichen Dong, Zhanhui Zhou, Chao Yang, Jing Shao, and Yu Qiao. 2024. Attacks, defenses and evaluations for llm conversation safety: A survey.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, et al. 2024. The llama 3 herd of models.

John Fields, Kevin Chovanec, and Praveen Madiraju. 2024a. A survey of text classification with transformers: How wide? how large? how long? how accurate? how expensive? how safe? *IEEE Access*, 12:6518–6531.

John Fields, Kevin Chovanec, and Praveen Madiraju. 2024b. A survey of text classification with transformers: How wide? how large? how long? how accurate? how expensive? how safe? *IEEE Access*.

Santiago González-Carvajal and Eduardo C Garrido-Merchán. 2020. Comparing bert against traditional machine learning text classification. *arXiv preprint arXiv:2005.13012*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification.

Zhanhao Hu, Julien Piet, Geng Zhao, Jiantao Jiao, and David Wagner. 2024. Toxicity detection for free.

Aristides Milios, Siva Reddy, and Dzmitry Bahdanau. 2023. In-context learning for text classification with many labels. In *Proceedings of the 1st GenBench Workshop on (Benchmarking) Generalisation in NLP*, pages 173–184, Singapore. Association for Computational Linguistics.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning based text classification: A comprehensive review.

Marcin Michał Mirończuk and Jarosław Protasiewicz. 2018. A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, 106:36–54.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, et al. 2024. Gpt-4 technical report.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Fábio Perez and Ian Ribeiro. 2022. Ignore previous prompt: Attack techniques for language models.

Nusrat Jahan Prottasha, Abdullah As Sami, Md Kowsher, Saydul Akbar Murad, Anupam Kumar Bairagi, Mehedi Masud, and Mohammed Baz. 2022. Transfer learning for sentiment analysis using bert based supervised fine-tuning. *Sensors*, 22(11):4157.

Rukhma Qasim, Waqas Haider Bangyal, Mohammed A Alqarni, and Abdulwahab Ali Almazroi. 2022. A fine-tuned bert-based transfer learning approach for text classification. *Journal of healthcare engineering*, 2022(1):3498123.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI*. Accessed: 2024-11-15.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model.

Ben Rodrawangpai and Witawat Daungjaiboon. 2022. Improving text classification with transformers and layer normalization. *Machine Learning with Applications*, 10:100403.

Sergio Rojas-Galeano. 2024. Zero-shot spam email classification using pre-trained large language models. *arXiv preprint arXiv:2405.15936*.

Jonathan Rusert. 2024. Vertattack: Taking advantage of text classifiers' horizontal vision. In *North American Chapter of the Association for Computational Linguistics*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to fine-tune bert for text classification?

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, et al. 2024. Gemma 2: Improving open language models at a practical size.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, et al. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding.

Cindy Wang and Michele Banko. 2021. Practical transformer-based multilingual text classification. In *North American Chapter of the Association for Computational Linguistics*.

Yau-Shian Wang and Yingshan Chang. 2022. Toxicity detection with generative prompt-based inference.

Yiwei Wang, Yujun Cai, Muhao Chen, Yuxuan Liang, and Bryan Hooi. 2023. Primacy effect of ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 108–115, Singapore. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, et al. 2024. Qwen2 technical report.

Wenpeng Yin, Jamaal Hay, and Dan Roth. 2019. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach.

A Large Language Models

- GPT-3.5: A robust large language model developed by OpenAI, capable of generating text based on instructions, and highly effective across diverse natural language processing tasks.
- GPT-4: An advanced multi-modal language model from OpenAI that accepts both image and text inputs for text generation, achieving near-human performance on various benchmarks.
- GPT-4O-MINI: A cost-efficient multimodal model, released by OpenAI on July 18, 2024, is a distilled version of GPT-4O, offering low latency and cost while supporting a wide range of tasks.
- GPT-4O: OpenAI's latest multimodal AI model, offering enhanced reasoning, generation, and understanding capabilities across text, image, and speech with faster and more efficient responses.
- LLAMA: Meta's open-weight AI models, designed for improved efficiency, reasoning, and multilingual capabilities, offering enhanced performance for various AI applications.
- GEMMA2: The next-generation open-source model from Google, released on June 27, 2024, as an improved version of GEMMA, available in 2B, 9B, and 27B parameter configurations.
- QWEN: The series of models developed by Alibaba are powerful open-source AI models with strong multilingual understanding, containing different versions with varying parameters.

B Implementation Details

For the GPT series models, we utilize the OpenAI API[2] for model invocation. During the word selection phase, we set *top_p* to 1.0 and *temperature* to 0.0 to ensure consistent word selection. In the text classification phase, we adjust *top_p* to 0.95 and keep *temperature* at 0.0 to maintain the reliability of the model's output.

For other open-source LLMs, we use either Hugging Face weights or the official API for model inference, applying the same parameter settings as above.

For each dataset, we randomly select 100 test samples for experimentation. To ensure fairness, the number of samples for each label is evenly distributed in the selected test data, particularly for unbalanced datasets.

[2]https://openai.com/

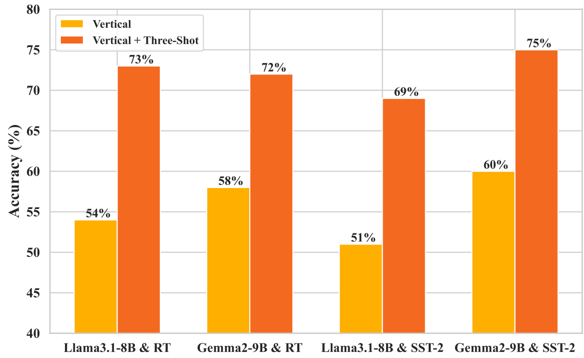


Figure 9: The accuracy comparison of two open-source LLMs after applying the three-shot learning method to interpret text input. The x-axis represents the model names and experimental datasets.

C Few-Shot with Open-Source LLMs

To further evaluate the effectiveness of the few-shot learning method in improving LLMs' ability to reconstruct vertical input content, we conduct experiments on two well-known open-source LLMs using two experimental datasets. The results are presented in Figure 9.

Our findings show that applying three-shot learning significantly enhances model accuracy. GEMMA2-9B achieves an improvement of approximately 19 percentage points across both datasets, while LLAMA3.1-8B gains around 15 percentage points. Although these improvements are smaller than those observed in the GPT models shown in Figure 5, they still represent substantial gains, considering the base models' initial reasoning and natural language capabilities.

D Does Explicit Prompt Work?

In real-world scenarios, natural system prompts do not explicitly instruct large language models to recognize and reconstruct vertical content in the input before generating the final output. To address this, we aim to examine how explicitly instructing the model to consider vertical content impacts its output.

In this section, we conduct an experiment by informing LLMs of the presence of vertical content

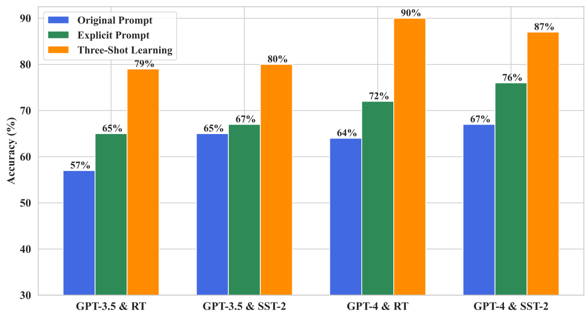


Figure 10: The accuracy comparison of two GPT models under three different prompt settings, with the accuracy after using an explicit prompt shown in green. The x-axis represents the model names and experimental datasets.

Explicit Prompt

I want you to determine whether the sentiment of a sentence is positive or negative. However, some of the words in the sentence are inputted vertically into the model. I expect you to identify these words, reconstruct the original sentence, and provide the correct sentiment classification.

Figure 11: An example of an explicit instruction prompt for the sentiment classification task with vertical words input.

in the input. Figure 11 shows an example of an explicit instruction for the sentiment classification task.

We conduct the experiment using two GPT models: GPT-3.5 and GPT-4. We compare the performance of explicit prompts with naive and few-shot prompts to investigate the actual impact of explicit prompts in helping LLMs recognize vertical input. The results are shown in Figure 10.

The findings indicate that although explicit prompts help LLMs mitigate the negative impact caused by vertical input to some extent, the improvement is quite small compared to the few-shot prompt. On the SST-2 dataset, GPT-3.5 improves its accuracy score by only 2 percentage points with an explicit prompt, whereas it gains 15 percentage points with a few-shot prompt. Similarly, for the GPT-4 model on the Rotten Tomatoes dataset, the improvement is only 8 percentage points with an explicit prompt, compared to 26 with a few-shot prompt.

On the one hand, although explicit prompts improve model accuracy, the improvements are not substantial. On the other hand, this type of prompt is not widely applicable to real-world scenarios, as LLMs are exposed to numerous attacks and vulnerabilities.

Type of Experiment	**Dataset**	**Accuracy(%)**
Original	*SST-2*	96.67
	QNLI	92.00
	Rotten Tomatoes	97.33
	Jigsaw Toxicity	95.00
Vertical	*SST-2*	96.33
	QNLI	92.00
	Rotten Tomatoes	97.33
	Jigsaw Toxicity	95.33

Table 5: The average accuracy of three human evaluations across four datasets, comparing results before and after vertical word input.

E Human vs. LLMs

To better understand the vulnerabilities of LLMs to vertical input compared to the human brain, we recruit three graduate-level students who are native English speakers to conduct experiments on four different datasets. Each student is presented with choices and required to select one, repeating this process twice—once before and once after the vertical input. Each time, the texts are shuffled to prevent memorization.

The experimental results of human classification are presented in Table 5, with accuracy computed as the average performance of the three students for each task. The results indicate that vertical input poses no challenge to human cognition, as evidenced by a minimal accuracy difference of just 0.33% across the two datasets — likely due to normal randomness rather than the input format. This demonstrates that while vertical input is easily comprehensible for humans, it remains a difficult problem for current LLMs, highlighting both the potential risks associated with relevant applications and the gap in format understanding between humans and LLMs.