Article

Comparative Analysis of Prompt Strategies for LLMs: Single-Task vs. Multitasking Prompts

Manuel Gozzi * and Federico Di Maio

Posted Date: 17 October 2024

doi: 10.20944/preprints202410.1334.v1

Keywords: artificial intelligence; large language model; prompt engineering

Article

Comparative Analysis of Prompt Strategies for LLMs: Single-Task vs. Multitasking Prompts

Gozzi Manuel [1,*] and Di Maio Federico [2]

[1] Department of Engineering Sciences, Guglielmo Marconi University, 00193 Roma, Italy
[2] Independent researcher
* Correspondence: m.gozzi@studenti.unimarconi.it

Abstract: This study examines the impact of prompt engineering on large language models (LLMs), focusing on a comparison between multitasking and single-task prompts. Specifically, we explore whether a single prompt handling multiple tasks — such as Named Entity Recognition (NER), sentiment analysis, and JSON output formatting — can achieve similar efficiency and accuracy to dedicated single-task prompts. The evaluation uses a combination of performance metrics to provide a comprehensive analysis of output quality. Experiments were conducted using a selection of open-source LLMs, including LLama3.1 8B, Qwen2 7B, Mistral 7B, Phi3 Medium, and Gemma2 9B. Results show that single-task prompts do not consistently outperform multitasking prompts, highlighting the significant influence of the model's data and architecture on performance.

Keywords: artificial intelligence, large language model, prompt engineering

1. Introduction

Large Language Models (LLMs) are increasingly integrated into everyday life, making it crucial to understand and learn the correct interaction methods to effectively interface with these tools. The countless parameters that define LLMs are sufficient to emulate natural conversations. However, their functionality fundamentally relies on predicting the next token based on the given input. For this reason, mastering best practices in prompting is essential, as even minor variations in a prompt can significantly affect the outcome. In light of this, we have undertaken a research study aimed at quantifying the performance of multi-task prompts in comparison to single-task prompts. Our initial hypothesis posits that single-task prompts, due to their inherently lower complexity, should yield better results than their multi-task counterparts. While this assumption may seem intuitive, quantifying these differences and conducting a comparative analysis across various models is essential for understanding the performance degradation in prompting. Furthermore, despite the intuition that simpler, single-task prompts should produce better results, empirical data suggest that this is not universally true for all LLMs. Indeed, while certain models support our initial hypothesis, others exhibit superior performance with multitask prompts. This observation highlights the necessity of a thorough analysis to understand the underlying dynamics of each model. The differences, particularly those seen in models such as LLama 3.1 and Mistral, demonstrate that the interaction between a prompt and a model cannot be reduced to a simple general rule. This makes our study critical for optimizing the use of LLMs in various pipelines.

The objective of this study is to systematically and quantitatively analyze the performance differences between single-task and multitask prompts across a variety of large-scale language models. The goal is to provide empirical data that can guide developers and users in selecting the most effective prompting strategy for each model and use case. Our methodology includes defining a series of standardized tasks, which will be presented to various LLMs in both single-task and multitask formats. Performance will be measured using a composite metric that combines the results of three metrics applied to specific NLP tasks: F1 score for Named Entity Recognition (NER), exact match for sentiment analysis, and BLEU for review coherence.

The findings from this research could have significant practical implications. Firstly, they will provide data-driven guidelines to optimize interaction with LLMs in various application contexts.

Additionally, any discrepancies observed among the models may offer valuable insights into the architectural and training differences that influence how LLMs respond to prompt complexity. One of the most well-known challenges of LLMs is the interpretability of their output. This study highlights this issue by revealing "non-standard" results when using what would be considered a standard approach.

2. Related Work

Recent research has explored various prompt engineering techniques for enhancing large language models' (LLMs) performance in natural language processing (NLP) tasks. Studies have investigated different types of prompts, including discrete, continuous, few-shot, and zero-shot approaches[1]. Discrete prompts are formulated using natural language, making them interpretable and easier to design, whereas continuous prompts leverage embeddings that are optimized through gradient-based methods, offering more flexibility but requiring specialized tuning techniques. Few-shot and zero-shot prompting techniques allow LLMs to perform tasks with minimal or no task-specific training examples, significantly reducing the need for extensive labeled data[1]. These strategies have shown promise in various contexts, highlighting the adaptability of LLMs to new tasks with limited supervision.

Researchers have also developed a catalog of prompt patterns to solve common problems when interacting with LLMs[2]. These patterns, which include guidelines for crafting effective prompts, address issues such as prompt ambiguity, model misinterpretations, and response consistency[2]. provide a taxonomy of prompt patterns that categorize different approaches based on task types, such as classification, generation, and summarization, thus offering a structured framework for prompt design that can be easily applied across different NLP tasks. The effectiveness of different prompting strategies, such as simple prefix, cloze, chain of thought, and anticipatory prompts, has been empirically evaluated for clinical NLP tasks[3]. For instance, the chain-of-thought prompting technique encourages LLMs to break down complex tasks into intermediate reasoning steps, improving performance on tasks that require logical progression and detailed explanation. Anticipatory prompts, on the other hand, guide models to predict subsequent responses by leveraging prior knowledge of likely outcomes, enhancing the coherence and relevance of generated text, particularly in domains requiring domain-specific reasoning like clinical NLP[3].

Additionally, novel prompting techniques like heuristic and ensemble prompting have been introduced[3]. Heuristic prompting involves using domain-specific rules or prior knowledge to construct prompts that better align with the task at hand, while ensemble prompting combines multiple prompts to aggregate the strengths of different strategies, enhancing overall model performance and reducing variability in responses. These approaches provide robust alternatives to traditional single prompt methods, demonstrating the potential of combining multiple prompt types for more reliable outputs. While most studies focus on single-task prompts, some research has explored multi-task prompting approaches across various applications, from question-answering to commonsense reasoning[4]. Multitask prompting aims to create a single prompt that can handle multiple related tasks, thus improving the efficiency of LLMs by reducing the need to design task-specific prompts. This approach has been particularly beneficial in scenarios where models must adapt quickly to a wide range of questions or tasks without retraining, underscoring the versatility of prompt engineering as a technique for broadening the applicability of LLMs[4].

These advancements in prompt engineering contribute to improving LLMs' performance across diverse NLP tasks without modifying core model parameters. By optimizing the way models interact with input prompts, researchers are able to enhance LLMs' capabilities in handling complex, varied, and specialized tasks, driving forward the field of NLP and expanding the utility of these powerful models in real-world applications.

3. Data Preparation

In this study, we constructed our dataset by utilizing the IMDB review dataset available on Kaggle[5], which is a widely recognized dataset for sentiment analysis research. The IMDB dataset comprises movie reviews labeled with binary sentiment values, where each review is marked as either positive or negative. This dataset serves as the foundation for our comparative analysis of prompt strategies for Large Language Models (LLMs) in performing sentiment analysis and named entity recognition (NER). Since that the cardinality of IMDB review is quite high, we casually sampled a thousand elements from them.

To enhance the dataset with named entity information, we employed the SpaCy library, which is known for its state-of-the-art performance in English language tasks[6]. SpaCy is based on the transformer architecture, which has demonstrated superior capabilities in capturing contextual information and identifying entities in text compared to traditional models. We chose the model en_core_web_trf due to its robust performance and accuracy in NER tasks, making it a suitable baseline for our experiments. For each review in the IMDB dataset, we applied SpaCy's NER function to extract named entities, which include people (PER), locations (LOC), and organizations (ORG). The extraction process involves tokenizing the text, identifying potential entities using the transformer-based architecture, and classifying them into the aforementioned categories. This addition of entity recognition enriches the dataset, allowing us to evaluate the capabilities of LLMs in performing both sentiment analysis and NER within a single framework. Upon processing the dataset with SpaCy, we introduced a new column labeled "entities" to store the extracted named entities for each review. This column contains a list of entities identified in the text, providing a structured representation of the entity data. Each entry in the "entities" column is a dictionary that includes the entity type (PER, LOC, ORG) and the corresponding entity text as identified by the SpaCy model.

The final dataset structure comprises three primary components: the review text, the sentiment label, and the named entities. We constructed the JSON output by assigning the "review" key to the original text from the IMDB dataset, the "sentiment" key to the sentiment label provided by the IMDB ground truth, and the "entities" key to the list of named entities extracted by SpaCy. This structured JSON format facilitates a comprehensive analysis of LLM performance in handling both single-task and multitask prompts, enabling a rigorous evaluation of the efficiency and accuracy of different prompt strategies. This dataset preparation process ensures that our experimental setup is robust and capable of testing the performance of LLMs across multiple tasks simultaneously. By leveraging the IMDB dataset's extensive sentiment annotations and enriching it with high-quality named entity data from SpaCy, we create a rich testing ground for assessing the efficacy of prompt strategies in LLMs. This approach not only provides a clear benchmark for performance comparison but also highlights the practical applications of LLMs in real-world tasks that require the integration of sentiment analysis and NER.

The decision to use the IMDB review dataset for this study was driven by our desire to work with common-use texts, which are typically written by everyday users who may not consistently adhere to standard grammatical rules. The IMDB dataset offers a collection of movie reviews that reflect a wide range of writing styles and levels of linguistic proficiency. This variability provides a realistic and challenging environment for testing the capabilities of Large Language Models (LLMs). Working with user-generated content such as IMDB reviews is particularly valuable because it simulates real-world conditions where language models must process and understand text that can be informal, contain typos, or lack standard punctuation and sentence structure. By choosing a dataset characterized by such linguistic diversity, we aim to evaluate how effectively LLMs can handle the complexity of natural language in a practical context. This choice also allows us to assess the robustness of LLMs in scenarios that closely resemble typical user interactions with AI systems. Furthermore, the IMDB dataset's extensive popularity and previous use in sentiment analysis research make it an ideal candidate for our study. Its well-established benchmark provides a solid foundation for comparison, enabling us to measure the effectiveness of different prompt strategies against known standards. By selecting the

IMDB dataset, we ensure that our findings are relevant and applicable to a wide range of applications involving user-generated text.

In this study, we focused on evaluating the performance of five different open-source Large Language Models (LLMs) for sentiment analysis and named entity recognition tasks. The primary motivation behind selecting open-source models was to enhance the reproducibility of our research. By choosing models that are accessible to the public, we ensure that other researchers can replicate our experiments, verify our findings, and build upon our work without the constraints often associated with proprietary models. While state-of-the-art LLMs such as OpenAI's GPT, Anthropic's Claude, and Google's Gemini have demonstrated remarkable performance across a wide range of language tasks, their proprietary nature poses challenges for academic research in terms of accessibility and transparency. These models often come with usage restrictions, limited customization options, and require considerable computational resources, which can hinder reproducibility and broader scientific exploration. Therefore, to foster an open and collaborative research environment, we selected a set of high-performing open-source models that provide a balance between accessibility and capability. The specific LLMs we employed in our study are as follows:

1. LLama 3.1 8B (8b-instruct-q4_0).
2. Phi3 Medium (14b-medium-128k-instruct-q4_0).
3. Qwen2 7B (7b-instruct-q4_0).
4. Gemma2 9B (9b-instruct-q4_0).
5. Mistral 7B (7b-instruct-v0.3-q4_0).

The selection of these specific open-source models was guided by several factors. By utilizing open-source models, we enhance the reproducibility of our research, enabling other researchers to replicate our experiments and validate our findings without proprietary restrictions. Open-source models are generally more accessible, allowing a wider range of researchers and practitioners to engage with the research, regardless of their institutional or financial resources. Despite not being the absolute state-of-the-art, the chosen models still offer competitive performance on sentiment analysis and named entity recognition tasks, providing meaningful insights into LLM capabilities. Open-source models benefit from active community involvement, which leads to continuous improvements and innovations. This collaborative environment fosters the development of robust models that evolve in response to community needs and feedback. The selected models have varying parameter sizes that allow for experimentation on different computational platforms, from local machines to more extensive cloud-based infrastructures, facilitating scalable research approaches. In order to determine the 5 candidates, we observed the "Open LLM Leaderboard 2" hosted on Huggingface that lists the open-source LLM performances[7–15]. We accessed that leaderboard in June 2024. By focusing on open-source LLMs, this study not only provides valuable insights into the effectiveness of prompt strategies but also contributes to a body of work that prioritizes transparency and accessibility in AI research. This approach aligns with the broader goals of promoting open science and collaborative innovation within the machine learning community.

4. The Experiment

The objective of our experiment is to generate a JSON output for each review in the dataset, which will be directly compared to the ground truth JSON. The ground truth JSON includes the sentiment label, named entities, and the original review text extracted from the IMDB dataset (as shown in Listing 1). To achieve this, we employed two distinct approaches: a single-task approach and a multitask approach. These methodologies were designed to test the efficiency and accuracy of different prompt strategies for Large Language Models (LLMs) in handling multiple tasks simultaneously versus individually. The two experimental workflows provide a comprehensive framework for evaluating the efficiency and accuracy of single-task versus multitask prompts in LLMs. The outputs from both approaches were systematically compared against the ground truth JSON to determine the following key metrics: accuracy, which involves the correctness of sentiment classification and named entity recognition in both single-task and multitask scenarios; and consistency, which refers to the consistency of JSON formatting and structure across different prompt strategies.

In the single-task prompt approach, we decoupled the tasks of sentiment classification, named entity recognition (NER), and JSON formatting into separate, distinct operations. The goal was to isolate each task to determine how effectively LLMs can perform when given a dedicated prompt for each task. The process began with sentiment classification. We first used a single-task prompt to classify the binary sentiment of each review. The LLMs were prompted to read the review text and determine whether the sentiment was positive or negative. All 1,000 elements in our dataset were processed independently, with each review being fed into the LLM, and a sentiment label (either "positive" or "negative") was generated for each review. This operation produced an initial output consisting solely of sentiment classifications. Following sentiment classification, a separate prompt was employed to extract named entities from the reviews. The entities included people (PER), locations (LOC), and organizations (ORG). Each review was again processed individually through the LLMs, this time with a focus on identifying and classifying named entities. The output of this step was a collection of lists containing the extracted entities for each review. The final step involved formatting the results into a structured JSON format using the outputs from the first two tasks. The sentiment labels and named entities were combined with the original review text to create a JSON object for each review. Each JSON object included keys for "review," "sentiment," and "entities," matching the structure of the ground truth JSON. This step ensured that the outputs were aligned with the expected format for comparison. The used single-task prompts are shown in Appendix 3.

The multitask prompt approach was designed to evaluate the performance of LLMs when tasked with performing multiple tasks simultaneously. In this method, a single prompt was used to instruct the LLMs to carry out sentiment classification, named entity recognition, and JSON formatting in one unified operation (as shown in Appendix 2. In this approach, we used a unified prompting strategy where the LLMs were provided with a single, comprehensive prompt for each review. This prompt instructed them to analyze the sentiment, extract named entities, and format the results into a JSON object. All 1,000 reviews were processed in a batch manner, with each review being fed into the LLM with a multitask prompt designed to handle all three tasks at once. The output for each review was a complete JSON object containing the sentiment classification, extracted entities, and the review text itself. By handling sentiment analysis, named entity recognition, and JSON formatting in parallel, the multitask approach leverages the LLMs' ability to process complex, integrated prompts. The final output for each review was a JSON object directly generated by the LLM, structured similarly to the ground truth JSON. This direct approach allows for the assessment of the LLMs' capability to multitask effectively and efficiently.

The experiment was executed in Jupyter notebooks, which are publicly available on our GitHub Repository[16]. This decision aligns with our commitment to transparency and reproducibility, allowing other researchers to access and replicate our findings with ease. The Jupyter notebooks provide a detailed step-by-step account of the entire experimental workflow, including data preprocessing, prompt formulation, and LLM interactions. By using Jupyter notebooks, we ensure that each experiment is documented in a manner that captures the nuances of our methodology, from data ingestion to output generation. These notebooks not only contain the code used for executing each task but also include commentary and insights into the decisions made throughout the experiment. This transparency is crucial for fostering collaboration and innovation within the research community. In addition to the code, the dataset utilized in our study is also hosted on the same GitHub repository. The dataset includes the processed IMDB reviews, along with the generated JSON outputs for both single-task and multitask approaches. By providing both the data and the code, we facilitate an open-access environment where researchers can easily validate and build upon our work. Our GitHub repository serves as a comprehensive resource for those interested in exploring the intricacies of LLM prompt strategies. It invites further exploration and experimentation, offering a platform for continuous improvement and shared learning within the field of natural language processing. The experiments

doi:10.20944/preprints202410.1334.v1

were conducted with a temperature setting of 0.8. This choice was made to simulate a standard-case scenario, where the temperature is neither at its maximum value nor at the zero level, ensuring a balance between variability and determinism in the model's output.

To facilitate the deployment and execution of the five chosen Large Language Models (LLMs) for our experiments, we utilized Ollama, a platform designed for serving LLMs locally. We chose Ollama due to its ease of use and its alignment with our commitment to reproducibility. The adoption of Ollama ensures that our experimental setup can be easily replicated by others in the research community, fostering transparency and collaboration. Ollama provides a user-friendly interface that simplifies the process of running LLMs on local machines. Its straightforward installation and configuration process allow researchers and practitioners to quickly set up and deploy models without the need for extensive technical expertise. This accessibility makes it an ideal choice for our study, as it ensures that the experiments can be reproduced with minimal effort. The choice of Ollama was also influenced by its active support for a wide range of open-source models, including the five we selected: LLama 3.1 7B, Phi3 Medium, Qwen2 7B, Gemma2 7B, and Mistral 7B. This compatibility allows for seamless integration of these models into our experimental framework, enabling consistent and efficient execution of tasks across different LLMs. By using Ollama, we adhere to the principle of reproducibility that we set for ourselves. The platform's widespread usability and accessibility mean that anyone interested in our research can easily replicate the environment and methodology used in our experiments. This commitment to open science not only enhances the credibility of our findings but also encourages further exploration and validation by other researchers.

4.1. Evaluation

In order to rigorously assess the performance of the Large Language Models (LLMs) employed in our study, we designed a comprehensive evaluation metric that integrates three critical components: sentiment accuracy, named entity recognition (NER) performance, and review text fidelity. Our evaluation metric is designed to provide a holistic measure of how effectively the LLMs handle the tasks of sentiment classification, named entity extraction, and text reproduction. The evaluation metric is defined as follows:

$$f(x, y, z) = \frac{x + y + z}{3}$$

Where x is the accuracy of sentiment detection, defined as it follows.

$$x = \begin{cases} 1 & \text{if the detected sentiment is correct} \\ 0 & \text{otherwise} \end{cases}$$

Then, y represents the F1 score of the named entity recognition (NER). True positives are entities present in both the ground truth and the output. False negatives refer to those missing from the output but included in the ground truth, while false positives are found in the output but absent from the ground truth.

This score reflects the precision and recall balance achieved by the LLMs in identifying and classifying named entities within the review text. The "exact match" approach has been used here. z is the BLEU (Bilingual Evaluation Understudy) score of the review, which measures how closely the generated review matches the original text. This component of the metric evaluates the LLM's ability to reproduce the review text accurately. The function $f(x, y, z)$ computes the arithmetic mean of x, y, and z, providing an overall performance score for each review processed by the LLMs. By averaging these three metrics, we achieve a balanced evaluation that considers both classification accuracy and text generation quality. The rationale behind this metric is to ensure that each aspect of the task is equally weighted, acknowledging the importance of sentiment accuracy, entity recognition precision, and fidelity to the original review text. This holistic approach allows us to capture the multifaceted nature of the task, offering a robust framework for evaluating the effectiveness of different prompting strategies in LLMs.

The LLM's output is postprocessed in order to extract the right JSON data. This is due to handle the fact that LLMs often respond discoursively without providing strict JSON. Two distincts regular expressions are used in this scenario. The former one, mentioned in Appendix 4 is used to remove JSON comments. Sometimes, LLMs answer following the JSONC format instead of JSON, so we need to remove all the comments in order to parse the data correctly. Then, the second one, mentioned in Appendix 5 expression is used to extract the JSON string. Basically, it extracts the group starting from the first left curly bracket and ending with the last right curly bracket. Moreover, LLMs tend to respond with a JSON that uses single apices instead of double quotes. So, in a pythonic manner the algorithm works as shown in the Appendix 6, assuming that the two regular expressions have already been applied. When the parsing fails, we attribute a score of 0% to that case.

5. Results

The analysis of our experimental results provides a nuanced perspective on the efficacy of multitask versus atomic single-task prompts. Contrary to our initial hypothesis, the data reveals that atomic single-task prompts approach does not uniformly outperform a multitask prompt across all contexts.

Our study highlights significant variability in prompt effectiveness depending on the specific model used. This observation suggests that the interaction between prompt type and model architecture is complex and warrants careful consideration. Specifically, the performance of a given prompt can be highly sensitive to the underlying model's characteristics, indicating that model-specific factors play a crucial role in determining the relative success of prompting strategies.

In detail, our experiments yielded mixed outcomes. Out of the five distinct experimental setups, three demonstrated that atomic single-task prompts were more effective than their multitask counterparts. These results suggest that for certain tasks, simpler and more specialized prompts may offer advantages in terms of accuracy or efficiency. Conversely, two experiments showed that multitask prompts provided superior performance, challenging the assumption that simplicity inherently leads to better outcomes. This variability underscores the importance of tailoring the prompting approach to the specific task and model, rather than relying on a one-size-fits-all strategy.

Furthermore, the unexpected nature of our findings is worth noting. Despite the theoretical benefits of atomic single-task prompts such as the potential for improved efficiency and generalization the empirical evidence from our study does not consistently support these advantages. We had anticipated that the low complexity associated with single-task prompts would correlate with enhanced performance. However, the results indicate that this expectation does not always hold true in practice. The complexity of single-task prompts did not translate into universally superior outcomes compared to the relatively straightforward multitask prompts.

Additionally, our investigation included a range of models with varying sizes, from 2 billion to 14 billion parameters. The results from these experiments did not reveal a clear relationship between model size and prompt effectiveness. This finding suggests that the performance of prompting strategies is not solely dependent on the scale of the model but is influenced by other factors, such as task characteristics and prompt design. The table below shows the mean scores for each model and approach, where "scores" mean the metric explained in Section 4.

Table 1. Mean Scores for Different Models

Model	Multitask	Single-Task
Gemma2 9B (9b-instruct-q4_0)	80.74%	**81.32%**
Qwen2 7B (7b-instruct-q4_0)	54.01%	**60.98%**
LLama 3.1 8B (8b-instruct-q4_0)	**71.88%**	67.21%
Phi3 Medium (14b-medium-128k-instruct-q4_0)	25.65%	**43.68%**
Mistral 7B (7b-instruct-v0.3-q4_0)	**62.88%**	60.26%

In this section we want to discuss the evaluation of models. In order to clarify the results, we prepared a set of density plots that highlight the score distribution comparing single-task to multitask.

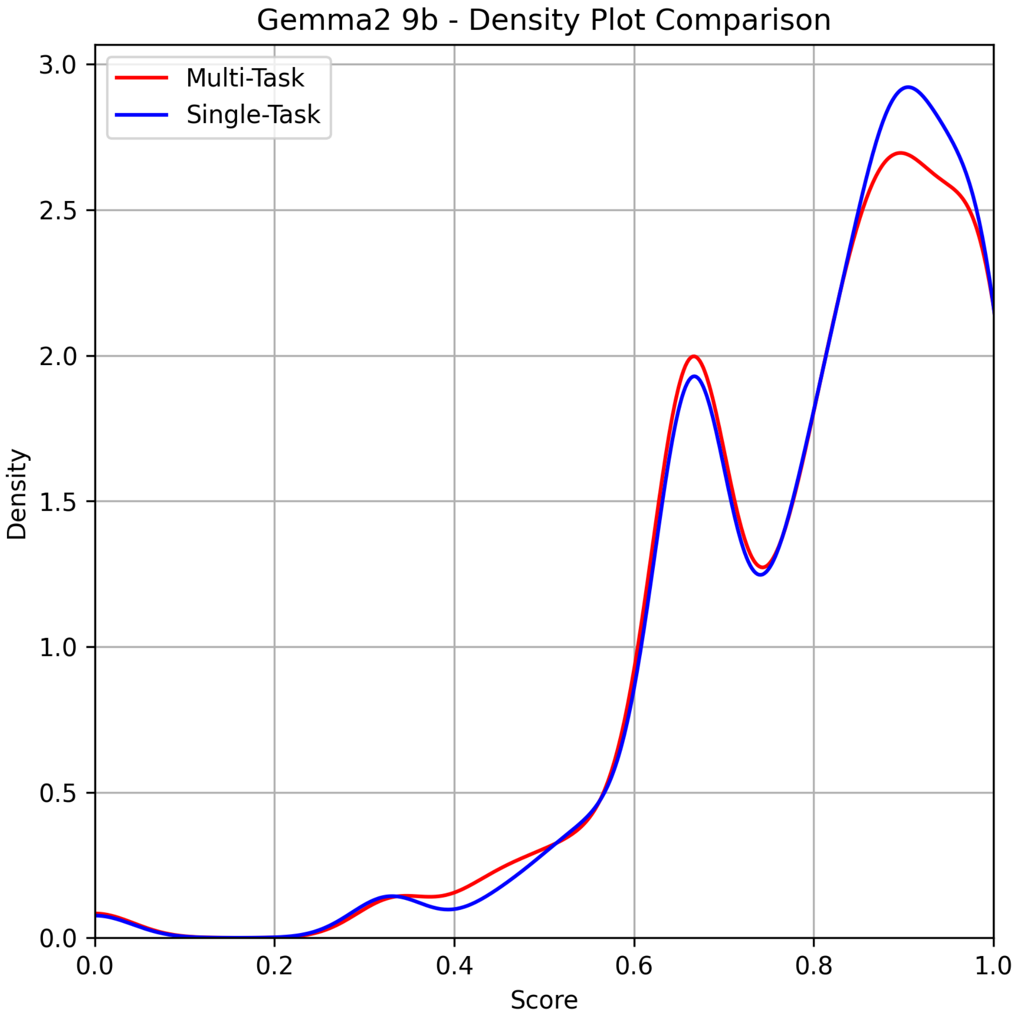

Figure 1. Gemma 2 9B.

Gemma 2 9B outperformed all other models in this study. Although the assumption that single-task prompts yield better results compared to the dual, multitask approach still holds true, it is worth noting that the difference in performance is not particularly pronounced. A key area of interest lies in the density range between 0.0 and 0.4, where the disparity is notably minimal. Shifting the focus to individual tasks, the table below provides a detailed breakdown of the results.

The data presented in Table 2 reveals nuanced differences between the multitask and single-task approaches for the Gemma 2 9B model across various NLP tasks. While the single-task approach slightly outperforms the multitask approach in certain metrics, such as Exact-Match on sentiment

(91.50% vs. 90.00%) and NER F1 score (55.75% vs. 54.75%), the differences are relatively small. Interestingly, multitask prompting shows better performance in terms of NER Precision (60.99% vs. 59.86%), indicating that the model's precision in recognizing named entities may benefit from the multitask approach. However, the slight advantage in NER Recall for the single-task approach (56.87% vs. 54.11%) suggests a trade-off between precision and recall. The formatting error rate remains comparably low for both methods, with a minimal difference (9.00‰vs. 8.00‰), further reinforcing the notion that prompt complexity does not drastically affect performance in this specific domain.

Table 2. Specific-task performances on Gemma 2 9B.

Metric	Multitask	Single-Task
Mean BLEU on review	**97.49%**	96.70%
Mean Exact-Match on sentiment	90.00%	**91.50%**
Mean NER F1	54.75%	**55.75%**
Mean NER Precision	**60.99%**	59.86%
Mean NER Recall	54.11%	**56.87%**
Formatting error rate	9.00‰	**8.00‰**

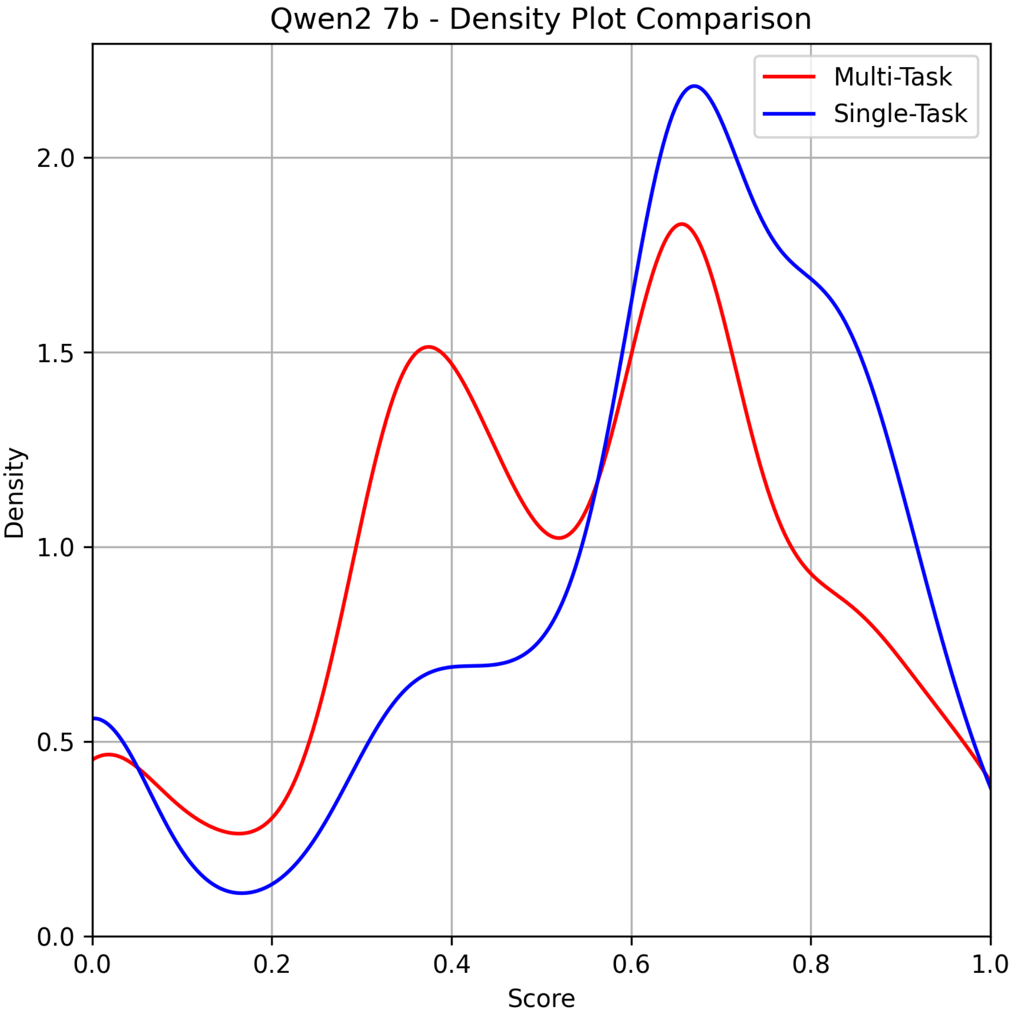

Figure 2. Qwen 2 7B.

For Qwen 2 7B, the assumption that single-task prompts yield better results compared to the dual, multitask approach remains valid. However, in contrast to Gemma 2, the observed density pattern is more erratic, and the performance gap between the two approaches becomes more pronounced in this case.

Table 3 highlights a more significant divergence between the multitask and single-task approaches for Qwen 2 7B compared to Gemma 2 9B. Single-task prompts outperform multitask prompts across most metrics, particularly in the Mean BLEU score on review tasks (73.26% vs. 56.09%), indicating a substantial advantage in generating coherent and accurate text for single-task prompts. Similarly, the Exact-Match score for sentiment analysis is higher for the single-task approach (82.70% vs. 80.80%).

Table 3. Specific-task performances on Qwen 2 7B.

Metric	Multitask	Single-Task
Mean BLEU on review	56.09%	**73.26%**
Mean Exact-Match on sentiment	80.80%	**82.70%**
Mean NER F1	25.13%	**26.98%**
Mean NER Precision	**32.20%**	27.97%
Mean NER Recall	24.32%	**30.79%**
Formatting error rate	**34.00‰**	88.00‰

However, an interesting deviation can be observed in NER Precision, where multitask prompts demonstrate better performance (32.20% vs. 27.97%), suggesting that Qwen 2 7B's ability to precisely recognize named entities benefits from the complexity of the multitask setup. Despite this, single-task prompts yield higher NER Recall (30.79% vs. 24.32%), reflecting a trade-off between precision and recall similar to what was observed in the previous model. Additionally, the formatting error rate is notably higher for single-task prompts (88.00‰vs. 34.00‰), suggesting that while single-task prompts may improve content accuracy, they introduce a greater risk of formatting errors, a factor worth considering in practical applications.

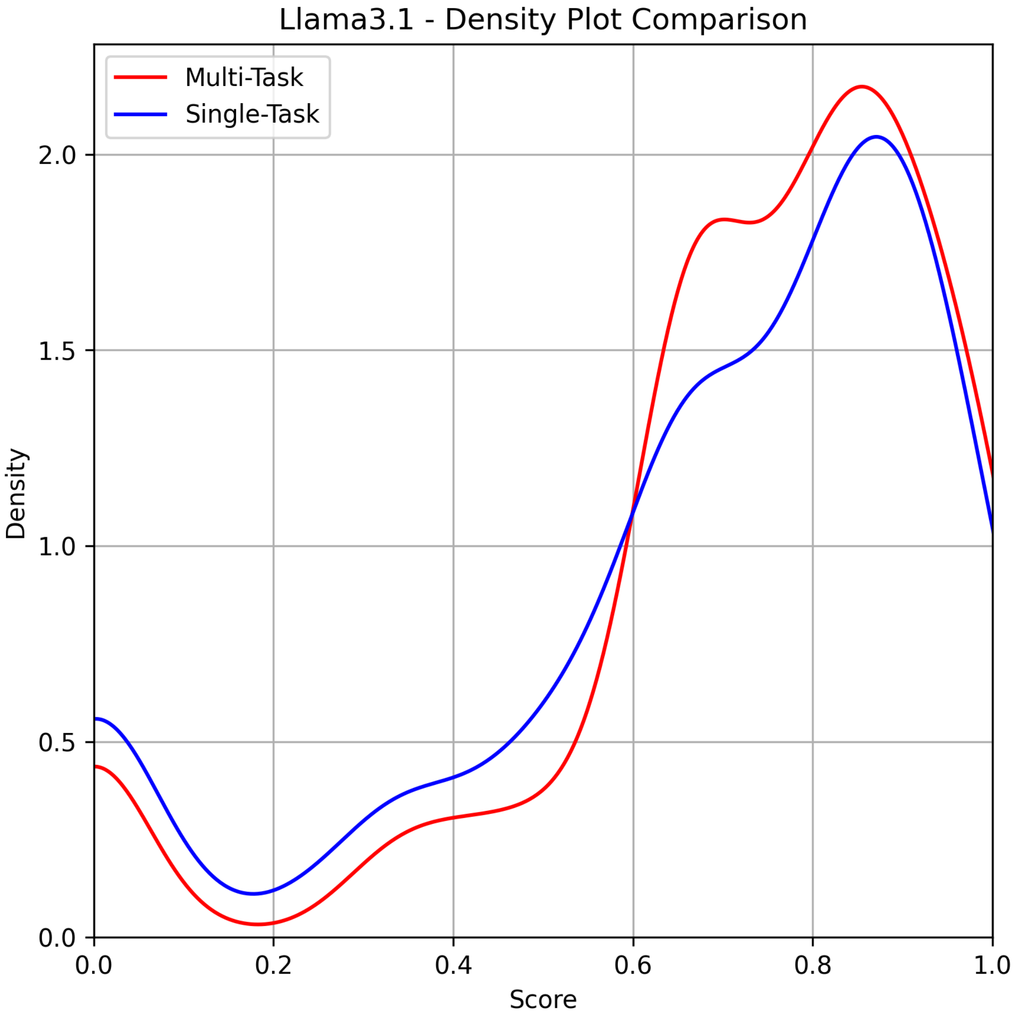

Figure 3. LLama 3.1 8B.

LLama 3.1 8B is the first model to deviate from the expected pattern. In contrast to previous models, the multitask approach outperforms the single-task approach, demonstrating a clear reversal of the trends observed earlier.

Table 4 reveals a distinctive performance pattern for LLama 3.1 8B, where the multitask approach shows superior results compared to the single-task approach across most metrics. Notably, the Mean BLEU score on review tasks is significantly higher for multitask prompts (88.94% vs. 76.55%), indicating that LLama 3.1 8B generates more coherent and contextually accurate responses when handling multiple tasks simultaneously. Similarly, in sentiment analysis, the multitask approach outperforms the single-task approach with a higher Exact-Match score (83.70% vs. 81.00%). However, the NER metrics present a more nuanced picture. While the single-task approach achieves a slightly higher F1 score (44.10% vs. 43.00%) and NER Recall (46.01% vs. 42.05%), multitask prompts excel in NER Precision (50.25% vs. 47.98%). This suggests that LLama 3.1 8B is more precise but slightly less comprehensive in recognizing named entities when dealing with multitask prompts. Additionally, the formatting error rate is notably lower in the multitask setting (69.00‰vs. 94.00‰), indicating that multitask prompts not only yield better content accuracy but also lead to fewer formatting errors. These results underscore the model's capacity to handle multitask scenarios effectively, challenging the conventional assumption that single-task prompting is inherently superior.

Table 4. Specific-task performances on LLama 3.1 8B.

Metric	Multitask	Single-Task
Mean BLEU on review	**88.94%**	76.55%
Mean Exact-Match on sentiment	**83.70%**	81.00%
Mean NER F1	43.00%	**44.10%**
Mean NER Precision	**50.25%**	47.98%
Mean NER Recall	42.05%	**46.01%**
Formatting error rate	**69.00‰**	94.00‰

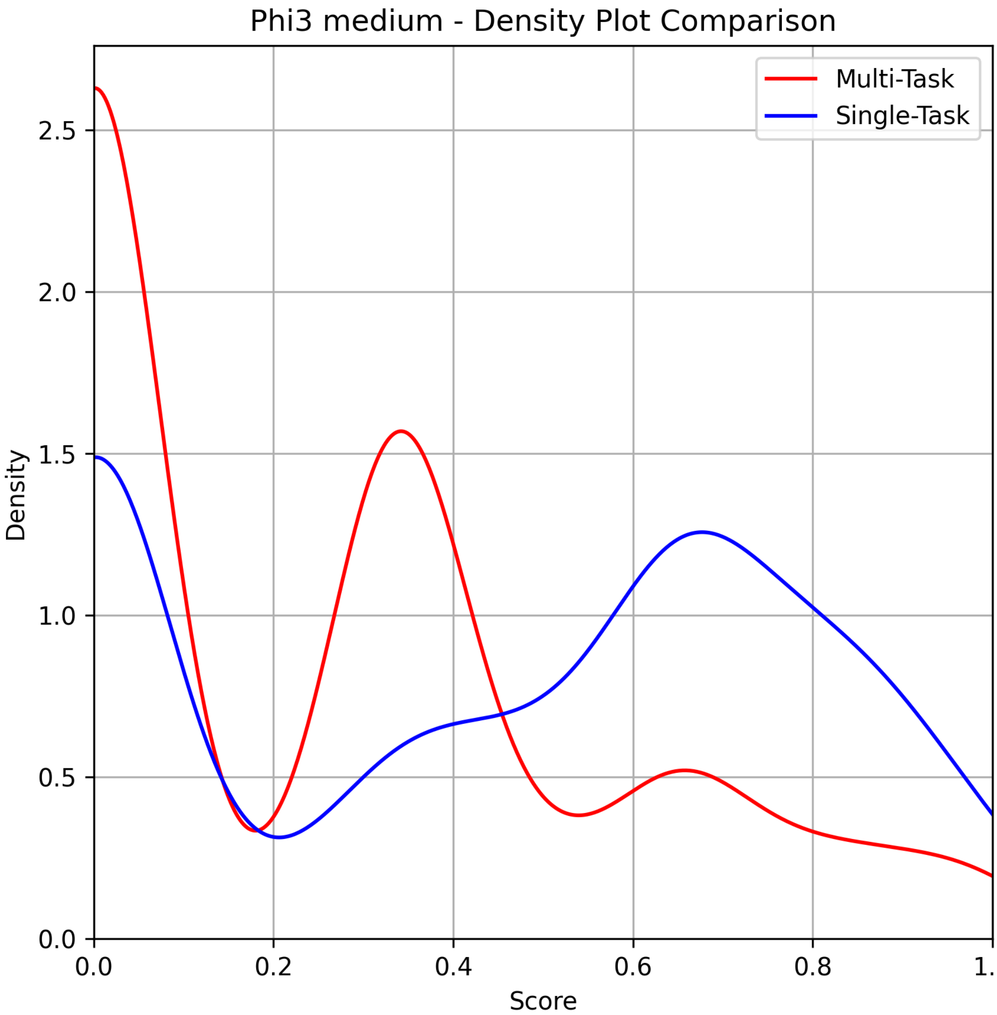

Figure 4. Phi 3 Medium.

Regarding Phi3 Medium 14B, it can be unequivocally stated that its performance was the worst among all models in the experimental set. The difference in performance between the two prompting approaches is particularly stark, with the single-task approach significantly outperforming the multitask approach.

Table 5 illustrates that Phi3 Medium 14B exhibits the weakest overall performance across all evaluated models. The results clearly demonstrate a substantial gap between the multitask and single-task approaches, with the latter consistently outperforming the former. For instance, the Mean BLEU score on review tasks is notably higher for single-task prompts (57.63% vs. 16.98%), indicating that Phi3 Medium 14B struggles significantly with generating coherent text in multitask scenarios.

Similarly, the Exact-Match score for sentiment analysis shows a slight but consistent improvement in single-task settings (50.80% vs. 48.30%).

Table 5. Specific-task performances on Phi 3 Medium.

Metric	Multitask	Single-Task
Mean BLEU on review	16.98%	**57.63%**
Mean Exact-Match on sentiment	48.30%	**50.80%**
Mean NER F1	11.68%	**22.62%**
Mean NER Precision	14.49%	**25.45%**
Mean NER Recall	11.06%	**23.78**%
Formatting error rate	**253.00‰**	307.00‰

The disparity is even more pronounced in NER tasks, where the single-task approach nearly doubles the F1 score (22.62% vs. 11.68%) and achieves higher Precision (25.45% vs. 14.49%) and Recall (23.78% vs. 11.06%). These results suggest that Phi3 Medium 14B's ability to recognize and categorize named entities is severely hindered in multitask settings.

Moreover, both approaches exhibit high formatting error rates, with the single-task method slightly worse (307.00‰vs. 253.00‰). This suggests that, although the single-task approach improves performance in content-related tasks, both prompting methods struggle with formatting precision. These results position Phi3 Medium 14B as the least capable model in handling complex or multitask scenarios, emphasizing the limitations of this particular architecture in the context of large-scale language models.

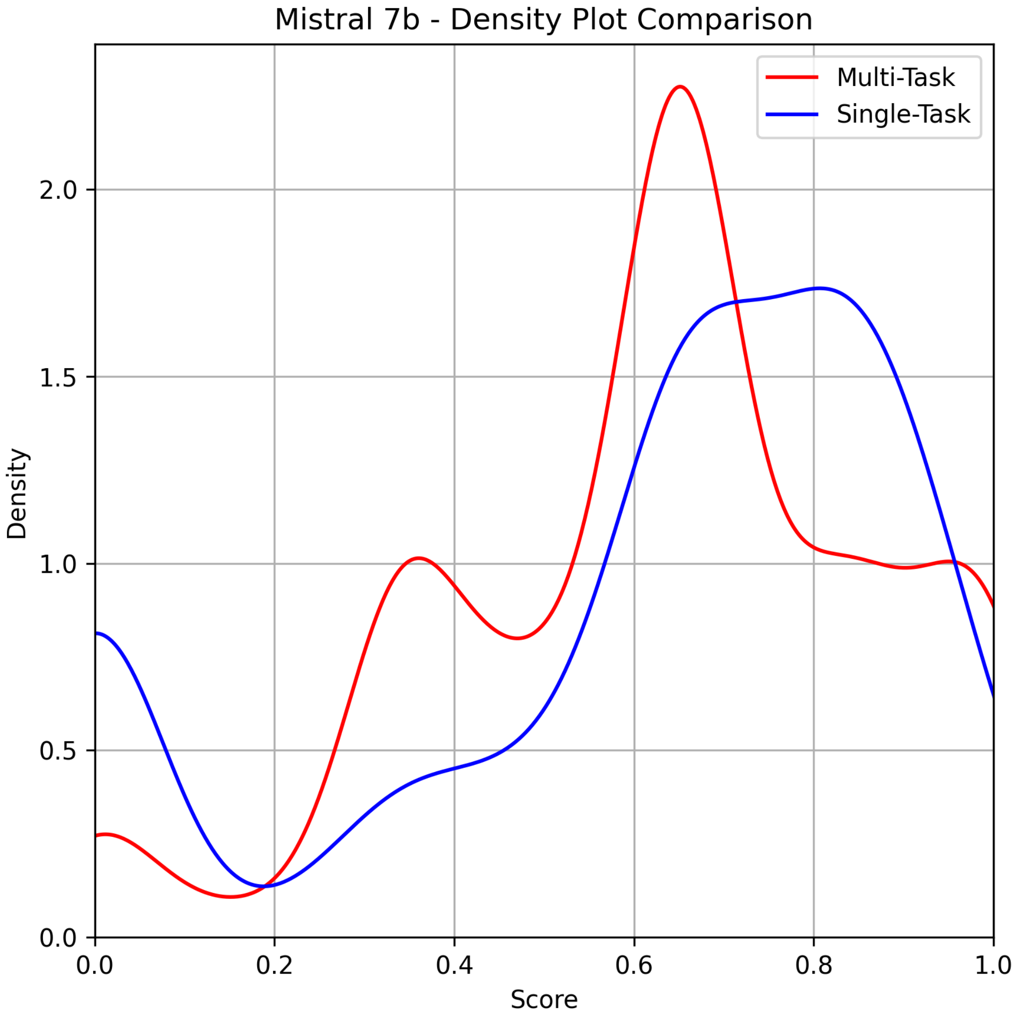

Figure 5. Mistral 7B.

Mistral 7B is the most emblematic model in this study. The performance trend is not only discontinuous, but it is also evident that the score density is higher for the multitask approach between 0.2 and 0.66, while from 0.66 to 0.9, the single-task approach performs better. However, the tail of the distribution clearly favors the multitask approach as the superior method.

Table 6 reflects the mixed performance of Mistral 7B across various tasks, showcasing both strengths and weaknesses depending on the task and prompting approach. In terms of review generation, the single-task approach achieves a higher Mean BLEU score (76.41% vs. 70.84%), indicating better text generation performance for single-task prompts. However, for sentiment analysis, the multitask approach far outperforms the single-task method with a notably higher Exact-Match score (87.20% vs. 71.80%). In Named Entity Recognition (NER), the results are more nuanced. While the single-task approach yields a slightly higher F1 score (32.56% vs. 30.60%) and Recall (33.50% vs. 28.04%), the multitask approach achieves better Precision (39.28% vs. 37.05%), highlighting a trade-off between completeness and accuracy in entity recognition. One of the most significant differences is observed in the formatting error rate, where the multitask approach significantly outperforms the single-task approach, with a much lower error rate (29.00‰vs. 155.00‰). This suggests that multitask prompts not only excel in specific tasks, such as sentiment analysis, but also lead to more reliable formatting outputs. The overall distribution of scores confirms that Mistral 7B performs better under multitask settings in certain scenarios, although single-task prompts still offer advantages in specific metrics like BLEU and NER Recall.

Table 6. Specific-task performances on Mistral 7B.

Metric	Multitask	Single-Task
Mean BLEU on review	70.84%	**76.41%**
Mean Exact-Match on sentiment	**87.20%**	71.80%
Mean NER F1	30.60%	**32.56%**
Mean NER Precision	**39.28%**	37.05%
Mean NER Recall	28.04%	**33.50%**
Formatting error rate	**29.00‰**	155.00‰

The experiments conducted on the five models—Gemma 2 9B, Qwen 2 7B, LLama 3.1 8B, Phi3 Medium 14B, and Mistral 7B—present a diverse and complex landscape of performance across single-task and multitask prompting approaches. While the general assumption that single-task prompts yield superior results holds true for most models, the nuances of performance reveal significant variations depending on the architecture and task type. Gemma 2 9B and Qwen 2 7B conform to expectations, with single-task prompts outperforming multitask prompts across the majority of metrics, though the difference is more pronounced in Qwen 2 7B. In contrast, LLama 3.1 8B challenges this assumption, demonstrating superior results with multitask prompts, particularly in generating coherent text and sentiment analysis, signaling that not all models adhere to a uniform performance pattern. Phi3 Medium 14B exhibited the weakest overall performance, with both single-task and multitask approaches underperforming compared to the other models, but with the single-task approach consistently outperforming the multitask one. This highlights potential limitations in the model's architecture when handling both simple and complex tasks. Mistral 7B presents a mixed profile, with performance fluctuating between the two prompting approaches depending on the task. While single-task prompts show an advantage in text generation and NER Recall, multitask prompts excel in sentiment analysis and NER Precision, with a notably lower formatting error rate, suggesting that Mistral 7B is more versatile but less predictable. Overall, these experiments underscore the importance of selecting the appropriate prompting strategy based on the specific model and task at hand. While single-task prompts generally offer better performance, certain models like LLama 3.1 8B and Mistral 7B demonstrate the potential of multitask prompts to exceed single-task results in specific contexts. The diverse outcomes across these models suggest that optimizing prompting strategies for LLMs should be model-specific and task-aware, rather than guided by a one-size-fits-all approach.

6. Conclusions

Our comparative study on the effectiveness of multi-task versus single-task prompts for Large Language Models (LLMs) has yielded complex and unexpected results, challenging the initial hypothesis of single-task prompt superiority. Gemma 2 9B exhibited the best overall performance, showing a slight preference for single-task prompts. However, the difference was minimal, with nearly equivalent performance in sentiment accuracy (91.50% vs 90.00%) and F1 score for entity recognition (55.75% vs 54.75%). Qwen 2 7B showed a more pronounced advantage for single-task prompts, particularly in the BLEU score for text generation (73.26% vs 56.09%), but it performed better with multi-task prompts for NER precision (32.20% vs 27.97%), indicating a trade-off between precision and recall. LLama 3.1 8B defied expectations by performing better with multi-task prompts, as demonstrated by its higher BLEU score (88.94% vs 76.55%) and sentiment accuracy (83.70% vs 81.00%). These results suggest greater consistency and precision in text generation and sentiment analysis in multi-task scenarios. Phi3 Medium 14B delivered the weakest performance, with a clear preference for single-task prompts. The difference was particularly stark in the BLEU score (57.63% vs 16.98%) and the F1 score for NER (22.62% vs 11.68%), indicating significant difficulties in handling simultaneous tasks. Mistral 7B presented the most varied profile. While single-task prompts resulted in a better BLEU score (76.41% vs 70.84%), multi-task prompts excelled in sentiment accuracy (87.20% vs 71.80%) and showed a significantly lower formatting error rate (29.00‰vs 155.00‰).

These findings underscore that prompt effectiveness not only varies across models, but some, such as LLama 3.1 8B and Mistral 7B, perform better with multi-task prompts. In certain cases, these results challenge the assumption that the simplicity of single-task prompts is always advantageous. Additionally, the analysis revealed intriguing trade-offs, such as the balance between precision and recall in entity recognition, which varies depending on the prompt strategy employed. This suggests that the choice between single-task and multi-task prompts can influence not only overall accuracy but also the balance between different aspects of performance.

The practical implications of this study are significant: not only does it help guide the selection of more effective prompts for different models and applications, but it also suggests that further research should focus on understanding the mechanisms underlying performance variations. This would lead to a deeper understanding of the interactions between model architecture, task execution, and prompt complexity. Prompt optimization could significantly enhance the effectiveness of these systems in real-world contexts. Future studies could explore how models of varying sizes and architectures respond to different prompt types, developing methods to automate the selection of optimal prompts and reduce the need for trial and error. Such advancements will be crucial for improving LLM performance in the future.

Appendix A

Listing 1: Example of a ground truth JSON data.

```
{
  "review": "But at least this movie got what it deserved - to be sent to the Satellite of
      ↪ Love to be ridiculed on by Mike, Tom Servo, and Crow T. Robot from Pearl Forrester
      ↪ on \"Mystery Science Theater 3000!\" \"Soultaker\" is one of those long lost,
      ↪ forgotten movies that are so bad you'll be guaranteed to have nightmares or
      ↪ depression later on in life. Even though the movie is not that old, it's still a
      ↪ very forgotten type of movie. If it had never been for the intelligent minds at \"
      ↪ Mystery Science Theater 3000,\" the movie would not only seem like it was never made,
      ↪  but the movie wouldn't be very enjoyable by us moviegoers.<br /><br />In real life:
      ↪  this movie is really bad. In the Satellite of Love: this movie is excellent!",
  "sentiment": "negative",
  "entities": [
    {
      "label": "PERSON",
      "value": "Mike"
    },
    {
      "label": "PERSON",
      "value": "Tom Servo"
    },
    {
      "label": "PERSON",
      "value": "Crow T. Robot"
    },
    {
      "label": "ORG",
      "value": "Pearl Forrester"
    }
  ]
}
```

Listing 2: Multi-Task prompts.

~~~
    PROMPT_TEMPLATE = """Instruction: Analyze the following review text and provide
        ↪ one distinct outputs formatted in JSON:

1. **Sentiment Classification:** Indicate whether the sentiment of the review is "
    ↪ positive" or "negative".
2. **Named Entity Extraction:** List all named entities present in the text,
    ↪ categorizing them by label (PERSON, ORG, LOC).
3. **Required JSON Format:** Ensure the response is formatted in JSON according to
    ↪ the following schema:

{{
  "sentiment": "<sentiment>",
  "review": "<review>",
  "entities": [
    {{
      "label": "<label>",
      "value": "<value>"
    }}
  ]
}}

example:

"I recently visited the restaurant 'La Dolce Vita' in Rome and was thrilled with the
    ↪  service and food. The waiter, Marco, was exceptionally friendly and the
    ↪ truffle risotto was simply divine. I can't wait to return and recommend this
    ↪ place to my friends."

```json
{{
  "sentiment": "positive",
  "review": "I recently visited the restaurant 'La Dolce Vita' in Rome and was
      ↪ thrilled with the service and food. The waiter, Marco, was exceptionally
      ↪ friendly and the truffle risotto was simply divine. I can't wait to return
      ↪ and recommend this place to my friends.",
  "entities": [
    {{
      "label": "ORG",
      "value": "La Dolce Vita"
    }},
    {{
      "label": "LOC",
      "value": "Rome"
    }},
    {{
      "label": "PERSON",
      "value": "Marco"
    }}
  ]
}}
~~~

````
‘‘‘

{content}"""
````

Listing 3: Single-Task prompts.

````
    NER_PROMPT = """Analyze the following review text listing all named entities
        ↪ present in the text, categorizing them by label. Consider only PERSON,
        ↪ ORG, and LOC categories.
Ensure the response is formatted in JSON according to the following schema:

[
  {{
    "label": "<label>",
    "value": "<value>"
  }}
]

Example:

"I recently visited the restaurant 'La Dolce Vita' in Rome and was thrilled with the
    ↪  service and food. The waiter, Marco, was exceptionally friendly and the
    ↪ truffle risotto was simply divine. I can't wait to return and recommend this
    ↪ place to my friends."

[
  {{
    "label": "ORG",
    "value": "La Dolce Vita"
  }},
  {{
    "label": "LOC",
    "value": "Rome"
  }},
  {{
    "label": "PERSON",
    "value": "Marco"
  }}
]

{content}"""

SENTIMENT_PROMPT = """Analyze the following review text indicating whether the
    ↪ sentiment of the review is "positive" or "negative".

Example:

"I recently visited the restaurant 'La Dolce Vita' in Rome and was thrilled with the
    ↪  service and food. The waiter, Marco, was exceptionally friendly and the
    ↪ truffle risotto was simply divine. I can't wait to return and recommend this
    ↪ place to my friends."
````

```
"positive"

{content}"""

FORMATTING_OUTPUT = """You are given three informations: sentiment, review and
    ↪ entities. Generate a JSON representation using the following schema. Use just
    ↪  the data you receive:

{{
  "sentiment": "<sentiment>",
  "review": "<review>",
  "entities": [
    {{
      "label": "<label>",
      "value": "<value>"
    }}
  ]
}}

example:

"Sentiment: positive
Review: I recently visited the restaurant ’La Dolce Vita’ in Rome and was thrilled
    ↪ with the service and food. The waiter, Marco, was exceptionally friendly and
    ↪ the truffle risotto was simply divine. I can’t wait to return and recommend
    ↪ this place to my friends.
Entities: [
    {{
      "label": "ORG",
      "value": "La Dolce Vita"
    }},
    {{
      "label": "LOC",
      "value": "Rome"
    }},
    {{
      "label": "PERSON",
      "value": "Marco"
    }}
  ]"

‘‘‘json
{{
  "sentiment": "positive",
  "review": "I recently visited the restaurant ’La Dolce Vita’ in Rome and was
      ↪ thrilled with the service and food. The waiter, Marco, was exceptionally
      ↪ friendly and the truffle risotto was simply divine. I can’t wait to return
      ↪ and recommend this place to my friends.",
  "entities": [
    {{
      "label": "ORG",
      "value": "La Dolce Vita"
    }},
    {{
      "label": "LOC",
      "value": "Rome"
    }},
    {{
      "label": "PERSON",
      "value": "Marco"
    }}
  ]
}}
‘‘‘



Sentiment: {sentiment}
Review: {review}
Entities: {entities}"""
```

Listing 4: Regular expression to clean JSON-like LLM output.

```
clean_regex: str = r"(?<!\S)//.*?$"
```

Listing 5: Regular expression to extract JSON-like LLM output.

```
extract_regex: str = r"(\{.*\})"
```

Listing 6: JSON output parsing algorithm.

```
def parse(string: str) -> dict:
    dictionary: dict = None
    try:
        dictionary = json.loads(string)
    except Exception:
        pass
    try:
        dictionary = eval(string)
    except Exception:
        pass
    return dictionary
```

References

1. Yinheng Li, *A Practical Survey on Zero-Shot Prompt Design for In-Context Learning*, https://doi.org/10.26615/978-954-452-092-2_069, 2023.
2. Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, Douglas C. Schmidt, *A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT*, https://doi.org/10.48550/arXiv.2302.11382, 2023.
3. Sonish Sivarajkumar, Mark Kelley, Alyssa Samolyk-Mazzanti, S. Visweswaran, Yanshan Wang, *An Empirical Evaluation of Prompting Strategies for Large Language Models in Zero-Shot Clinical Natural Language Processing*, https://doi.org/10.48550/arXiv.2309.08008, 2023.

4. Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, S. Mondal, Aman Chadha, *A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications*, https://doi.org/10.48550/arXiv.2402.07927, 2024.
5. Lakshmipathi N, IMDB Dataset of 50K Movie Reviews, https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews, 2019.
6. Hemlata Shelar, Gagandeep Kaur, Neha Heda, Poorva Agrawal, Named Entity Recognition Approaches and Their Comparison for Custom NER Model, https://doi.org/10.1080/0194262x.2020.1759479, 2020.
7. Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. *Open LLM Leaderboard v2*. 2024. Published by Hugging Face. Available at: https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard.
8. Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. *A framework for few-shot language model evaluation*. Zenodo, 2021, v0.0.1. Available at: https://doi.org/10.5281/zenodo.5371628.
9. Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. *Instruction-Following Evaluation for Large Language Models*. arXiv, 2023. Available at: https://arxiv.org/abs/2311.07911.
10. Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. *Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them*. arXiv, 2022. Available at: https://arxiv.org/abs/2210.09261.
11. Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. *Measuring Mathematical Problem Solving With the MATH Dataset*. arXiv, 2021. Available at: https://arxiv.org/abs/2103.03874.
12. David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. *GPQA: A Graduate-Level Google-Proof Q&A Benchmark*. arXiv, 2023. Available at: https://arxiv.org/abs/2311.12022.
13. Zayne Sprague, Xi Ye, Kaj Bostrom, Swarat Chaudhuri, and Greg Durrett. *MuSR: Testing the Limits of Chain-of-thought with Multistep Soft Reasoning*. arXiv, 2024. Available at: https://arxiv.org/abs/2310.16049.
14. Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. *MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark*. arXiv, 2024. Available at: https://arxiv.org/abs/2406.01574.
15. Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. *Open LLM Leaderboard (2023-2024)*. 2023. Published by Hugging Face. Available at: https://huggingface.co/spaces/open-llm-leaderboard-old/open_llm_leaderboard.
16. Manuel Gozzi, Federico Di Maio. *Comparative Analysis of Prompt Strategies for LLMs: Single-Task vs. Multitasking Prompts*. 2024. Available at: https://github.com/gozus19p/llm-benchmark.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.