Exploring Limitations of LLM Capabilities with Multi-Problem Evaluation

Zhengxiang Wang and **Jordan Kodner** and **Owen Rambow**
Department of Linguistics & Institute for Advanced Computational Science
Stony Brook University, Stony Brook, NY, USA
{first.last}@stonybrook.edu

Abstract

We propose using prompts made up of multiple problems to evaluate LLM capabilities, an approach we call *multi-problem evaluation*. We examine 7 LLMs on 4 related task types constructed from 6 existing classification benchmarks. We find that while LLMs can generally perform multiple homogeneous classifications at once (Batch Classification) as well as when they do so separately, they perform significantly worse on two selection tasks that are conceptually equivalent to Batch Classification and involve selecting indices of text falling into each class label, either independently or altogether. We show that such a significant performance drop is due to LLMs' inability to adequately combine index selection with text classification. Such a drop is surprisingly observed across all LLMs attested, under zero-shot, few-shot, and CoT settings, and even with a novel synthetic dataset, potentially reflecting an inherent capability limitation with modern LLMs.

1 Introduction

In recent years, large language models (LLMs) have demonstrated remarkable natural language understanding and reasoning capabilities measured by a wide range of benchmarks (OpenAI, 2023; Beltagy et al., 2020; Gemini-Team, 2023; Anthropic, 2024). However, given their internet-scale training data, there is growing concern over whether LLMs' often superhuman benchmark performance is achieved due to data contamination (Jacovi et al., 2023; Sainz et al., 2023). Several studies (Wu et al., 2024; Mirzadeh et al., 2024) have demonstrated the limitations of LLMs' reasoning capabilities by showing that their performance significantly drops when given the same reasoning tasks but with different assumptions or conditions. These studies are often done through synthetic data generation.

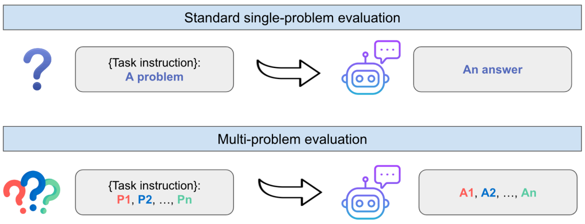

Figure 1: Standard single-problem evaluation versus multi-problem evaluation.

In this study, we explore the limitations of LLM capabilities through *multi-problem evaluation*, a simple evaluation method that leverages existing benchmarks to construct prompts made up of multiple problems. As illustrated in Fig 2, unlike conventional single-problem evaluation that prompts an LLM to solve a single problem at a time, multi-problem evaluation prompts an LLM to solve multiple problems at once in a single input prompt. In this study, we leverage 6 existing classification benchmarks to construct prompts made up of multiple homogeneous problems to form 4 contrastive task types and examine various LLMs on these task types to explore their limitations. Because of the combinatory nature of constructing prompts from multiple problems, it is less likely for LLMs to encounter exact long multi-problem prompts during pre-training, which makes our evaluation less susceptible to data contamination.

We find that while LLMs can typically handle multiple classifications simultaneously (Batch Classification) as well as when performed separately, they exhibit a significant drop in performance on two selection tasks that are conceptually equivalent to Batch Classification and involve selecting indices of text falling into each class label, either independently or altogether. We show that this drop results from LLMs' inability to adequately combine index selection with text classification, which persists across all tested LLMs, under zero-shot, few-shot, or Chain-of-Thought (CoT, Wei et al., 2023) settings, and with a novel synthetic dataset.

The Sixth Workshop on Insights from Negative Results in NLP, pages 121–140
May 4, 2025 ©2025 Association for Computational Linguistics

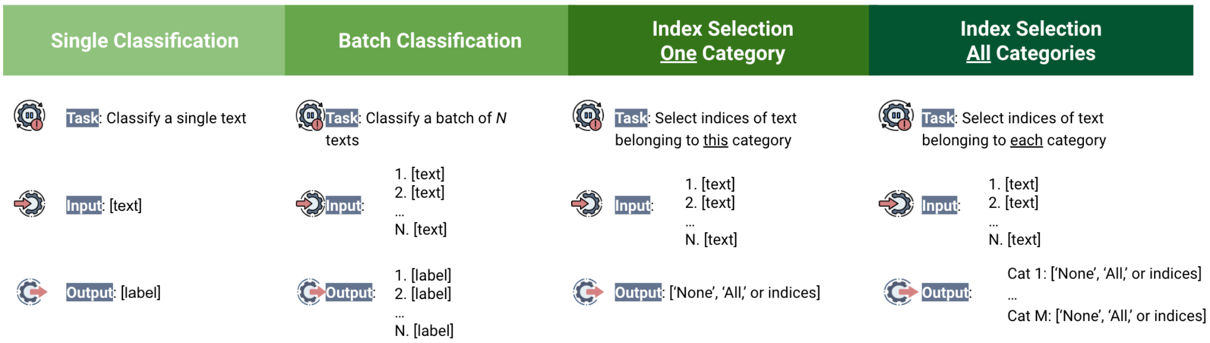
Single Classification
Batch Classification
Index Selection One Category
Index Selection All Categories
Task: Classify a single text
Task: Classify a batch of N texts
Task: Select indices of text belonging to this category
Task: Select indices of text belonging to each category
Input: [text]
Input: 1. [text] 2. [text] ... N. [text]
Input: 1. [text] 2. [text] ... N. [text]
Input: 1. [text] 2. [text] ... N. [text]
Output: [label]
Output: 1. [label] 2. [label] ... N. [label]
Output: ['None', 'All,' or indices]
Output: Cat 1: ['None', 'All,' or indices] ... Cat M: ['None', 'All,' or indices]

Figure 2: The 4 types of evaluation tasks in the form of a <task, input, output> triplet.

Input Format	Benchmark	# Samples	Objective
Single-text	SST-2	1,821	Sentiment analysis
	CoLA	1,043	Grammatical acceptability
	AGNews	1,000	Topic classification
Text-pair	MRPC	1,725	Paraphrase detection
	SNLI	1,000	Natural language inference
	WiC	1,400	Word sense disambiguation

Table 1: Classification benchmarks used in the study. We use the test splits wherever possible, except for CoLA, for which we use the dev split, since the test split is not publicly available. For AGNews and SNLI, we randomly sample 1,000 examples from the test splits.

2 Related Work

Prompting LLMs with multiple problems at once. Given the non-trivial cost of deploying LLMs at large scale, recent studies (Cheng et al., 2023; Laskar et al., 2023; Son et al., 2024; Lin et al., 2024) have proposed various prompt-level approaches that place multiple problems in a single prompt to improve input token utilization to save LLM inference costs. However, they do not aim to discover limitations in LLM capabilities.

LLM Evaluation While there have been several surveys dedidcated to the evaluation of specific topics, e.g., hallucination (Huang et al., 2023; Rawte et al., 2023), bias and fairness (Gallegos et al., 2024; Li et al., 2024), and alignment (Wang et al., 2023; Liu et al., 2024), we note that current LLM evaluation has predominantly focused on LLM's performance on prompts consisting of single problems. Each of such prompts presents a single problem, which expects one specific answer.

3 Experimental Setup

This section describes the data, tasks, LLMs, and performance metric used for our experiments.

3.1 Data

We construct homogeneous multi-problem tasks from existing single-problem benchmarks. We consider the following 6 classification benchmarks, as described in Table 1: SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019), AGNews (Gulli, 2004), MRPC (Dolan and Brockett, 2005), SNLI (Bowman et al., 2015), and WiC (Pilehvar and Camacho-Collados, 2019). They cover two classification paradigms, i.e., single-text and text-pair classification, as well as six distinct task objectives.

3.2 Evaluation Tasks

We conduct our multi-problem evaluation on 4 related types of tasks, conceptualized in Fig 2, using the 6 existing benchmarks introduced above. We define task size n as the number of classification problems included in a prompt and m as the number of unique class labels in a given benchmark. The full prompt templates used for these task types are provided in Appendix C, where the overall limited effect of prompt variation is also discussed.

Among these 4 task types, Single Classification (**SingleClf**) and Batch Classification (**BatchClf**) are classification tasks where an LLM is prompted to solve one or a batch of classification problems at once, respectively. Index Selection One Category (**SelectOne**) and Index Selection All Categories (**SelectAll**) are two reformulations of BatchClf. Instead of making multiple classifications under BatchClf, these two tasks instruct LLMs to select indices of text falling into each label class, either independently in m separate prompts (SelectOne) or altogether in a single prompt (SelectAll).

We design the two selection tasks to test LLM's understanding of the classifications performed under BatchClf. Since selection tasks of size n may have anywhere from 0 to n correct indices per class, spurious correlations are less likely during our eval-

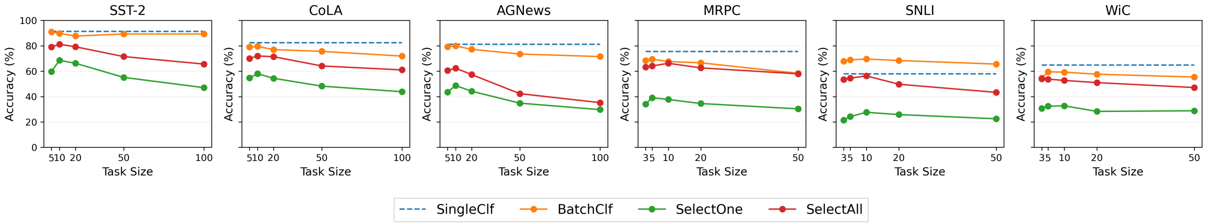

Figure 3: Average accuracy of the 7 LLMs on the 4 task types across task sizes for each benchmark.

uation, given the combinatory answer space.

3.3 LLMs and Evaluation Settings

We evaluate 7 LLMs from 4 model families with greedy decoding: Vicuna (13B, Chiang et al., 2023), Mistral 7B (Jiang et al., 2023), Mixtral 8x7B (Jiang et al., 2024), Llama-3 8B and 70B (Instruct, Meta, 2024), GPT-3.5, and GPT-4 (OpenAI, 2023). We conduct the main experiments under zero-shot and selectively experiment with other prompting strategies in the follow-up experiments in Section 5. See Table 5 in Appendix A for model details.

Performance metric We measure the average per-problem accuracy (PPA) to unify the evaluation across the four task types. PPA, defined in Equation 1, is the average accuracy of classifying n problems with each prompt or, in the case of SelectOne, in each set of directly related prompts targeting different class labels.

$$\text{PPA} = \frac{1}{n}\sum_{i=1}^{n}\delta(I(P_i), A_i) \tag{1}$$

$I(P_i)$ is the inferred LLM-generated answer to the ith problem in the input prompt, A_i is the ground truth, and $\delta(i, j) = 1$ iff $i = j$ and 0 otherwise. For the two index selection tasks, $I(P_i)$ is determined by considering the LLM's assignments of indices to all class labels. Other than assigning an index to a wrong class label, there are two more error types. First, LLMs may assign an index with more than one class label, i.e., an *contradiction* error. Second, LLMs may assign no labels to an index at all, namely, an *non-excluded middle* error.

To compare performance difference, we use Mann-Whitney U tests for significant testing and Cohen's d (Cohen, 1969) for measuring effect size.

4 Results

Fig 3 shows the average accuracy on the 4 task types across task sizes for each benchmark with the related full results for each LLM provided in Appendix A.3. We exclude the results of Vicuna on AGNews at task size 100 as the prompts exceed the model's context length. Two main observations are as follows.

	> 90% SCAcc	> 80% SCAcc	> 75% SCAcc
Vicuna 13B	79.3	93.1	93.1
Mistral 7B	76.7	83.3	100.0
Mixtral 8x7B	63.3	83.3	86.7
Llama-3 8B	73.3	90.0	100.0
Llama-3 70B	80.0	100.0	100.0
GPT-3.5	56.7	83.3	90.0
GPT-4	100.0	100.0	100.0
Overall	75.6	90.4	95.7

Table 2: Percent of time that BatchClf performance surpasses a threshold percent of SingleClf accuracy (SCAcc) across benchmarks.

	BatchClf vs SelectOne	BatchClf vs SelectAll	SelectOne vs SelectAll
Mean Acc Dif	32.0	12.1	-19.9
Std Dev	16.9	15.3	12.0
Cohen's d	1.8	0.8	-1.0

Table 3: Pairwise accuracy difference (x vs y = x - y). All the differences are statistically significant and with a large effect size (| Cohen's d | ≥ 0.8).

LLMs can handle multiple classifications at once with minimal performance loss. Although BatchClf accuracy generally declines as the task size increases, all LLMs achieve accuracy of at least 90% that of SingleClf across benchmarks most of the time (see Table 2). Overall, the SingleClf accuracy for the 7 LLMs is 75.5% on average and the BatchClf counterpart is 72.3%, a minor 3.2% absolute drop from the former.

LLMs perform significantly worse on the selection tasks. Despite the impressive BatchClf performance, LLMs nearly always perform much worse in SelectOne and SelectAll than BatchClf, even when the task size is just 3 or 5. The overall discrepancy in accuracy between BatchClf and the two tasks is large and statistically significant (32.0% for SelectOne and 12.1% for SelectAll, see Table 3) and generally increases with a larger task

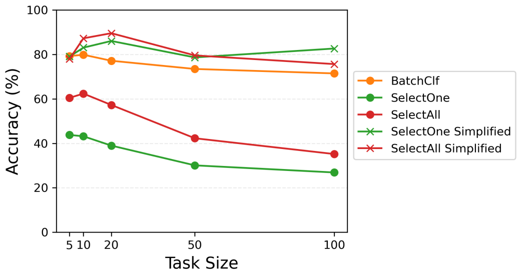

Figure 4: Average accuracy of the 7 LLMs on the two simplified index selection tasks based on AGNews. We provide the original results from BatchClf, SelectOne, and SelectAll for easy comparisons.

size. Similarly, Table 3 also shows a large and statistically significant difference between SelectOne and SelectAll, in favor of the latter.

These significant performance differences may not be human-like, given the conceptual equivalence of the three tasks underlyingly. For example, humans should at least be able to classify and select a small number (e.g., 3/5) of texts (mostly short sentences) equally well simply by thinking over the problems (i.e., zero-shot).

5 Follow-up Experiments

We perform a series of follow-up experiments to further understand and validate our main findings. More details about these experiments are presented in Appendix B.

5.1 Can LLMs do Index Selection?

While we argue that the LLMs' weaker performance on SelectOne and SelectAll is due to a weakness in combining classification with index selection, an alternative explanation would be that they just generally struggle at index selection. To exclude this possibility, we simplify the two index selection tasks by directly replacing each line of text with its gold standard label in the prompts for AGNews, which has most labels. We then ask the LLMs to select the indices of lines containing each label with minimal modifications to the original task instructions.

Fig 4 shows that LLMs can perform in the simplified selection tasks much better than the original ones, with Llama-3 70B, GPT-3.5, and GPT-4 even achieving (nearly) 100% accuracy across task sizes. The 7 LLMs' overall performance in the simplified selection tasks is even slightly higher than their overall BatchClf accuracy. The previously observed performance gap between SelectOne and SelectAll also disappears for almost all LLMs (see Appendix B for full results). In general, LLMs can indeed do index selection.

Task	SelectOne	+ CoT	SelectAll	+ CoT	BatchClf
Model					
Vicuna	23.0	25.8	53.2	57.6	68.7
Mistral 7B	38.3	47.5	56.8	60.5	71.2
Mixtral 8x7B	47.4	38.7	65.1	58.4	73.4
Llama-3 8B	39.0	41.8	62.4	59.0	73.3
Llama-3 70B	59.4	67.1	72.9	79.2	79.4
GPT-3.5	45.5	47.5	66.7	66.3	71.9
GPT-4	66.3	71.8	78.8	81.8	81.9
Overall	45.5	48.6	65.1	66.1	74.3

Table 4: Aggregate average accuracy of SelectOne and SelectAll with and without 1-shot-CoT for each LLM. BatchClf performance is also provided for comparisons.

In conclusion, it appears that the tested LLMs perform less well on SelectOne and SelectAll as compared to BatchClf because they cannot adequately combine the index selection task and the classification tasks in response to a zero-shot prompt. Put differently, LLMs lack true understanding of the problems presented in different forms, even when the number of problems is quite small (e.g., 3 or 5), which may not be human-like.

5.2 Does CoT Help?

In light of the results above, we use 1-shot-CoT to prompt LLMs to do BatchClf first and then perform the two index selection tasks on the 6 benchmarks with a fixed task size 10. Table 4 shows that while overall LLMs benefit from CoT for both SelectOne and SelectAll, the benefits are generally larger for SelectOne than SelectAll (3.1% versus 1.0% improvement) and not consistent across models with Mixtral 8x7B performing even worse with CoT. Moreover, the task complexity hierarchy among BatchClf, SelectOne, and SelectAll remains in virtually all cases. We leave to future studies the investigation of how CoT may further help with the two selection tasks.

5.3 Does Few-Shot Prompting Help?

To test the generality of the main findings, we first re-run all experiments for CoLA providing 2 exemplars in the prompts and with a fixed task size 5 for tasks other than SingleClf. We find that few-shot prompting is mostly detrimental across LLMs, particularly so for SelectOne and BatchClf. As a result, SelectAll shows an overall better performance than BatchClf and the performance gap between SelectOne and SelectAll becomes much larger. Full

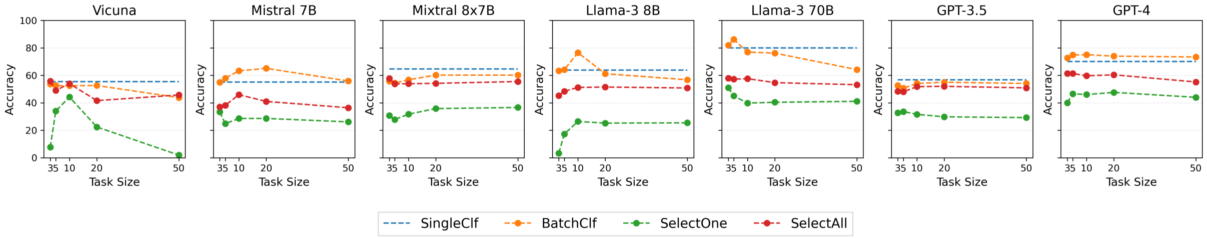

Figure 5: Full results on the novel benchmark created on top of SST-2 where the task objective is to decide if a text pair shares the same sentiment.

results are in Table 8 in Appendix B for space reasons.

5.4 Testing on a Novel Benchmark

As argued in Section 1, multi-problem evaluation is less susceptible to data contamination, given the combinatory nature of constructing prompts from multiple problems. To further mitigate the concern of data contamination, we create a novel benchmark with 1,000 distinct and label-balanced text pairs sampled from SST-2. The task is to determine if each text pair shares the same sentiment, which we believe is unlikely to appear in the training data of LLMs. We run experiments on this benchmark using the same experimental setup described in Section 3 for the text-pair benchmarks. The results in Fig 5 are consistent with our main results.

6 Conclusion

We propose multi-problem evaluation and present a comprehensive multi-problem evaluation of LLMs, leveraging 6 existing classification benchmarks and 4 related task types constructed from those benchmarks. Our results provide new insights into the multiple problem handling capabilities of LLMs: LLMs are competent multi-problem solvers for multiple homogeneous classification problems (Batch Classification), but they perform significantly worse on two selection tasks that are conceptually equivalent to Batch Classification and involve selecting indices of text falling into each class label, either independently or altogether. This is due to their inability to adequately combine index selection with text classification. The surprisingly consistent performance drop on the two selection tasks observed across 7 LLMs and a wide range of evaluation settings potentially indicates an inherent limitation with modern LLM capabilities. This also showcases the potential of multi-problem evaluation as a useful and effective in discovering limitations of LLM capabilities.

There are several directions worth future explorations. For example, to better understand how well how LLMs can handle multiple problems in general, it is important to test LLMs with other types of problems (e.g., reasoning problems) and with multiple heterogeneous problems (e.g., mixing different benchmarks/tasks). It is also important to understand what causes LLMs to perform worse or better when prompted with multiple problems, such as benchmark, task size, and model's context length. In particular, model-level ablation studies are needed if we want to know how LLMs obtain the ability to handle multiple problems at once and how to improve their understanding capabilities.

Acknowledgment

We are grateful for the supports from the Institute for Advanced Computational Science (IACS) at Stony Brook University, in particular the free GPT access it provides. Zhengxiang Wang is supported by IACS's Junior Researcher Award since Fall 2024. We thank three anonymous reviewers from the Workshop on Insights From Negative Results 2025 for their helpful comments. The paper was presented at All Things Language And Computation (ATLAC) organized by the NLP reading group at Stony Brook University. We also thank the audience there for their discussions and feedback about the paper.

Limitations

While we have done our best to make our experiments as comprehensive as possible, there is always more work that can be done in a large comparison study, including the addition of even more benchmarks and language models. We were limited by cost, time, and space and have attempted to select an informative and representative sample of experiments. Since we used pre-existing benchmarking data sets, we inherit any labeling errors that they may contain. Finally, despite our efforts to compare different prompts, as is the case with all prompt-based LLM studies, we cannot guarantee that slight differences in the prompts would not meaningfully alter the results.

Ethical Concerns

To the best of our knowledge, all results published in this paper are accurate. All data sources are free, publicly available, and cited in the article. No sensitive data was used which could violate individuals' privacy or confidentiality.

References

Anthropic. 2024. The Claude 3 model family: Opus, Sonnet, Haiku.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Zhoujun Cheng, Jungo Kasai, and Tao Yu. 2023. Batch prompting: Efficient inference with large language model APIs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 792–810, Singapore. Association for Computational Linguistics.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing GPT-4 with 90%* chatgpt quality.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

J. Cohen. 1969. *Statistical Power Analysis for the Behavioral Sciences*. Academic Press.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2024. Bias and fairness in large language models: A survey.

Gemini-Team. 2023. Gemini: A family of highly capable multimodal models.

Antonio Gulli. 2004. AG's corpus of news articles.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions.

Alon Jacovi, Avi Caciularu, Omer Goldman, and Yoav Goldberg. 2023. Stop uploading test data in plain text: Practical strategies for mitigating data contamination by evaluation benchmarks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5084, Singapore. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts.

Md Tahmid Rahman Laskar, M Saiful Bari, Mizanur Rahman, Md Amran Hossen Bhuiyan, Shafiq Joty, and Jimmy Huang. 2023. A systematic study and comprehensive evaluation of ChatGPT on benchmark datasets. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 431–469, Toronto, Canada. Association for Computational Linguistics.

Yingji Li, Mengnan Du, Rui Song, Xin Wang, and Ying Wang. 2024. A survey on fairness in large language models.

Jianzhe Lin, Maurice Diesendruck, Liang Du, and Robin Abraham. 2024. Batchprompt: Accomplish more with less.

Yang Liu, Yuanshun Yao, Jean-Francois Ton, Xiaoying Zhang, Ruocheng Guo, Hao Cheng, Yegor Klochkov, Muhammad Faaiz Taufiq, and Hang Li. 2024. Trustworthy LLMs: a survey and guideline for evaluating large language models' alignment.

Meta. 2024. The llama 3 herd of models.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models.

OpenAI. 2023. GPT-4 technical report.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Vipula Rawte, Amit Sheth, and Amitava Das. 2023. A survey of hallucination in large foundation models.

Oscar Sainz, Jon Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10776–10787, Singapore. Association for Computational Linguistics.

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.

Guijin Son, SangWon Baek, Sangdae Nam, Ilgyun Jeong, and Seungone Kim. 2024. Multi-task inference: Can large language models follow multiple instructions at once? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5606–5627, Bangkok, Thailand. Association for Computational Linguistics.

Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. 2023. Aligning large language models with human: A survey.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

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.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2024. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1819–1862, Mexico City, Mexico. Association for Computational Linguistics.

A Experimental details

A.1 LLM Details

Table 5 describes the specific versions for the 7 LLMs we use in the paper and highlights their differences in terms of open weights, training with Reinforment Learning from Human Feedback or RLHF (Christiano et al., 2017), architecture, number of parameters, and context window size. We use OpenAI API and TogetherAI API to call GPT LLMs and non-GPT LLMs, respectively.

A.2 LLM Output Parsing Code

The LLM output parsing code contains over 200 lines of regular expressions to parse LLM outputs and we include it in this submission for review. The code takes into accounts the following five variables during parsing: task type, benchmark, model, task size, and target label (for SelectOne). This is to ensure our code can handle cases when LLMs generate undefined labels, un-instructed explanations, or even wrong answer format (e.g., a non-JSON output for SelectOne and SelectAll).

Overall, our code achieves about 99.9% overall parsing rate. The unparsable cases mostly come from SingleClf outputs where LLMs output undefined labels, such as “Mixed” or “Netual” for SST-2. For BatchClf outputs, we implement a series of rules to extract both defined and undefined labels, because of the order of the extracted labels affects the final evaluation. For SelectOne and SelectAll, our code extracts JSON object, fixes cases where the JSON object has formatting issues, or extracts a series of text indices when there is no JSON object identified in the output.

A.3 Full Results

The full results obtained from the main experiments are visualized in Fig 6. We exclude the results of Vicuna on AGNews when task size is 100 because the prompts exceed the model’s context length.

A.4 Task Complexity Hierarchy

In Table 3 from Section 4, we demonstrate the overall task complexity hierarchy among BatchClf,

Model	Version	Open Weights	With RLHF	MoE	# Parameter	Context Window Size
Vicuna (Chiang et al., 2023)	v1.5	✓	✗	✗	13B	4,096
Mistral 7B (Jiang et al., 2023)	Instruct-v0.2	✓	✗	✗	7B	8,192
Mixtral 8x7B (Jiang et al., 2024)	Instruct-v0.1	✓	✗	✓	47B	8,192
Llama-3 8B (Meta, 2024)	Instruct	✓	✓	✗	8B	8,192
Llama-3 70B (Meta, 2024)	Instruct	✓	✓	✗	70B	8,192
GPT-3.5	turbo-0125	✗	✓	✗	-	16,385
GPT-4 (OpenAI, 2023)	turbo-2024-04-09	✗	✓	✗	-	128,000

Table 5: The 7 LLMs we use, all instruction finetuned. For the two GPT LLMs, it is commonly assumed that both are larger than GPT-3 (a 175B LLM) with GPT-4 being the largest. For Mixtral 8x7B, a Mixture of Experts (MoE) LLM, although each token has access to 47B parameters, but only uses 13B active parameters during inference.

SelectOne, and SelectAll. It is found that SelectOne > SelectAll > BatchClf, where ">" denotes a "more complex than" relationship.

Similar to Fig 3, which describes the aggregate average accuracy across benchmarks, Fig 7 describes the aggregate average accuracy across LLMs. Although there are few cases shown in Fig 6 where the overall complexity hierarchy among BatchClf, SelectOne, and SelectAll does not hold (e.g., Vicuna on MRPC for task size 50), these cases are exceptional and likely due to the interactions of multiple factors in play, such as model, benchmark, input length, and context windows. Nevertheless, the overall task complexity hierarchy is clear both in Fig 3 (benchmark-level) and in Fig 7 (model-level).

A.5 Limited Effect of Prompt Variations

Throughout our research project, we have also tried prompts with different wordings and structures until we finally unified the prompt designs presented above. More concretely, we initially instructed LLMs to produce indices line by line for SelectOne and did not include any formatted output example in SelectAll prompts for almost all classification-related experiments. The table below shows the average performance of each LLM on SelectAll and SelectOne using current and earlier prompt templates. Clearly, the effects of prompt variations are minimal (except for GPT-3.5 on SelectOne) and the findings in the paper remain valid.

task model	SelectOne (Early)	SelectAll (Early)
Vicuna	17.9 (15.4)	43.2 (44.5)
Mistral 7B	31.5 (31.8)	51.5 (44.2)
Mixtral 8x7B	38.4 (39.1)	60.1 (60.5)
Llama-3 8B	33.0 (34.5)	54.6 (56.8)
Llama-3 70B	54.2 (54.7)	70.0 (70.4)
GPT-3.5	41.4 (28.3)	63.4 (61.3)
GPT-4	65.4 (64.7)	78.0 (76.7)
Overall	40.4 (38.5)	60.2 (59.2)

Table 6: Model performance on the two selecting tasks using the finalized prompts and early prompts.

B Follow-up Experiments

B.1 Simplified Index Selection Tasks

Fig 8 shows the full results for the two simplified index selection tasks, along with results for the two original selection tasks and BatchClf, based on AGNews. Clearly, (1) all LLMs perform much better in the simplified tasks than the original ones; (2) Mixtral 8X7B, Llama-3 70B, GPT-3.5, and GPT-4 can do these simplified selection tasks even better than their BatchClf performance consistently, with the latter three achieving nearly 100% accuracy in most cases; (3) the task complexity hierarchy between SelectOne and SelectAll nearly disappears for all LLMs except Vicua, implying that it may be challenging for Vicuna to perform index selection tasks in general.

B.2 1-shot-CoT Results

Table 7 presents the full 1-shot-CoT results across benchmarks and LLMs with a fixed task size 10. As demonstrated in Section 5, CoT improves the LLMs' performance in SelectOne and SelectAll with an overall larger positive effect on the former, but the improvement is not consistent both across benchmarks and across LLMs. For examples, LLMs tend to benefit from CoT for both selection tasks constructed from SST-2, CoLA, and SNLI (see the "Overall" results in Table 7).

B.3 Few-shot Results

Table 8 shows the full 2-shot results on CoLA for each LLM, which shows a general negative effect of few-shot prompting. According to these results, LLMs performs much worse in SelectOne with an overall accuracy going down to 25.2% from 58.1%. Similarly, the negative effect is larger on BatchClf than on SelectAll, making the overall BatchClf accuracy lower than SelectAll, although the former

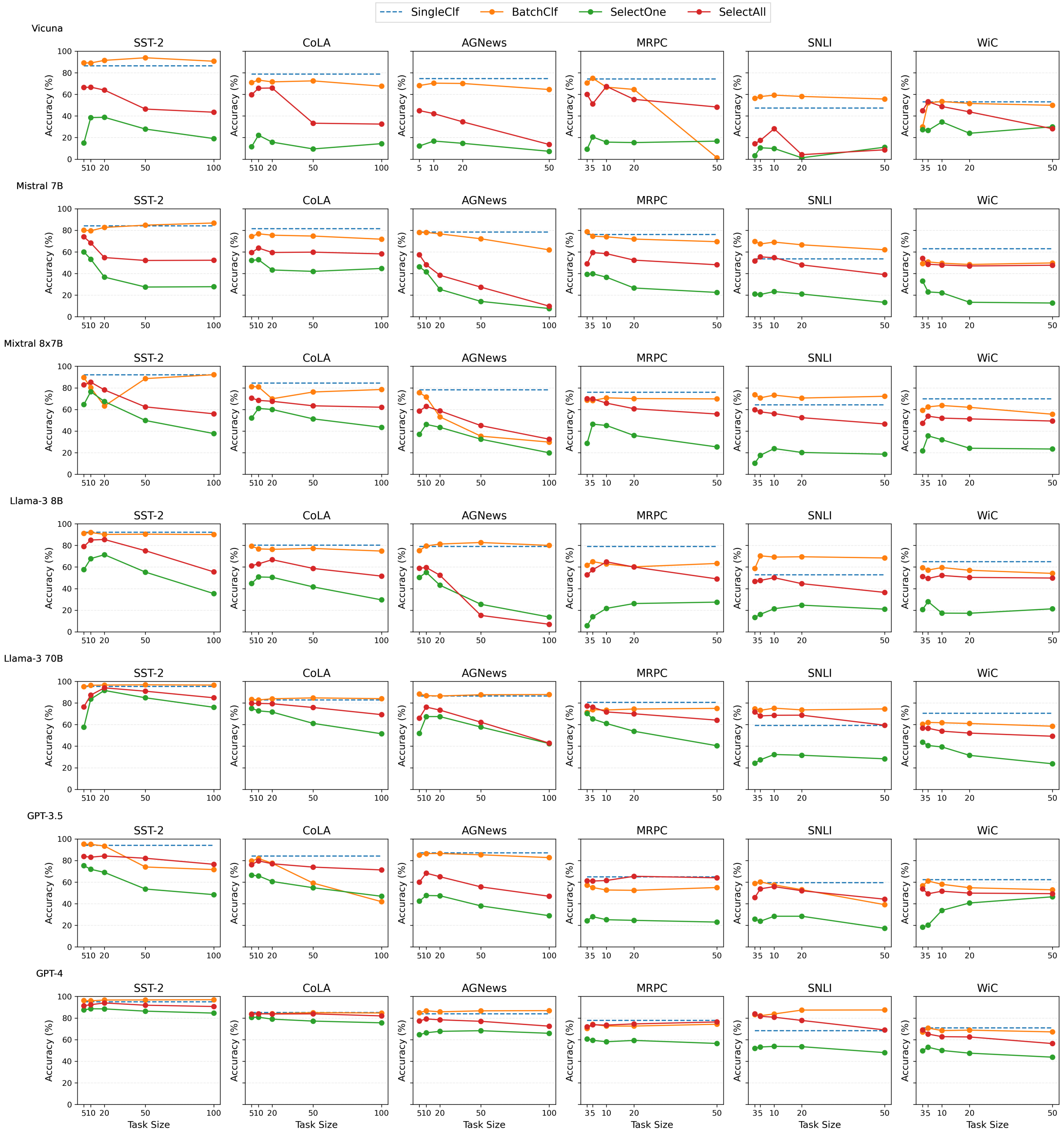

Figure 6: Full average accuracy of the 7 LLMs on the 4 tasks across the 6 benchmarks.

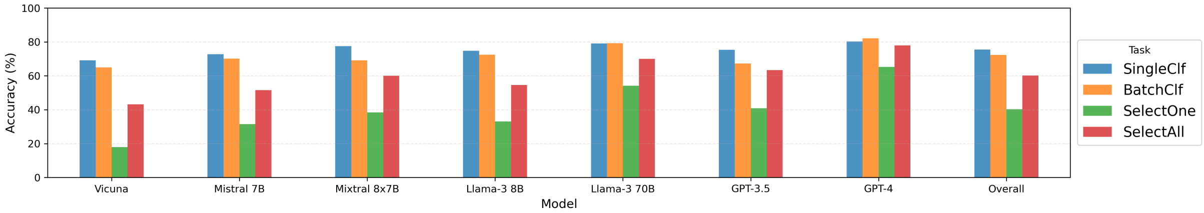

Figure 7: Aggregate average accuracy on the 4 tasks averaging over results from the 6 benchmarks for each LLM. "Overall" presents the average results across all LLMs for each one of the 4 task.

is still often higher than the latter under the same test conditions.

C Full Prompts

C.1 Prompt Templates for SingleClf, BatchClf, SelectOne, and SelectAll

Tables 13 to 14 show the complete prompt templates for the four task types (i.e., SingleClf, BatchClf, SelectOne, and SelectAll) tailored for SST-2, CoLA, AGNews, MPRC, SNLI, and WiC, respectively. While there are differences in the exact wording of a prompt template for each task type across the 6 classification benchmarks, each prompt template type shares a similar underlying structure and can be easily applied to other classification benchmarks.

Throughout our research project, we have also tried prompts with different wordings and structures until we finally unified the prompt designs presented above. For example, we initially asked LLMs to directly generate indices line by line instead of a JSON output for SelectOne and we did not provide any formatted example for SelectAll. We also put the output format instruction in the end of each prompt for SelectAll, instead of in the beginning. Although we observed certain task-level performance variations, which are expected, the overall complexity among the 4 task types (SelectOne > SelectAll > BatchClf > SingeClf) remains unchanged, despite the variations in the prompts. This indicates the overall limited effects of rewording and restructuring prompts.

C.2 Prompt Templates for the Two Index Selection Tasks

Table 15 presents the two prompt templates used for the two simplified index selection tasks based on AGNews.

C.3 One-Shot-CoT Prompt Templates for the Two Index Selection Tasks

To construct a 1-shot-CoT exemplar, we first randomly sampled 10 examples from each benchmark that do not appear in our test examples. We then put these 10 examples into the prompt template for SelectOne or SelectAll, followed by a constructed answer. The answer first explicitly classifies the 10 examples into the corresponding class labels, and then perform index selection based on the generated class labels.

Table 16 shows the complete answer templates for SelectOne and SelectAll for each classification benchmark.

	Benchmark	SST-2	CoLA	AGNews	MRPC	SNLI	WiC
Model	Task						
Vicuna	SelectOne	38.5	22.2	16.8	15.8	9.9	34.5
	+ CoT	52.6	53.9	12.1	12.1	11.5	12.5
	SelectAll	66.7	65.7	42.2	67.4	28.2	48.8
	+ CoT	81.7	69.2	40.9	63.9	40.9	48.9
	BatchClf	89.1	73.2	70.4	66.7	59.3	53.4
Mistral 7B	SelectOne	53.3	52.8	41.5	36.6	23.3	22.2
	+ CoT	66.5	55.1	44.8	59.4	29.5	29.5
	SelectAll	68.3	63.6	48.2	58.4	54.7	47.9
	+ CoT	74.4	68.2	57.0	68.5	47.6	47.3
	BatchClf	79.6	76.8	78.1	74.0	69.1	49.6
Mixtral 8x7B	SelectOne	76.5	61.0	46.2	45.2	23.8	32.0
	+ CoT	41.5	67.3	12.9	48.4	25.1	37.2
	SelectAll	85.3	68.4	62.9	65.9	56.2	52.0
	+ CoT	49.0	67.4	57.8	66.5	58.9	51.1
	BatchClf	80.2	80.9	71.6	70.8	73.3	63.7
Llama-3 8B	SelectOne	67.8	50.8	55.1	21.7	21.4	17.3
	+ CoT	85.0	48.9	22.1	50.7	22.3	22.0
	SelectAll	85.0	62.9	59.6	64.8	50.1	52.2
	+ CoT	85.8	66.5	45.3	64.5	39.8	52.3
	BatchClf	92.0	76.8	79.5	62.8	69.2	59.5
Llama-3 70B	SelectOne	83.6	72.7	67.4	61.0	32.3	39.3
	+ CoT	94.7	79.7	75.2	67.9	69.4	15.8
	SelectAll	87.3	79.6	76.2	71.7	68.6	53.9
	+ CoT	96.1	83.7	85.5	73.9	77.3	58.4
	BatchClf	96.4	82.8	86.8	73.5	75.2	61.7
GPT-3.5	SelectOne	71.9	65.8	47.6	25.2	28.4	33.8
	+ CoT	80.2	71.7	57.9	26.8	26.5	22.1
	SelectAll	83.1	79.7	68.3	61.5	55.8	51.6
	+ CoT	89.5	77.5	58.8	70.2	50.5	51.2
	BatchClf	95.0	81.8	86.4	52.7	57.4	58.0
GPT-4	SelectOne	88.6	80.8	66.4	58.1	53.8	50.0
	+ CoT	84.5	81.9	67.8	60.1	77.0	59.6
	SelectAll	92.4	84.0	79.2	73.5	80.8	62.8
	+ CoT	91.3	83.7	85.4	76.2	83.3	71.0
	BatchClf	96.0	84.0	86.7	72.6	83.8	68.4
Overall	SelectOne	68.6	58.0	48.7	37.7	27.6	32.7
	+ CoT	72.1	65.5	41.8	46.5	37.3	28.4
	SelectAll	81.2	72.0	62.4	66.2	56.3	52.7
	+ CoT	81.1	73.7	61.5	69.1	56.9	54.3
	BatchClf	89.8	79.5	79.9	67.6	69.6	59.2

Table 7: Full 1-shot-CoT results for all benchmarks and LLMS. The task size is fixed at 10.

Model	Task Benchmark	SingleClf	BatchClf	SelectOne	SelectAll
Vicuna	CoLA	78.8	71.0	48.6	59.6
	+ 2-shot	69.0	50.4	14.8	64.0
Mistral 7B	CoLA	81.5	74.4	47.4	59.4
	+ 2-shot	80.2	65.0	25.0	60.4
Mixtral 8x7B	CoLA	84.4	81.2	65.0	70.6
	+ 2-shot	81.8	73.2	19.8	72.4
Llama-3 8B	CoLA	80.2	79.4	33.4	61.2
	+ 2-shot	74.3	46.6	0.6	63.8
Llama-3 70B	CoLA	82.8	83.4	71.2	79.6
	+ 2-shot	81.8	82.0	29.2	80.8
GPT-3.5	CoLA	84.2	79.6	63.4	76.2
	+ 2-shot	79.5	76.8	38.2	72.8
GPT-4	CoLA	85.1	83.8	77.8	83.6
	+ 2-shot	85.7	80.8	48.6	81.0
Overall	CoLA	82.4	79.0	58.1	70.0
	+ 2-shot	78.9	67.8	25.2	70.7

Table 8: Full 2-shot results on CoLA. We use a fixed task size 5 for tasks other than SingleClf, whose task size is 1 by default. We provide the related zero-shot results for easy comparisons.

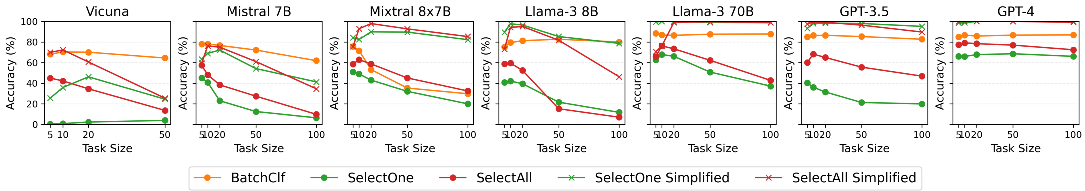

Figure 8: Full results for the two simplified index selection tasks versus the original tasks based on AGNews.

Task	Prompt template
SingleClf	Indicate the sentiment for the following line of text. The sentiment shall be either 'Positive' or 'Negative.' Text: $text Sentiment:
BatchClf	Indicate the sentiment for each of the $num following lines of text. The sentiment shall be either 'Positive' or 'Negative.' Texts, one per line: $texts The sentiments for each of the $num lines of text, one per line:
SelectOne	Go over the $num lines of text below and list the index numbers of the lines with $polarity sentiment according to the following instructions: If none of the texts show $polarity sentiment, write 'None.' If all the texts show $polarity sentiment, write 'All.' Otherwise, provide the index numbers for each text with $polarity sentiment. Output your responses in JSON format with the key '$polarity'. A formatted example output is provided below. {'$polarity': [None/All or index numbers for the texts with $polarity sentiment]} Texts, one per line: $texts JSON output:
SelectAll	Go over the $num lines of text below. First, list the index numbers of the lines with positive sentiment. Then, list the index numbers of the lines with negative sentiment. If none of the texts show a particular sentiment, write 'None.' If all the texts show a particular sentiment, write 'All.' Otherwise, provide the index numbers of the texts that fit a particular category. Output your responses in JSON format with two keys: 'positive' and 'negative.' A formatted example output is provided below. {'positive': [None/All or index numbers of positive sentences], 'negative': [None/All or index numbers of negative sentences]} Texts, one per line: $texts JSON output:

Table 9: Prompt templates for SST-2. Words immediately preceded by the dollar sign $ are placeholders. For the single-text classification task (SST-2, CoLA, AGNews), the sequence of texts in the place of '$texts' are indexed starting with '1' and each text is separated by a newline.

Task	Prompt template
SingleClf	Indicate the grammatical acceptability for the following line of text. The acceptability shall be either ‘Acceptable’ or ‘Unacceptable.’ Text: $text Grammatical acceptability:
BatchClf	Indicate the grammatical acceptabilities for each of the $num following lines of text. The acceptability shall be either ‘Acceptable’ or ‘Unacceptable.’ Texts, one per line: $texts Grammatical acceptabilities for each of the $num lines of text, one per line:
SelectOne	Go over the $num lines of text below and list the index numbers of the lines that are grammatically $acceptability according to the following instructions: If none of the texts are grammatically $acceptability, write ‘None.’ If all the texts are grammatically $acceptability, write ‘All.’ Otherwise, provide the index numbers for each grammatically $acceptability text. Output your responses in JSON format with the key ‘$acceptability’. A formatted example output is provided below. {‘$acceptability’: [None/All or index numbers of $acceptability sentences]} Texts, one per line: $texts JSON output:
SelectAll	Go over the $num lines of text below. First, list the index numbers of the lines that are grammatically acceptable. Then, list the index numbers of the lines that are grammatically unacceptable. If none of the sentences show a particular acceptability, write ‘None.’ If all the sentences show a particular acceptability, write ‘All.’ Otherwise, provide the index numbers of the texts that fit a particular category. Output your responses in JSON format with two keys ‘acceptable’ and ‘unacceptable.’ A formatted example output is provided below. {‘acceptable’: [None/All or index numbers of acceptable texts], ‘unacceptable’: [None/All or index numbers of unacceptable texts]} Texts, one per line: $texts JSON output:

Table 10: Prompt templates for CoLA.

Task	Prompt template
SingleClf	Classify which news category the following line of text belongs to among the following four categories: ‘Business,’ ‘Sports,’ ‘World,’ and ‘Sci/Tech.’ Text: $text News category:
BatchClf	Classify which news category each of the $num following lines of text belongs to among the following four categories: ‘Business,’ ‘Sports,’ ‘World,’ and ‘Sci/Tech.’ Texts, one per line: $texts News categories for each of the $num lines of text, one per line:
SelectOne	This is a news classification task in which each line of text belongs to one of four categories ‘Business,’ ‘Sports,’ ‘World,’ and ‘Sci/Tech.’ Go over the $num lines of text below and list the index numbers of the lines that can be classified as $category according to the following instructions: If none of the texts can be classified as $category, write ‘None.’ If all the texts can be classified as $category, write ‘All.’ Otherwise, provide the index numbers of the texts that can be classified as $category. Output your responses in JSON format with the key ‘$category’. A formatted example output is provided below. {‘$category’: [None/All or index numbers of the texts that can be classified as $category]} Texts, one per line: $texts JSON output:
SelectAll	This is a news classification task in which each line of text belongs to one of four categories ‘Business,’ ‘Sports,’ ‘World,’ and ‘Sci/Tech.’ Go over the $num lines of text below and list the index numbers of the lines that belong to each category according to the following instructions: If none of the texts can be classified as a particular category, write ‘None.’ If all the texts can be classified as a particular category, write ‘All.’ Otherwise, provide the index numbers of the texts that can be classified as the category. Output your responses in JSON format with the following keys: ‘business,’ ‘sports,’ ‘world,’ and ‘sci/tech.’ A formatted example output is provided below. {‘business’: [None/All or index numbers of texts in ‘business’ category], ‘sports’: [None/All or index numbers of texts in ‘sports’ category], ‘world’: [None/All or index numbers of texts in ‘world’ category], ‘sci/tech’: [None/All or index numbers of texts in sci/tech category]} Texts, one per line: $texts JSON output:

Table 11: Prompt templates for AGNews.

Task	Prompt template
SingleClf	Compare text A with text B and determine if text A is a paraphrase of text B. Respond with ‘Yes’ if text A is a paraphrase, and ‘No’ if it is not. $text Answer:
BatchClf	Compare text A with text B for the following $num text pairs and determine if text A is a paraphrase of text B line by line. Respond with ‘Yes’ if text A is a paraphrase, and ‘No’ if it is not. Provide your answers line by line. $texts Answers:
SelectOne	Go over the $num text pairs below and list the index numbers of the text pairs where text A $be a paraphrase of text B according to the following instructions: If none of the text pairs satisfy this condition, write ‘None.’ If all the text pairs satisfy this condition, write ‘All.’ Otherwise, provide the index numbers of the text pairs where text A $be a paraphrase of text B. Output your responses in JSON format with the key ‘answer’. A formatted example output is provided below. {‘answer’: [None/All or index numbers of the text pairs where text A $be a paraphrase of text B]} Here are the text pairs: $texts JSON output:
SelectAll	Go over the $num text pairs below. First, list the index numbers of the text pairs that contain paraphrases. Then, list the index numbers of the text pairs that contain non-paraphrases. If none of the text pairs satisfy a condition, write ‘None.’ If all the text pairs satisfy a condition, write ‘All.’ Otherwise, provide the index numbers of the text pairs that satisfy each condition. Output your responses in JSON format with two keys: ‘yes’ for paraphrases and ‘no’ for non-paraphrases. A formatted example output is provided below. {‘yes’: [None/All or index numbers of text pairs that contain paraphrases], ‘no’: [None/All or index numbers of text pairs that contain non-paraphrases]} Here are the text pairs: $texts JSON output:

Table 12: Prompt templates for MRPC. For the text-pair classification task (MRPC, SNLI, WiC), the sequence of text pairs in the place of ‘$texts’ are indexed starting with ‘1’ and each text pair is separated by two newlines (each text pair ends with a newline be design, followed by another newline before the next text pair).

Task	Prompt template
SingleClf	Given the following premise and hypothesis, determine the inference relation between them. Respond with ‘Entailment’ if the hypothesis logically follows from the premise, ‘Contradiction’ if they are in direct opposition, and ‘Neutral’ if neither applies. $text Inference relation:
BatchClf	Given the following $num pairs of premises and hypotheses, determine the inference relation for each pair line by line. Respond with ‘Entailment’ if the hypothesis entails the premise, and ‘Contradiction’ if they contradict. If neither is the case, respond with ‘Neutral.’ Provide your answers line by line. $texts Inference relations for the $num text pairs provided above:
SelectOne	Go over the $num text pairs below and list the index numbers of the text pairs where the inference relation between the premise and the hypothesis is $relationship according to the following instructions: If none of the text pairs contain $relationship inference relation, write ‘None.’ If all text pairs contain $relationship inference relation, write ‘All.’ Otherwise, provide the index numbers of the text pairs where the inference relation between the premise and the hypothesis is $relationship. Output your responses in JSON format with the key ‘$relationship’. A formatted example output is provided below. ‘$relationship’: [None/All or index numbers of text pairs that contain $relationship inference relation] Here are the text pairs: $texts JSON output:
SelectAll	Go over the $num text pairs below. First, list the index numbers of the text pairs that contain entailment inference relation. Then, select all text pairs that contain contradiction inference relation. Finally, select all text pairs that contain neutral inference relation. If none of the text pairs satisfy a condition, write ‘None.’ If all the text pairs belong satisfy a condition, write ‘All.’ Otherwise, provide the index numbers of the text pairs that satisfy each condition. Output your responses in JSON format with three keys: ‘entailment’, ‘contradiction’, and ‘neutral’. A formatted example output is provided below. {‘entailment’: [None/All or index numbers of text pairs that contain entailment inference relation], ‘contradiction’: [None/All or index numbers of text pairs that contain contradiction inference relation], ‘neutral’: [None/All or index numbers of text pairs that contain neutral inference relation]} Here are the text pairs: $texts JSON output:

Table 13: Prompt templates for SNLI.

Task	Prompt template
SingleClf	Analyze the usage of the given target word in the two subsequent contexts. The target word may appear in various grammatical forms in each context. Respond with ‘Yes’ if it maintains the same meaning across both contexts, and ‘No’ if it does not. $text Answer:
BatchClf	Analyze the usage of the following $num target words in the two contexts that immediately follow them. These target words may appear in different grammatical forms across the two subsequent contexts. Determine if each target word maintains the same meaning in the two subsequent contexts. Provide your answers line by line, indicating ‘Yes’ if it does and ‘No’ if it does not. $texts Answers:
SelectOne	Analyze the following $num target words and determine the index numbers of the target words where the same meaning $be maintained across the two contexts that immediately follow them. These target words may appear in different grammatical forms in each context. If none of the target words satisfy this condition, write ‘None.’. If all the target words satisfy this condition, write ‘All.’ Otherwise, provide the index numbers. Output your responses in JSON format with the key ‘answer’. A formatted example output is provided below. {‘answer’: [None/All or index numbers of the target words where the same meaning $be maintained in the two subsequent contexts]} Here are the target words along with their contexts: $texts JSON output:
SelectAll	Analyze the following $num target words, which may appear in different grammatical forms in the two subsequent contexts. First, list the index numbers of target words that maintain the same meaning in the two subsequent contexts. Then, list the index numbers of target words that do not maintain the same meaning in the two subsequent contexts. If none of the target words satisfy a condition, write ‘None.’ If all the target words satisfy a condition, write ‘All.’ Otherwise, provide the index numbers of the target words that satisfy each condition. Output your responses in JSON format with two keys: ‘yes’ for target words used with consistent meanings and ‘no’ for those used with inconsistent meanings. A formatted example output is provided below. {‘yes’: [None/All or index numbers of target words used with consistent meanings], ‘no’: [None/All or index numbers of target words used with inconsistent meanings]} Here are the target words along with their contexts: $texts JSON output:

Table 14: Prompt templates for WiC.

Task	Prompt template
SelectOne Simplified	Go over the $num lines of text below and list the index numbers of the lines that contain the word '$category' according to the following instructions: If none of the texts contain the word '$category,' write 'None.' If all the texts contain the word '$category,' write 'All.' Otherwise, provide the index numbers of the texts that contain the word '$category' each on a separate line. Texts, one per line: $texts 'None,' 'All,' or the index numbers of the texts that contain the word '$category,' one per line:
SelectAll Simplified	In this task, each line of text contains one of four words 'Business,' 'Sports,' 'World,' and 'Sci/Tech.' Go over the $num lines of text below and list the index numbers of the lines that contain each word according to the following instructions: If none of the texts contain a particular word, write 'None.' If all the texts contain a particular word, write 'All.' Otherwise, provide the index numbers of the texts that contain each word. Output your responses in JSON format with the following keys: 'business,' 'sports,' 'world,' and 'sci/tech.' A formatted example output is provided below. {'business': [None/All or index numbers of texts containing 'Business'], 'sports': [None/All or index numbers of texts containing 'Sports'], 'world': [None/All or index numbers of texts containing 'World'], 'sci/tech': [None/All or index numbers of texts containing 'Sci/Tech']} Texts, one per line: $texts JSON output:

Table 15: Prompt templates for simplified SelectOne and SelectAll based on AGNews. For SelectOne Simplified, we used an earlier prompt template that asks LLMs to produce outputs line by line, instead of a JSON object. As illustrated earlier, we find the effect of the two output formats for SelectOne to be minimal. Therefore, having a JSON output should not make a meaningful difference.

Benchmark	Task	Answer template
SST-2	SelectOne	To solve this task, let's first classify the 10 lines of text above, one per line: {labels} From here, we see that lines {indices} show positive sentiment. Therefore, the answer in JSON format is as follows: {final answer}
SST-2	SelectAll	To solve this task, let's first classify the 10 lines of text above, one per line: {labels} From here, we see that lines {indices} show positive sentiment, and lines {indices} show negative sentiment. Therefore, the answer in JSON format is as follows: {final answer}
CoLA	SelectOne	To solve this task, let's first classify the 10 lines of text above, one per line: {labels} From here, we see that texts in lines {indices} are unacceptable. Therefore, the answer in JSON format is as follows: {final answer}
CoLA	SelectAll	To solve this task, let's first classify the 10 lines of text above, one per line: {labels} From here, we see that texts in lines {indices} are acceptable and texts in lines {indices} are unacceptable. Therefore, the answer in JSON format is as follows: {final answer}
AGNews	SelectOne	To solve this task, let's first classify the 10 lines of text above, one per line: {labels} From here, we see that texts in lines {indices} can be classified as 'Sports.'. Therefore, the answer in JSON format is as follows: {final answer}
AGNews	SelectAll	To solve this task, let's first classify the 10 lines of text above, one per line: {labels} From here, we see that texts in lines {indices} can be classified as 'Business,' texts in lines {indices} can be classified as 'Sports,' texts in lines {indices} can be classified as 'World,' and texts in lines {indices} are 'Sci/Tech.' Therefore, the answer in JSON format is as follows: {final answer}
MRPC	SelectOne	To solve this task, let's first determine if text A is a paraphrase of text B for the 10 lines of text above, one per line: {labels} From there, we see that text pairs in lines {indices} are paraphrases. Therefore, the answer in JSON format is as follows: {final answer}
MRPC	SelectAll	To solve this task, let's first determine if text A is a paraphrase of text B for the 10 lines of text above, one per line: {labels} From there, we see that text pairs in lines {indices} are paraphrases and text pairs in lines {indices} are not. Therefore, the answer in JSON format is as follows: {final answer}
SNLI	SelectOne	To solve this task, let's first determine the inference relation between the premise and the hypothesis for the 10 lines of text above, one per line: {labels} From there, we see that text pairs in lines {indices} contain entailment inference relation. Therefore, the answer in JSON format is as follows: {final answer}
SNLI	SelectAll	To solve this task, let's first determine the inference relation between the premise and the hypothesis for the 10 lines of text above, one per line: {labels} From there, we see that text pairs in lines {indices} contain entailment inference relation, text pairs in lines {indices} contain contradiction inference relation, and text pairs in lines {indices} contain neutral inference relation. Therefore, the answer in JSON format is as follows: {final answer}
WiC	SelectOne	To solve this task, let's first determine if the target word is used with consistent meanings in the two subsequent contexts for the num lines of text above, one per line: {labels} From there, we see that text pairs in lines {indices} contain use the target words with inconsistent meanings. Therefore, the answer in JSON format is as follows: {final answer}
WiC	SelectAll	To solve this task, let's first determine if the target word is used with consistent meanings in the two subsequent contexts for the num lines of text above, one per line: {labels} From there, we see that text pairs in lines {indices} contain use the target words with inconsistent meanings and text pairs in lines {indices} do not. Therefore, the answer in JSON format is as follows: {final answer}

Table 16: Answer templates for the two selection tasks for each classification benchmark. {indices}: a list of indices seperated by comma. {final answer}: answer in a JSON format specified by the SelectOne or SelectAll prompt.