Do LLMs Implicitly Determine the Suitable Text Difficulty for Users?

Seiji Gobara, Hidetaka Kamigaito, Taro Watanabe
Nara Institute of Science and Technology
{gobara.seiji.gt6, kamigaito.h, taro}@is.naist.jp

Abstract

Educational applications, including adaptive learning platforms and intelligent tutoring systems, need to provide personalized content with feedback in order to help improve learners' skills, and it is important for such applications to understand the individual learning level. When using large language models (LLMs) for educational applications leveraging its response generation capacity, the LLMs should be able to provide appropriate feedbacks to users. This work investigates how well LLMs can implicitly adjust their difficulty level to match with the user input when generating their responses. We introduce a new dataset from Stack-Overflow, consisting of question-answer pairs related to programming, and propose a method to analyze the ability in aligning text difficulties by measuring the correlation with various text difficulty metrics. Experimental results on our Stack-Overflow dataset show that LLMs can implicitly adjust text difficulty between user input and its generated responses. Similar trends were observed for the multi-turn English lesson dataset of Teacher Student Chatroom Corpus (TSCC). We also observed that some LLMs, when instruction-tuned, can surpass humans in varying text difficulty.

1 Introduction

Educational applications, including adaptive learning platforms and intelligent tutoring systems, need to provide personalized content with feedback in order to help improve learners' skills. It is important for those applications to understand the individual learning level to enhance learners' understanding in educational applications (Wang et al., 2024b; Huber et al., 2024).

As one such application, Dijkstra et al. (2022) use large language models (LLMs) to spark curiosity for boosting children's motivation to learn. Gabajiwala et al. (2022) incorporate LLMs into interactive contents such as quizzes and flashcards to enhance engagement and learning of users. Abu-Rasheed et al. (2024) proposed a method that utilizes a knowledge graph to extract information relevant to learners' input questions and incorporates it into prompts, thereby providing information aligned with the learners' intentions.

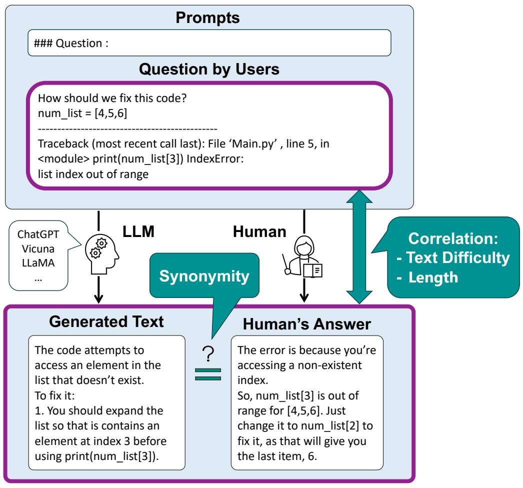

Figure 1: Overview of our evaluation procedure. We evaluate the generated texts from LLMs by comparing the correlation of text difficulty and appropriateness in length with user questions. We also measure the synonymity between the LLM generated texts and human answers.

Current research suggests LLMs may be useful for generating personalized problems and lecture content aligned with learners' comprehension levels (Baskara et al., 2023). LLMs can achieve this adaptation through reinforcement learning from human feedback (RLHF) that takes human preferences into consideration (Ouyang et al., 2022).

As an aspect of personalized response, LLMs should adjust the text difficulty to match user's comprehension level. Imperial and Madabushi (2023) examined the capability of GPT-2 (Radford et al., 2019) to adjust and generate complex texts. However, their analysis was limited to GPT-2, and a comprehensive study of LLMs has not yet been conducted. Thus, we explored the ability of LLMs to adjust the difficulty of responses to match the difficulty of user input as an approach in understanding the learner's level for educational appllications. Responses at the same level as the user will be better understood by the user. Accordingly, we hypothesized that the "simplicity/difficulty level" is nearly equal to the "user's understanding level". Here, the simplicity or difficulty level of user text serves as a proxy to estimate the user's understanding level. Figure 1 shows an overview of our experiment. The figure illustrates how we evaluate LLM performance by comparing their generated responses to human answers, focusing on how well they implicitly adjust text difficulty. We give the same user question to both LLMs and humans, measuring their ability to adjust responses based on text difficulty and length. By comparing the synonymity between LLM-generated responses and human answers, we further measure the LLMs' ability to generate plausible responses.

To measure the ability of LLMs to adjust simplicity/difficulty level in their response, we conducted an experiment on two different datasets. We created a Stack-Overflow dataset, which is related to programming, by extracting the question-answer pairs that cover a wide range of text difficulty. In addition, we run our experiments using the Teacher Student Chatroom Corpus (TSCC) (Caines et al., 2020) in order to understand how LLMs respond to English language learners.

Experimental results on our Stack-Overflow dataset and the TSCC dataset show that LLMs adjust the difficulty of the generated text to match those of the user input, even in the zero-shot setting. We also observed that the text difficulty of LLMs' output is more closely correlated to the question's text difficulty than to the original human answers. Instruction-tuned models exhibited even stronger correlations, indicating that Instruction-Tuning may enhance the ability to adjust implicit text difficulty. The response with the same level of user will be better understood by user.

2 Related Work

Dataset There are existing datasets such as BoolQ (Clark et al., 2019) (yes/no questions), Natural Question (Kwiatkowski et al., 2019) (short and paragraph-length answers), CommonsenseQA (Talmor et al., 2019) (common knowledge), and OpenQA (Wang et al., 2023) (factuality) mainly feature short answers, making them unsuitable for reliably measuring text difficulty using automatic evaluation metrics, such as FKGL (Klare, 1974) and FRE (Kincaid et al., 1975), which require longer inputs. Thus, existing datasets lack the longer input lengths necessary to reliably measure LLMs' ability to implicitly adjust text difficulty. This highlights the need for a new dataset with longer questions and answers for better text difficulty evaluation.

Statistics	Question Title	Question Body	Answer Body
Min	3.0	23.0	3.0
Max	41.0	9,382.0	6,545.0
Median	13.0	334.0	222.0
Mean	14.6	537.8	337.9

Table 1: Token count statistics for the Stack Overflow dataset, calculated using the LLaMa-2 Tokenizer. For more details, see Appendix A.2.

Adaptive Learning Some studies aim to provide personalized learning methods through prompt tuning and model training for educational purposes (Wang et al., 2024b; Huber et al., 2024; Baskara et al., 2023), which have further evolved into user-friendly applications (Dijkstra et al., 2022; Gabajiwala et al., 2022; Abu-Rasheed et al., 2024; Imperial and Madabushi, 2023; Pu and Demberg, 2023). For the further development of LLMs, it is crucial to assess if LLMs can understand not only the content of questions but also adjust to text difficulty. The TSCC dataset, which focuses on dialogues between teachers and language learners, is relevant for this analysis. However, its short responses make thorough analysis challenging. Thus, current research has yet to sufficiently address this issue, highlighting the need for more detailed analysis.

3 Dataset Construction

We constructed a dataset for effectively comparing text difficulty, comprising two parts, questions and answers. Since short target sentences may lead to inaccurate difficulty assessments, existing QA datasets such as SQuAD (Rajpurkar et al., 2016), which typically contain brief answers (for example, a single word or sentence), do not meet our criteria. Thus, both parts maintain sentences of sufficient length to ensure reliable difficulty estimation by checking token counts (see Appendix A.2).

To address this challenge, we created a dataset from Stack-Overflow[1], selecting data as of July

[1]https://stackoverflow.com/

Stack-Overflow	
Setting	Prompt
Normal	### Question : {$Title$} {$Question$}
Simple	Please respond to the question using simple and user-friendly language. ### Question : {$Title$} {$Question$}
Complex	Please respond to the question using complex and less user-friendly language. ### Question : {$Title$} {$Question$}

TSCC
Please generate a response from the teacher to the student in the ongoing dialogue. ### Dialogue : {$Dialogue$}

Table 2: Prompts for each setting. Note that TSCC has only one prompt.

1, 2023. We then extracted 1,000 posts starting from the most recent ones to optimize the scope of feasible experiments under constrained resources. The extracted posts contain significantly more tokens than typically observed ones in QA datasets such as BoolQ(Clark et al., 2019), Natural Question(Kwiatkowski et al., 2019), CommonsenseQA(Talmor et al., 2019), and Open-QA(Wang et al., 2023).

We then extracted the "QuestionTitle", "QuestionBody", and "AnswerBody" fields from each post. We combined "QuestionTitle" and "QuestionBody" to form the Questions parts and designated "AnswerBody" as the Answers. Table 1 summarizes the token count statistics for the Stack Overflow dataset, showing the distribution across "QuestionTitle", "QuestionBody", and "AnswerBody". As shown in the table, some inputs exceed the context size manageable by many models (approximately 4,096 to 8,192 tokens). However, the majority of questions fit within 2,048 tokens, which allows us to evaluate the models' implicit difficulty adjustment capabilities. Thus, in this study, we limit the input to LLMs to 2,048 tokens, truncating any spurious tokens, as detailed in Appendix A.2. We will release our code and dataset at https://github.com/satoshi-2000/llms-suitable.

4 Evaluation Procedure

4.1 Prompts

When prompts explicitly indicate the difficulty level, there's a risk of leakage of the difficulty level adjustment, which might lead to inappropriate personalization not aligned with the learner's understanding (Rooein et al., 2023). Therefore, to evaluate the LLM's implicit ability to adjust the difficulty level, we excluded the user's text comprehension level from the prompts or inputs, as detailed in Tables 2.

To assess the effectiveness of prompts, we collected and compared examples of language model outputs across three settings — simple, normal, and complex — within the Stack Overflow dataset, and another setting within the TSCC dataset. Table 2 shows the prompts we employed in each setting. In the "normal" setting, we did not provide explicit instructions for difficulty adjustment, allowing us to observe the model's inherent ability to adapt.

Conversely, in the "simple" setting, we instructed the model to generate responses that were simple and user-friendly, while in the "complex" setting, we explicitly directed the model to produce responses that were complex and less user-friendly. This approach allowed us to compare the model's ability to adjust difficulty under both implicit and explicit guidance.

4.2 Metrics

We examine the difficulty adjustment ability of LLMs using three evaluation indicators: text difficulty; synonymity; and appropriate text length. In text difficulty and appropriate text length, we calculated Spearman's rank correlation coefficient between the input and generated texts after ranking them based on the scores of these metrics. Additionally, we recorded the number of inappropriate text generations (skip rows), such as blanks. Furthermore, we computed the Mean Absolute Error (MAE) and the mean of the above metrics, as detailed in Appendix B.

Text Difficulty In language education contexts, it's crucial for teachers to adapt explanations to match learners vocabulary and comprehension levels. Thus, we measure this ability using text difficulty metrics assuming that it reflects the level understanding. The indicators include traditional ones like FKGL (Klare, 1974), FRE (Kincaid et al., 1975), and SMOG (Mc Laughlin, 1969), as well as NERF (Lee and Lee, 2023). NERF uses manually created features based on vocabulary difficulty, sentence structure complexity, the diversity of unique words, and bias to formalize text difficulty, offers a more accurate estimation of text difficulty than traditional metrics like FKGL and SMOG.

Synonymity To assess synonymity, it's essential to determine if LLMs deliver the correct content. Thus, we calculated BERTScore (Zhang et al., 2020) for texts generated by LLMs using the collected dataset's texts as references to ensure that LLMs align with the user's intended content.

Appropriate Length In question-answering and educational contexts, it's crucial that responses are both concise and appropriately detailed. Responses that are too short or too long can hinder user comprehension and clarity. However, determining a universally optimal response length is challenging, as it varies with the user's expertise and preferences. In this study, we operate under the assumption that longer input questions warrant more detailed responses, while shorter questions call for greater brevity. Thus, we investigated if LLMs can produce responses of appropriate length — neither too long nor too short — by comparing the length of LLMs generated texts to the input texts.

5 Experimental Setup

5.1 Dataset

We conducted experiments on two datasets: our Stack-Overflow dataset consisting of question-answer pairs related to programming , and the Teacher-Student Chatroom Corpus (TSCC) (Caines et al., 2020) consisting of dialogue histories collected during English lessons. These datasets were used to compare how the ability to adjust text difficulty changes in single-turn Stack Overflow pairs and multi-turn TSCC dialogues.

Stack-Overflow We used our created Stack-Overflow dataset in Section 3, consisting of 1,000 selected entries with HTML tags removed. It was scraped from question datasets as of July 1, 2023.

TSCC We extracted the TSCC dialogue histories from the beginning, prefixed each turn with the label of the speaker ("teacher" or "student"). For our experiments, we used the dialogues up to just before the first turn where the teacher speaks after the initial 10 turns, ensuring that the LLM is given the teacher's turn.

5.2 Models

To assess the ability of LLMs to adjust text difficulties for users, we compared various models. We hypothesized that LLMs, when trained on data reflecting human preferences, have the potential to align with learners' comprehension levels. Accordingly, we focused our evaluation on models such as GPT3.5/4 (Ouyang et al., 2022; OpenAI et al., 2023) and Vicuna (Zheng et al., 2023).

We also hypothesized that instruction-tuning is effective in acquiring the implicit ability to adjust text difficulty. Thus, we chose several models: LLaMa-2 and LLaMa-2-chat (Touvron et al., 2023b); CodeLLaMa and CodeLLaMa-Instruct (Roziere et al., 2023); Mistral and Mistral-Instruct (Jiang et al., 2023); Orca (Mitra et al., 2023); and OpenChat (Wang et al., 2024a).[2]

GPT3.5/4 (Ouyang et al., 2022; OpenAI et al., 2023) is an LLM that uses Reinforcement Learning from Human Feedback (RLHF) to align with human preferences, and it stands out for its exceptionally high performance among current LLMs.

LLaMA-2 (Touvron et al., 2023b) is an LLM pre-trained and fine-tuned across a range of 700 million to 7 billion parameters. This model not only outperforms LLaMA and its variants (Touvron et al., 2023a) in numerous benchmarks but has also undergone manual reviews for its usefulness and safety, indicating its potential to substitute closed-source models. Besides, it includes variations with different parameter sizes and versions fine-tuned for dialogue data and source code, such as LLaMA-2-chat and Code-LLaMA (Roziere et al., 2023).

Vicuna (Zheng et al., 2023) is an LLM trained to align with human preferences using data from ShareGPT [3] interactions, and based on LLaMA (Touvron et al., 2023a). We selected the 1.5 version of this model based on LLaMA-2 to analyze the impact of on text difficulty adaptation.

Orca (Mitra et al., 2023) is a model fine-tuned with prompts from various strategies, enabling it to adjust difficulty and offer flexible outputs in response to input sentences.

[2] See Appendix C for further details.

[3] https://sharegpt.com/

Mistral (Jiang et al., 2023) is a pre-trained model with 7 billion parameters. Compared to the larger parameter-sized 13B model of LLaMA-2, Mistral has recorded high performance in benchmarks.

OpenChat (Wang et al., 2024a) builds on Mistral (Jiang et al., 2023) and ShareGPT for training, enhancing learning by leveraging data quality variance between GPT-3.5 and GPT-4 as a reward mechanism.

Starling (Zhu et al., 2023a) is trained with a reward model derived from feedback on GPT-4 (OpenAI et al., 2023) and builds upon OpenChat (Wang et al., 2024a), which itself was fine-tuned from Mistral. We aim to explore whether models based on Mistral can develop the ability to modulate difficulty levels through fine-tuning.

To ensure reproducibility, we fixed the random seed and temperature for sentence generation. We detailed inference setting in Appendix A.

6 Results and Discussion

6.1 Stack-Overflow

Normal Figure 2 shows the result on our Stack-Overflow dataset and TSCC dataset under each setting. Although many models score high on BERTScores, the LLaMA-2 base model presents lower scores due to over- and under-generation. This result contrasts LLaMa-2-chat, showing instruction-tuning's effectiveness in considering human responses. Also, LLaMa-2-chat performs well in the correlation of text difficulty with other instruction-tuned models, Vicuna-13B and Mistral-7B-Instruct. From the result, we can understand the importance of instruction-tuning in the correlation.

On the other hand, CodeLlama-Instruct, which is instruction-tuned for code generation, shows low performance. Based on the successful result by LLaMa-2-chat, also instruction-tuned from LLaMa-2, this result indicates the importance of target tasks in instruction-tuning. We can observe similar trends between Mistral-7B-Instruct and its instruction-tuned variants, Openchat-3.5-7B and Starling-LM-7B.

Orca shows high performance as an instruction-tuned model. When comparing Orca-2-7B and Orca-2-13B, Orca-2-13B performs better across all metrics, underscoring the model's adherence to the scaling law. Nevertheless, LLaMA-2-chat maintains strong performance regardless of an increase in model size. Therefore, we can conclude the importance of the instruction-tuning method rather than model parameter size.

LLaMA-2-chat scores comparable to GPT-3.5 and GPT-4 in all metrics. This result is consistent with the human evaluations for helpfulness by LLaMA-2-chat reported in (Touvron et al., 2023b) and shows the potential of open-source models.

Simple In Figure 2, the results show a similar tendency to the normal setting, with instruction-tuned models also able to adjust text difficulty in response to the input in the simple setting.

Complex In Figure 2, even in the complex setting, instruction-tuned models adjust text difficulty based on the input, similar to the normal setting. However, the Spearman's correlation is lower in complex setting compared to the simple and normal settings. This may be due to prompts deliberately designed to elicit more complex responses, resulting in expressions that are harder to understand than the original responses.

Overall Comparing the normal, simple, and complex settings, the Spearman's correlation is consistently higher in the normal setting. The results in the normal setting were slightly higher than those in the simple setting, suggesting that human preference and instruction tuning have implicitly gave these models the capability to adjust text difficulty.

We also measured the correlation between input and output lengths, assuming longer questions require detailed responses and shorter ones should be concise. However, the correlation was consistently low across all models, showing the difficulty of handling generation lengths by LLMs similar to Juseon-Do et al. (2024). This suggests that input length alone doesn't fully capture the respondent's expertise or preference for response length. For example, step-by-step explanations may help beginners but can be redundant for professionals who prefer concise summaries. Further research is needed to determine the optimal level of detail based on the respondent's needs.

6.2 TSCC

Figure 2 shows the results of the TSCC dataset. The correlation coefficient scores for the difficulty of input and generated text are lower than that in the Stack-Overflow dataset, in contrast to the scores in BERTScore. Despite the challenging model outputs, we can observe the positive correlations by humans that indicate the validity of this dataset.

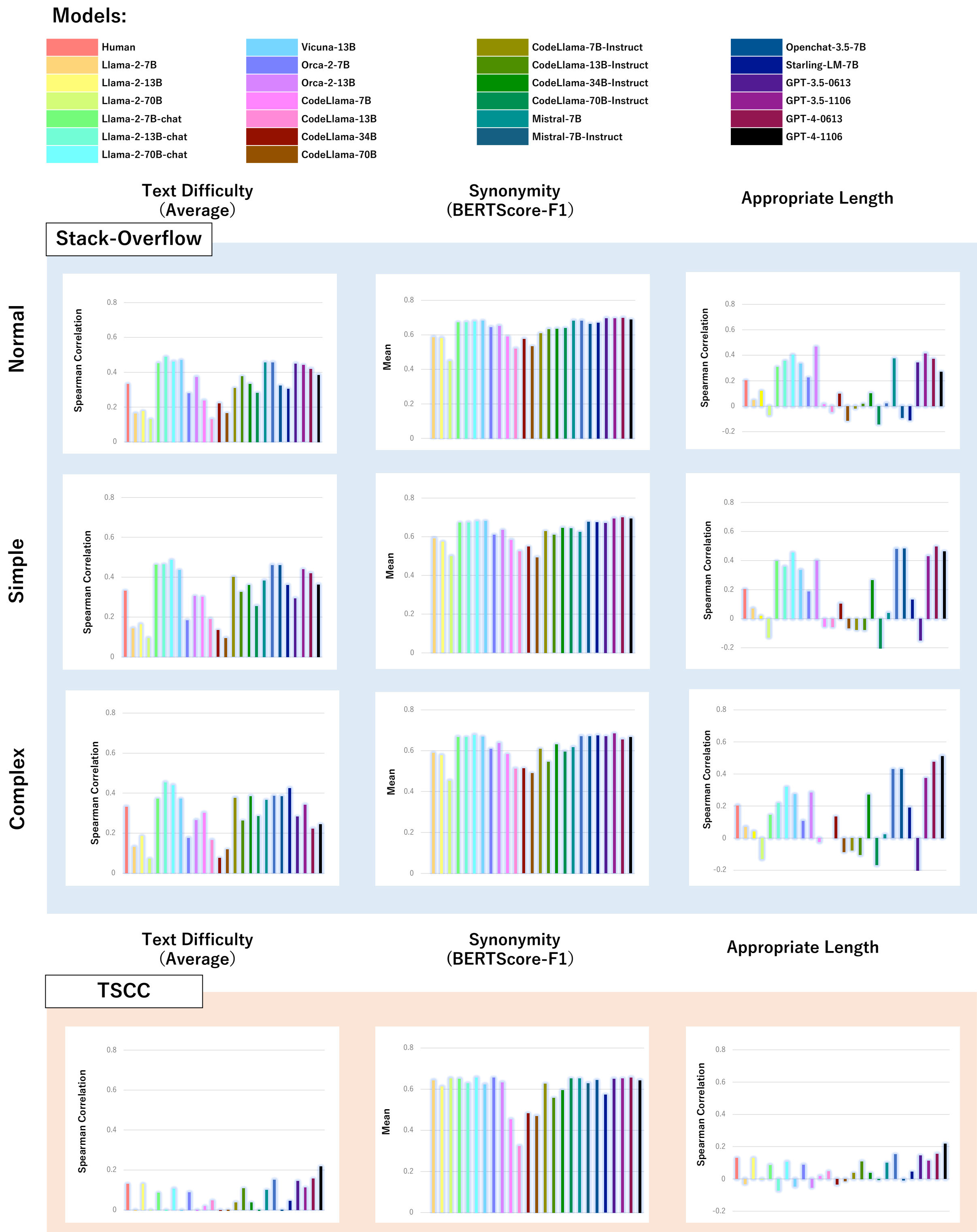

Figure 2: Results on the Stack-Overflow dataset and TSCC dataset. We observed similar trends across all evaluation metrics for text difficulty. Thus, we report the average scores of FKGL, FRE, SMOG, and NERF. Note that Table 5 and 9 in Appendix B include the detailed values.

Question Title	How is arbitrary distributed for Int? Why is it limited by so small values?
Question Body	I am trying to compare the QuickCheck library to the SmallCheck one. In SmallCheck I can reach particular value manipulating depth parameter. In QuickCheck: >a<-generate (replicateM 10000 arbitrary) :: IO [Int] >length a 10000 >maximum a 30 and my question then is: why are 10,000 ""random"" (""arbitrary"") integers limited by 30?! I expected to see more ""widely"" distributed values within the range 0..10,000, maybe the maximum value close to 5,000.
Human	The documentation contains a clue: The size passed to the generator is always 30 By default QuickCheck works by starting with ’easy’ or ’small’ inputs to see if it can find counterexamples with those. Only if it finds no problems with the small inputs does it gradually widen the range of generated input. The size value (which runs implicitly throughout everything that QuickCheck does) is the value that controls this behaviour. When you run QuickCheck (e.g. with quickCheck) it automatically increases the size as it goes. You’re not really supposed to use the generate function directly, but if you do, you can resize it: ghci> b <- generate (replicateM 10000 (resize 60 arbitrary)) :: IO [Int] ghci> maximum b 60 That said, how are you supposed to use QuickCheck? The documentation describes quickCheck along with a multitude of variations you can use to evaluate properties. Personally, I integrate my QuickCheck properties with a unit testing framework with testProperty. You can see examples here: Property-based testing is not the same as partition testing.
GPT-4	In QuickCheck, the ‘arbitrary‘ function for generating random values of type ‘Int‘ is designed to produce a wide range of values, including small and large numbers, positive and negative. However, the distribution is not uniform across the entire range of ‘Int‘. Instead, QuickCheck biases the distribution towards smaller numbers, which are often more likely to uncover edge cases and bugs in your code. The reason you’re seeing a maximum value of 30 in your example is likely due to the default size parameter used by QuickCheck when generating values. Quickheck uses a size parameter to control the ""size"" of generated values, which can mean different things for different types. For integral types, it often controls the maximum absolute value of the numbers generated. When you use ‘arbitrary‘ without specifying a size, QuickCheck uses a default size that is relatively small to ensure that the tests run quickly and that the generated values are not too large to handle. This is why you’re seeing a maximum value of 30 in your list of 10,000 integers. If you want to generate larger numbers, you can use the ‘resize‘ function to increase the size parameter, or the ‘scale‘ function to modify it. Here’s an example of how you might use ‘resize‘ to generate larger ‘Int‘ values: “‘haskell import Test.QuickCheck main :: IO () main = do largeInts <- generate $ resize 10000 $ vectorOf 10000 arbitrary print $ maximum largeInts “‘ In this example, ‘resize 10000‘ sets the size parameter to 10000, which should result in larger integers being generated. The ‘vectorOf 10000 arbitrary‘ generates a list of 10000 random integers using the new size parameter. Keep in mind that even with a larger size parameter, QuickCheck may still generate many small numbers, as it tries to cover a range of cases, including those that are more likely to cause problems in real-world code. If you want a uniform distribution across the entire range of ‘Int‘, you might need to use a different approach or a different library that provides such functionality." The behavior you observed in QuickCheck

Table 3: Examples of Question and Answering (gpt-4-1106-preview, normal)

Input	Please generate a response from the teacher to the student in the ongoing dialogue. ### Dialogue:student: Hi! teacher: Hi <STUDENT>! teacher: Everything alright with the chatroom for you? student: I tried to use it a few seconds ago and I couldn’t change my name, but now it is working, thanks. student: How are you? teacher: Oh good! teacher: Fine, thank you! It’s summer here at last, we’ve had a week of non-stop sunshine! teacher: How are you? student: I’m fine thank you! It looks like summer has arrived here too! student: Even though we still had a couple of storms... student: with hail and everything teacher:
Human	Have you guys been mostly unlocked now?
GPT-4	Oh wow, that sounds intense!

Table 4: Examples of dialogues (gpt-4-1106-preview)

In the open-source models, only Llama-2-70B-chat and Mistral-7B-Instruct achieve positive correlations in all metrics, whereas other models sometimes show negative correlations. However, these scores are lower than that of humans, demonstrating different tendencies from the Stack-Overflow dataset. Since the text in the TSCC dataset is often shorter than that in the Stack-Overflow dataset and uses dialogue-specific slang, models need to cover various domains and capture the implicit context of the conversation. Therefore, this result shows room for improvement in the instruction-tuning of open-source models by covering more various domains and diversified conversational text. Furthermore, the inconsistent tendencies of model parameter size support the conclusion induced by the results on the Stack-Overflow dataset that instruction-tuning is more important than the model parameter size.

Regarding GPT-3.5 and GPT-4, the results are remarkably high. These models achieve positive correlations in all metrics similar to humans. Because the details of GPT-3.5 and GPT-4 are not publicly available, we cannot judge what causes this remarkable performance. At least this result indicates the potential of LLMs in handling the correlation of text difficulty between user input and its corresponding response.

7 Analysis

7.1 Stack-Overflow

Table 3 presents examples of question-and-answer pairs from our Stack-Overflow dataset, comparing responses generated by GPT-4 (gpt-4-1106-preview) with original human responses under normal setting. The table includes a question about the default size parameter in the Haskell library QuickCheck and the corresponding answers. Both the human and GPT-4 responses mention the need to use “resize” to adjust the size parameter. The human response is concise, while GPT-4 offers a more detailed, step-by-step explanation. The most appropriate response depends on the user’s preferences and level of expertise, making this judgment subjective. However, it is qualitatively clear that the text difficulty in both the question and the responses is comparable.

While this example focuses on a question about a Haskell library, our Stack-Overflow dataset contains many other questions and answers related to various programming languages and environment setups. The expertise required for human annotators to accurately evaluate these responses is considerable and making manual evaluation challenging. Given these difficulties, our analysis relies on statistical data, which appears to be an effective approach for evaluating LLMs, particularly in contexts where manual evaluation is difficult.

7.2 TSCC

Table 4 presents an example of a single-turn teacher response for evaluation. As illustrated in Table 4, we compare the previous utterance, such as “Hi <STUDENT>!” and “with hail and everything,” to the response, “Oh wow, that sounds intense!”, focusing on text difficulty, synonymity, and appropriate length. As seen in Table 4, the generated responses in dialogue generation are often brief and do not reflect the assumed proficiency level of the interlocutor. This suggests that to accurately assess the implicit difficulty adjustment capability in dialogue generation, it is crucial to generate sufficiently detailed responses.

8 Conclusion

We explored LLMs’ ability to implicitly handle text difficulty between user input and generated text by comparing open-source LLMs and GPT-3.5/4 models in our Stack-Overflow dataset, based on question-answering, and the TSCC dataset, based on dialogue scenarios.

Experimental results on the Stack-Overflow show strong correlations in the text difficulty between texts from LLMs such as LLaMA-2-chat, Vicuna, GPT-3.5, and GPT-4 and their inputs. Notably, sometimes, LLMs even show higher correlation coefficients than human responses, underlining their potential for effective difficulty adjustment in question-answering. Furthermore, the experimental results on the TSCC dataset show the difficulty of handling text difficulty between user input and generated text.

Based on the results, we conclude the importance of instruction-tuning rather than the size of model parameters for implicitly handling text difficulty between user input and generated text by LLMs.

As our future work, we plan to identify preferences that improve this difficulty adjustment ability by examining how well LLMs acquire this skill from training data like dialogue histories. We will also explore the optimal response length for users with varying levels of expertise to further refine LLM performance.

9 Limitations

We conducted comparative experiments across various model types, yet we recognize the need for further exploration into datasets and evaluation methodologies.

Datasets We chose the Stack-Overflow dataset and TSCC for analyzing LLMs. These datasets focus on distinct domains: coding question-and-answer pairs and dialogue generation for educational guidance, respectively. To effectively evaluate the ability of LLMs to adjust difficulty implicitly, we suggest expanding the evaluations to include a wider variety of domains. This expansion should encompass specialized areas such as law or mathematics and general knowledge domains. Nonetheless, it's crucial to gather responses that are long enough to accurately evaluate the difficulty of texts produced by LLMs.

Evaluation To assess text difficulty, we selected an evaluation metric designed specifically for the English language. Therefore, adapting this evaluation method to other languages requires the use of metrics tailored to each respective language. Additionally, it's vital to verify if the difficulty level of texts produced by LLMs matches users' actual comprehension levels. Although we confirmed that texts generated by models can address certain issues within specific datasets, the extent of the data's contribution to solving problems and the reasons for failures when solutions are not achieved and remain unclear.

Analyzed Languages To assess the LLM's ability to implicitly adjust text difficulty, our analysis was limited to English. Consequently, for other languages, particularly those with limited linguistic resources, the model may not have fully developed this capability due to the reduced number of training tokens available.

10 Ethics Statement

The LLMs we used in our experiments might contain biases in the datasets utilized during training and the criteria used to ensure their quality. Additionally, the Stack-Overflow dataset employed in this study was collected by the authors themselves. However, for models released after the dataset was collected, there is a possibility that they were trained using the collected dataset.

References

Hasan Abu-Rasheed, Christian Weber, and Madjid Fathi. 2024. Knowledge graphs as context sources for llm-based explanations of learning recommendations. *arXiv preprint arXiv:2403.03008*.

Risang Baskara et al. 2023. Exploring the implications of chatgpt for language learning in higher education. *Indonesian Journal of English Language Teaching and Applied Linguistics*, 7(2):343–358.

Andrew Caines, Helen Yannakoudakis, Helena Edmondson, Helen Allen, Pascual Pérez-Paredes, Bill Byrne, and Paula Buttery. 2020. The teacher-student chatroom corpus. In *Proceedings of the 9th Workshop on NLP for Computer Assisted Language Learning*, pages 10–20, Gothenburg, Sweden. LiU Electronic Press.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Ramon Dijkstra, Zülküf Genç, Subhradeep Kayal, Jaap Kamps, et al. 2022. Reading comprehension quiz generation using generative pre-trained transformers.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3029–3051, Singapore. Association for Computational Linguistics.

Ebrahim Gabajiwala, Priyav Mehta, Ritik Singh, and Reeta Koshy. 2022. Quiz maker: Automatic quiz generation from text using nlp. In *Futuristic Trends in Networks and Computing Technologies: Select Proceedings of Fourth International Conference on FTNCT 2021*, pages 523–533. Springer.

Stefan E Huber, Kristian Kiili, Steve Nebel, Richard M Ryan, Michael Sailer, and Manuel Ninaus. 2024. Leveraging the potential of large language models in education through playful and game-based learning. *Educational Psychology Review*, 36(1):25.

Joseph Marvin Imperial and Harish Tayyar Madabushi. 2023. Uniform complexity for text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12025–12046, 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, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Juseon-Do, Jingun Kwon, Hidetaka Kamigaito, and Manabu Okumura. 2024. Instructcmp: Length control in sentence compression through instruction-based large language models.

J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel.

George R Klare. 1974. Assessing readability. *Reading research quarterly*, pages 62–102.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Bruce W Lee and Jason Hyung-Jong Lee. 2023. Traditional readability formulas compared for english. *arXiv preprint arXiv:2301.02975*.

G Harry Mc Laughlin. 1969. Smog grading-a new readability formula. *Journal of reading*, 12(8):639–646.

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, et al. 2023. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*.

OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al.

2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Dongqi Pu and Vera Demberg. 2023. ChatGPT vs human-authored text: Insights into controllable text summarization and sentence style transfer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 1–18, Toronto, Canada. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Donya Rooein, Amanda Cercas Curry, and Dirk Hovy. 2023. Know your audience: Do llms adapt to different age and education levels? *arXiv preprint arXiv:2312.02065*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Cunxiang Wang, Sirui Cheng, Qipeng Guo, Yuanhao Yue, Bowen Ding, Zhikun Xu, Yidong Wang, Xiangkun Hu, Zheng Zhang, and Yue Zhang. 2023. Evaluating open-QA evaluation. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Guan Wang, Sijie Cheng, Xianyuan Zhan, Xiangang Li, Sen Song, and Yang Liu. 2024a. Openchat: Advancing open-source language models with mixed-quality data. In *The Twelfth International Conference on Learning Representations*.

Shen Wang, Tianlong Xu, Hang Li, Chaoli Zhang, Joleen Liang, Jiliang Tang, Philip S Yu, and Qingsong Wen. 2024b. Large language models for education: A survey and outlook. *arXiv preprint arXiv:2403.18105*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. 2024. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The Twelfth International Conference on Learning Representations*.

Tianyi Zhang, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623. Curran Associates, Inc.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, and Jiantao Jiao. 2023a. Starling-7b: Improving llm helpfulness & harmlessness with rlaif.

Banghua Zhu, Hiteshi Sharma, Felipe Vieira Frujeri, Shi Dong, Chenguang Zhu, Michael I Jordan, and Jiantao Jiao. 2023b. Fine-tuning language models with advantage-induced policy alignment. *arXiv preprint arXiv:2306.02231*.

A Inference

A.1 Hyperparameters

We conducted 4-bit quantization for inference with a maximum input length of 2,048 tokens and a maximum output length of 3072 tokens. Since many models used in this comparative experiment employ the LLaMa-2 Tokenizer, we used it to measure the token count for consistency across evaluations. We limited the process to a single run since we used the already trained publicly available models in HuggingFace Transformers[4]. We set the random number seed to 42. We also set the temperature to 1.0.

A.2 Handling Long Inputs

Figure 3 shows a histogram of the number of tokens calculated using the tokenizer of Llama-2-7B (Touvron et al., 2023a) for the input data of the Stack-Overflow dataset. In Figure 3, 97.0% of all input data has 2,048 tokens or fewer, 98.1% has 3,072 tokens or fewer, and 1.9% has more than 3,072 tokens. To evaluate whether the model has acquired the ability to adjust difficulty levels in the outputs it generates for input sentences, it is not necessary to consider all input sentences; it is considered possible to capture the content of many input sentences sufficiently with 2,048 tokens. Therefore, to standardize the length of input and output sentences generated, the input to the model was truncated to up to 2,048 tokens, and the maximum number of tokens generated was adjusted to match the input tokens, resulting in 3072 tokens.

Additionally, we checked the input tokens and found that 94.3% are longer than 100 tokens. Thus, we can reliably estimate text difficulty.

A.3 Total Computational Budget

We utilized NVIDIA RTX A6000 GPUs for a total of 2,500 hours to evaluate open models. Additionally, we incurred $246.36 in costs through the OpenAI API[5] for evaluating GPT-3.5 and GPT-4 models.

B Detailed Results

We calculate the scores using pairs of input texts and their generated texts (human responses). Additionally, we calculate document length based on the number of characters.

[4] https://huggingface.co

[5] https://openai.com/api/

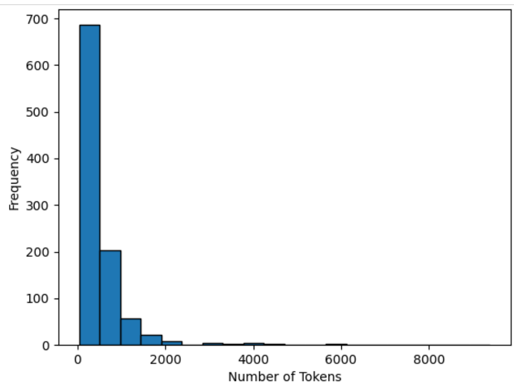

Figure 3: Histgrams of input tokens (Stack-Overflow)

B.1 Spearman Correlation

We compare LLMs' ability to adjust text difficulty and appropriate length using the Spearman correlation. Tables 5–8 show the actual scores.

B.2 Mean Scores

In Table 9–12 , we observe that models, with the exception of CodeLLaMa, which have enhanced ability to adjust difficulty, tend to produce shorter texts. This indicates that instruction-tuning likely facilitates the development of skills to appropriately regulate response lengths. Although this study evaluated the length of texts generated by LLMs in comparison to their original lengths, the ideal text length should naturally vary from one user to another. Thus, aside from extreme cases like CodeLLaMa, there's a need to explore effective evaluation methods for determining the suitable length of LLM-generated texts and to establish credible criteria for assessing longer text outputs.Additionally, GPT-4-1106 produced longer texts than those by previous versions, GPT-3.5 and GPT-4, suggesting it might use longer sequences for training. This indicates that GPT-4 may generate redundant responses without specific tuning prompts.

B.3 Mean Absolute Error

Tables 13–15 show that mean absolute error between input texts and generated texts. As shown in Table 13–15, we observed the tendency similar to the Spearman correlation. Additionally, well instruction-tuned models, such as LLaMA-2-chat and GPT4 score low mean absolute error.

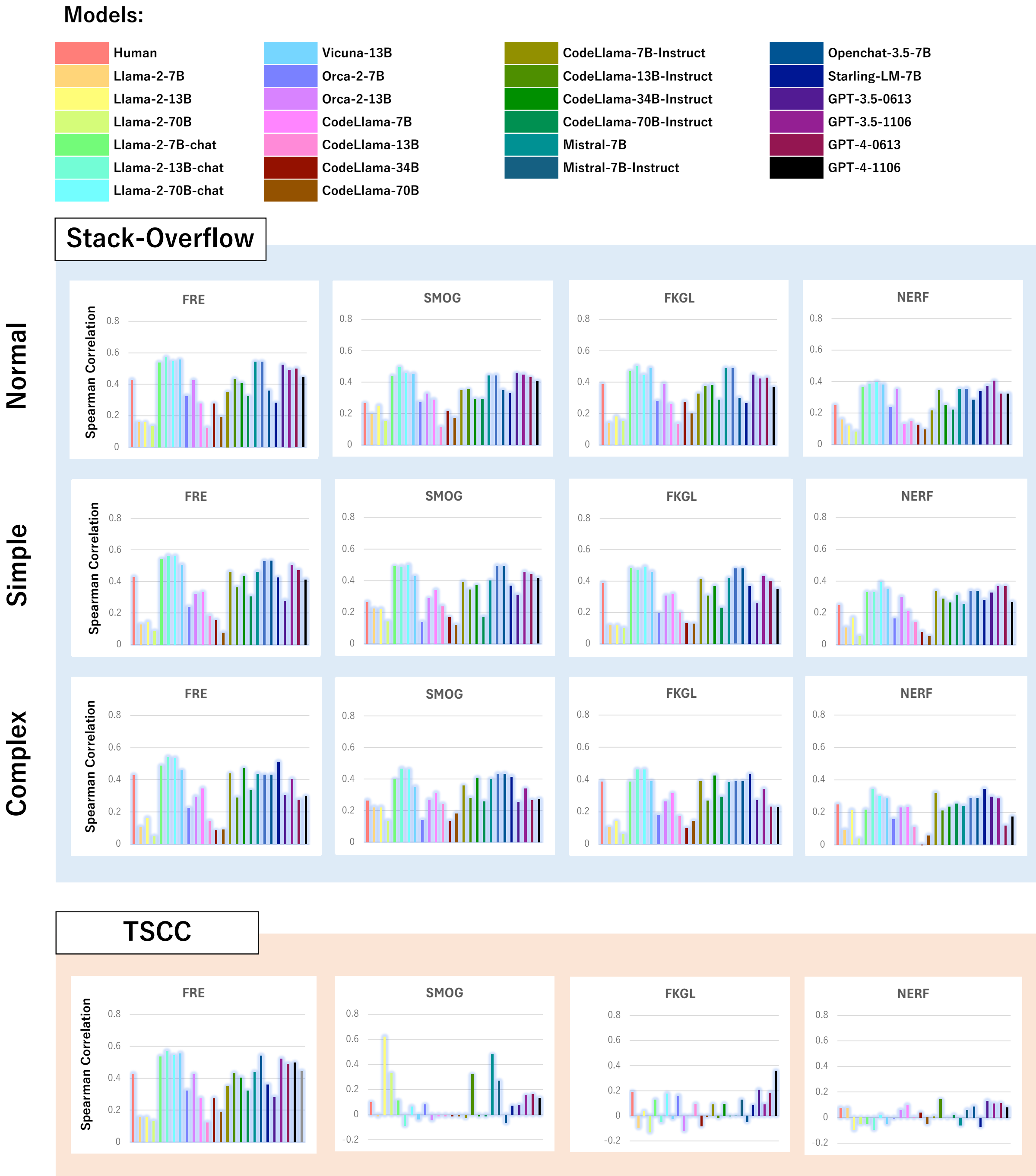

Figure 4: Results of the text difficulty on the Stack-Overflow dataset and TSCC dataset.

Models	FRE	SMOG	FKGL	NERF	Length
Human	0.428	0.265	0.387	0.248	0.203
Llama-2-7B	0.157	0.196	0.140	0.159	0.047
Llama-2-13B	0.157	0.249	0.182	0.118	0.119
Llama-2-70B	0.133	0.150	0.154	0.082	-0.070
Llama-2-7B-chat	0.538	0.438	0.469	0.364	0.306
Llama-2-13B-chat	0.571	0.495	0.502	0.386	0.356
Llama-2-70B-chat	0.545	0.459	0.445	0.397	0.402
Vicuna-13B	0.555	0.452	0.491	0.380	0.333
Orca-2-7B	0.324	0.271	0.280	0.239	0.226
Orca-2-13B	0.426	0.325	0.388	0.350	0.467
CodeLlama-7B	0.275	0.288	0.260	0.130	0.016
CodeLlama-13B	0.123	0.114	0.135	0.149	-0.043
CodeLlama-34B	0.275	0.212	0.275	0.125	0.098
CodeLlama-70B	0.192	0.173	0.199	0.093	-0.113
CodeLlama-7B-Instruct	0.349	0.347	0.325	0.215	-0.018
CodeLlama-13B-Instruct	0.433	0.354	0.376	0.343	0.017
CodeLlama-34B-Instruct	0.405	0.294	0.383	0.251	0.102
CodeLlama-70B-Instruct	0.322	0.293	0.288	0.222	-0.143
Mistral-7B	0.361	0.343	0.316	0.260	0.042
Mistral-7B-Instruct	0.542	0.443	0.489	0.353	0.375
Openchat-3.5-7B	0.359	0.348	0.300	0.283	-0.092
Starling-LM-7B	0.281	0.328	0.265	0.340	-0.110
GPT-3.5-0613	0.523	0.455	0.448	0.373	0.342
GPT-3.5-1106	0.492	0.448	0.422	0.405	0.414
GPT-4-0613	0.498	0.430	0.428	0.323	0.370
GPT-4-1106	0.443	0.407	0.366	0.322	0.268

Table 5: Stack-Overflow Normal Setting (Spearman Correlation)

B.4 Skip rows

Table 16 presents the skipped rows. As indicated in Table 16, instruction-tuned models adhere to the formats, exhibiting only a few skipped rows, with the exception of CodeLLaMA.

B.5 Text Difficulty

Figure 4 shows the results of text difficulty on Stack-Overflow dataset and TSCC dataset. In Figure 4, we can observe the same trends in the all metrics for text difficulty.

C Models Description

Table 17 shows various training methods for model tuning, including Supervised Fine-Tuning (SFT (Xu et al., 2024; Ding et al., 2023)), Reinforcement Learning Fine-Tuning (RLFT) (Schulman et al., 2017; Ouyang et al., 2022), Conditioned RLFT (C-RLFT) (Wang et al., 2024a), Advantage-Induced Policy Alignment (APA) (Zhu et al., 2023b), and Direct Preference Optimization (DPO) (Rafailov et al., 2024).

D Packages

We used several packages for scoring such as evaluate ($ver.$ 0.4.0)[6], textstat ($ver.$ 0.7.3) [7], spacy ($ver.$ 3.5.2) [8], and lftk ($ver.$ 1.0.9) [9].

E Ensuring License Compliance in Artifact Usage

We reviewed the license terms before comparing models to ensure adherence to the intended use. Additionally, we utilized AI assistants, including GPT3.5/4 and Copilot, for coding and writing the thesis.

[6]https://huggingface.co/docs/evaluate/index
[7]https://github.com/textstat/textstat
[8]https://spacy.io/
[9]https://github.com/brucewlee/lftk

Models	FRE	SMOG	FKGL	NERF	Length
Human	0.428	0.265	0.387	0.248	0.203
Llama-2-7B	0.128	0.222	0.117	0.109	0.070
Llama-2-13B	0.143	0.221	0.118	0.170	0.019
Llama-2-70B	0.085	0.142	0.100	0.053	-0.129
Llama-2-7B-chat	0.541	0.492	0.483	0.331	0.395
Llama-2-13B-chat	0.562	0.490	0.472	0.331	0.357
Llama-2-70B-chat	0.560	0.500	0.492	0.391	0.454
Vicuna-13B	0.503	0.428	0.460	0.351	0.335
Orca-2-7B	0.238	0.139	0.195	0.164	0.186
Orca-2-13B	0.322	0.289	0.308	0.300	0.400
CodeLlama-7B	0.332	0.341	0.316	0.215	-0.054
CodeLlama-13B	0.182	0.238	0.200	0.140	-0.057
CodeLlama-34B	0.154	0.167	0.133	0.081	0.104
CodeLlama-70B	0.075	0.120	0.128	0.053	-0.067
CodeLlama-7B-Instruct	0.460	0.392	0.412	0.338	-0.076
CodeLlama-13B-Instruct	0.362	0.343	0.307	0.289	-0.078
CodeLlama-34B-Instruct	0.435	0.370	0.369	0.265	0.265
CodeLlama-70B-Instruct	0.306	0.171	0.230	0.313	-0.379
Mistral-7B	0.461	0.400	0.418	0.257	0.040
Mistral-7B-Instruct	0.530	0.495	0.480	0.338	0.481
Openchat-3.5-7B	0.424	0.369	0.369	0.280	0.130
Starling-LM-7B	0.279	0.312	0.259	0.329	-0.149
GPT-3.5-0613	0.503	0.456	0.430	0.368	0.430
GPT-3.5-1106	0.472	0.442	0.401	0.367	0.496
GPT-4-0613	0.413	0.417	0.350	0.269	0.461
GPT-4-1106	0.432	0.397	0.363	0.323	0.335

Table 6: Stack-Overflow Simple Setting (Spearman Correlation)

Models	FRE	SMOG	FKGL	NERF	Length
Human	0.428	0.265	0.387	0.248	0.203
Llama-2-7B	0.107	0.221	0.105	0.092	0.070
Llama-2-13B	0.165	0.221	0.142	0.213	0.042
Llama-2-70B	0.049	0.137	0.064	0.038	-0.130
Llama-2-7B-chat	0.487	0.397	0.388	0.216	0.144
Llama-2-13B-chat	0.542	0.467	0.464	0.342	0.218
Llama-2-70B-chat	0.535	0.461	0.463	0.298	0.319
Vicuna-13B	0.458	0.352	0.390	0.285	0.273
Orca-2-7B	0.224	0.141	0.181	0.158	0.108
Orca-2-13B	0.296	0.271	0.264	0.229	0.285
CodeLlama-7B	0.346	0.313	0.315	0.233	-0.025
CodeLlama-13B	0.143	0.241	0.174	0.108	0.000
CodeLlama-34B	0.084	0.134	0.099	-0.011	0.134
CodeLlama-70B	0.089	0.182	0.144	0.058	-0.087
CodeLlama-7B-Instruct	0.440	0.359	0.389	0.321	-0.077
CodeLlama-13B-Instruct	0.288	0.280	0.270	0.212	-0.105
CodeLlama-34B-Instruct	0.471	0.409	0.425	0.236	0.272
CodeLlama-70B-Instruct	0.333	0.257	0.294	0.253	-0.169
Mistral-7B	0.438	0.400	0.384	0.240	0.023
Mistral-7B-Instruct	0.431	0.434	0.389	0.287	0.430
Openchat-3.5-7B	0.511	0.415	0.432	0.343	0.191
Starling-LM-7B	0.305	0.255	0.274	0.295	-0.218
GPT-3.5-0613	0.404	0.340	0.341	0.284	0.374
GPT-3.5-1106	0.276	0.266	0.231	0.118	0.475
GPT-4-0613	0.297	0.274	0.230	0.174	0.513
GPT-4-1106	0.370	0.304	0.311	0.197	0.297

Table 7: Stack-Overflow Complex Setting (Spearman Correlation)

Models	FRE	SMOG	FKGL	NERF	Length
Human	0.157	0.098	0.192	0.075	0.288
Llama-2-7B	-0.093	-0.010	-0.094	0.075	-0.062
Llama-2-13B	-0.041	0.622	0.035	-0.097	0.252
Llama-2-70B	-0.162	0.329	-0.129	-0.049	0.100
Llama-2-7B-chat	0.146	0.111	0.131	-0.048	0.047
Llama-2-13B-chat	-0.052	-0.089	-0.051	-0.095	0.061
Llama-2-70B-chat	0.159	0.066	0.178	0.022	0.288
Vicuna-13B	-0.076	-0.037	-0.024	-0.049	0.104
Orca-2-7B	0.124	0.079	0.160	-0.007	0.087
Orca-2-13B	-0.111	-0.041	-0.120	0.058	0.021
CodeLlama-7B	-0.016	-0.010	-0.001	0.099	0.044
CodeLlama-13B	0.098	-0.010	0.096	0.002	-0.020
CodeLlama-34B	-0.082	-0.010	-0.083	0.037	0.024
CodeLlama-70B	0.013	-0.010	-0.006	-0.049	-0.013
CodeLlama-7B-Instruct	0.074	-0.024	0.093	0.008	0.014
CodeLlama-13B-Instruct	-0.013	0.321	-0.016	0.141	-0.012
CodeLlama-34B-Instruct	0.062	-0.010	0.096	-0.002	0.044
CodeLlama-70B-Instruct	-0.029	-0.013	-0.003	0.019	-0.017
Mistral-7B	-0.022	0.478	0.002	-0.061	0.007
Mistral-7B-Instruct	0.149	0.270	0.130	0.059	0.001
Openchat-3.5-7B	-0.007	-0.065	-0.049	0.084	-0.031
Starling-LM-7B	0.096	0.071	0.084	-0.071	0.069
GPT-3.5-0613	0.163	0.076	0.210	0.130	0.301
GPT-3.5-1106	0.095	0.152	0.091	0.110	0.285
GPT-4-0613	0.167	0.163	0.184	0.113	0.285
GPT-4-1106	0.300	0.132	0.357	0.080	0.388

Table 8: TSCC Setting (Spearman Correlation)

Models	FRE	SMOG	FKGL	NERF	BERTScore (F1)	Length
Human	42.358	11.228	11.557	6.765	–	1729.109
Llama-2-7B	-3.915	8.785	21.369	3.843	0.587	5745.329
Llama-2-13B	-169.850	6.917	49.335	30.439	0.581	4583.894
Llama-2-70B	64.929	6.606	9.636	3.617	0.448	3995.069
Llama-2-7B-chat	49.029	11.994	11.040	3.758	0.672	1894.843
Llama-2-13B-chat	0.272	11.769	17.712	3.827	0.673	2100.051
Llama-2-70B-chat	49.231	12.006	11.013	4.200	0.679	1965.053
Vicuna-13B	48.784	11.442	10.807	4.627	0.682	1592.608
Orca-2-7B	74.026	8.663	6.453	2.839	0.646	1164.153
Orca-2-13B	72.520	8.637	6.708	3.072	0.652	1213.115
CodeLlama-7B	19.119	9.329	19.519	10.321	0.591	5979.621
CodeLlama-13B	-3.200	8.839	20.220	5.913	0.520	5309.517
CodeLlama-34B	34.064	7.996	13.963	3.287	0.577	3680.992
CodeLlama-70B	13.301	8.062	19.851	4.179	0.534	5884.443
CodeLlama-7B-Instruct	39.036	10.038	13.560	2.580	0.609	5778.107
CodeLlama-13B-Instruct	33.698	10.659	13.536	2.181	0.633	5014.724
CodeLlama-34B-Instruct	38.577	9.585	12.440	3.540	0.635	3912.342
CodeLlama-70B-Instruct	33.505	10.033	14.082	3.550	0.640	5985.720
Mistral-7B	30.479	10.171	16.333	1.278	0.619	5014.421
Mistral-7B-Instruct	43.342	11.579	12.014	4.425	0.683	1901.848
Openchat-3.5-7B	33.378	10.829	12.943	2.333	0.664	5747.161
Starling-LM-7B	8.850	11.288	16.642	3.150	0.670	6941.246
GPT-3.5-0613	47.954	11.901	10.775	4.939	0.697	1392.241
GPT-3.5-1106	47.598	12.308	11.199	5.157	0.695	1428.607
GPT-4-0613	54.886	11.190	9.617	4.348	0.699	1323.731
GPT-4-1106	50.680	12.286	10.829	5.660	0.688	2328.291

Table 9: Stack-Overflow Normal Setting (Mean)

Models	FRE	SMOG	FKGL	NERF	BERTScore (F1)	Length
Human	42.358	11.228	11.557	6.765	–	1729.109
Llama-2-7B	-44.747	9.201	30.078	9.021	0.591	6144.573
Llama-2-13B	-102.317	7.000	49.811	60.371	0.574	5583.394
Llama-2-70B	15.357	7.387	23.598	18.986	0.499	4715.437
Llama-2-7B-chat	52.186	11.782	10.514	3.732	0.672	1668.559
Llama-2-13B-chat	14.154	11.539	15.701	3.878	0.673	1883.881
Llama-2-70B-chat	50.721	11.805	10.640	4.183	0.680	1723.086
Vicuna-13B	53.171	10.886	10.121	4.274	0.681	1524.795
Orca-2-7B	66.388	6.515	6.583	1.268	0.609	933.419
Orca-2-13B	92.495	7.521	3.435	1.990	0.634	1091.195
CodeLlama-7B	-49.313	9.495	28.247	5.624	0.583	6344.100
CodeLlama-13B	39.978	8.331	13.847	1.397	0.525	5727.908
CodeLlama-34B	45.189	7.562	12.502	4.665	0.548	3846.843
CodeLlama-70B	22.740	7.409	18.632	3.908	0.493	5632.746
CodeLlama-7B-Instruct	21.654	10.439	15.220	1.592	0.629	6342.817
CodeLlama-13B-Instruct	48.177	9.885	10.735	1.162	0.609	5553.601
CodeLlama-34B-Instruct	53.111	10.520	9.878	3.002	0.646	2935.139
CodeLlama-70B-Instruct	39.935	11.960	12.655	2.310	0.643	8288.849
Mistral-7B	39.831	10.262	13.967	0.920	0.624	4611.053
Mistral-7B-Instruct	50.899	11.490	10.790	3.814	0.676	1647.081
Openchat-3.5-7B	46.104	11.085	10.975	3.363	0.674	3931.610
Starling-LM-7B	19.575	11.430	13.878	3.566	0.671	7286.648
GPT-3.5-0613	53.527	11.522	9.950	4.354	0.694	1181.735
GPT-3.5-1106	50.124	11.592	10.836	4.298	0.700	1009.199
GPT-4-0613	59.545	10.902	8.972	3.842	0.694	1004.923
GPT-4-1106	52.309	12.131	10.700	5.333	0.688	2112.660

Table 10: Stack-Overflow Simple Setting (Mean)

Models	FRE	SMOG	FKGL	NERF	BERTScore (F1)	Length
Human	42.358	11.228	11.557	6.765	–	1729.109
Llama-2-7B	-52.236	8.987	33.757	9.682	0.589	6134.990
Llama-2-13B	-64.823	7.199	41.513	57.876	0.578	5596.936
Llama-2-70B	27.943	6.778	19.205	13.451	0.453	4635.005
Llama-2-7B-chat	49.262	12.313	11.097	4.149	0.667	2018.452
Llama-2-13B-chat	44.077	11.584	11.635	3.836	0.666	2021.876
Llama-2-70B-chat	46.869	12.633	11.660	4.582	0.677	1996.049
Vicuna-13B	-153.948	11.281	39.172	4.811	0.668	1730.558
Orca-2-7B	102.040	7.175	1.910	1.479	0.609	1062.560
Orca-2-13B	78.777	9.046	5.805	2.742	0.638	1318.739
CodeLlama-7B	8.682	9.430	20.338	6.238	0.582	6280.695
CodeLlama-13B	37.556	8.202	14.556	1.852	0.512	5164.743
CodeLlama-34B	50.118	6.954	11.484	10.591	0.513	3610.031
CodeLlama-70B	23.125	7.549	18.884	3.738	0.490	5581.595
CodeLlama-7B-Instruct	45.469	10.016	12.308	1.235	0.608	6487.346
CodeLlama-13B-Instruct	63.227	9.083	8.981	1.438	0.545	5600.361
CodeLlama-34B-Instruct	59.502	10.586	8.969	2.824	0.631	2802.091
CodeLlama-70B-Instruct	57.045	10.067	9.969	1.521	0.596	7059.423
Mistral-7B	40.518	10.209	14.164	1.431	0.618	4777.848
Mistral-7B-Instruct	44.273	12.599	12.254	3.522	0.671	2033.776
Openchat-3.5-7B	44.957	12.189	11.445	4.209	0.675	3517.100
Starling-LM-7B	30.399	12.958	13.320	3.515	0.670	8060.675
GPT-3.5-0613	48.464	12.475	11.044	4.656	0.684	1380.164
GPT-3.5-1106	39.075	14.527	13.211	5.053	0.655	1233.771
GPT-4-0613	44.493	13.819	12.164	5.314	0.666	1615.374
GPT-4-1106	36.727	14.807	13.683	7.223	0.674	2661.302

Table 11: Stack-Overflow Complex Setting (Mean)

Models	FRE	SMOG	FKGL	NERF	BERTScore (F1)	Length
Human	88.507	0.567	3.119	-0.393	–	68.677
Llama-2-7B	82.350	0.025	5.864	0.203	0.642	113.088
Llama-2-13B	108.542	0.144	1.152	4.904	0.613	170.804
Llama-2-70B	110.196	0.125	-0.733	13.679	0.653	88.888
Llama-2-7B-chat	90.516	0.861	2.545	0.359	0.652	88.362
Llama-2-13B-chat	46.364	1.755	8.728	0.826	0.628	364.665
Llama-2-70B-chat	91.138	1.384	2.616	6.912	0.658	131.462
Vicuna-13B	88.828	0.390	2.607	9.391	0.623	89.227
Orca-2-7B	98.840	0.326	1.311	-0.221	0.655	62.408
Orca-2-13B	76.594	0.472	4.229	-0.280	0.634	84.462
CodeLlama-7B	124.331	0.012	-0.395	-0.337	0.454	78.050
CodeLlama-13B	152.196	0.039	-7.438	1.453	0.324	78.938
CodeLlama-34B	131.738	0.034	-4.277	22.400	0.483	97.919
CodeLlama-70B	127.126	0.012	-1.671	14.973	0.469	181.892
CodeLlama-7B-Instruct	104.029	0.141	0.207	-0.557	0.626	66.923
CodeLlama-13B-Instruct	107.991	0.090	1.725	7.333	0.558	117.608
CodeLlama-34B-Instruct	117.322	0.036	-1.806	42.973	0.594	221.046
CodeLlama-70B-Instruct	95.991	0.062	3.783	13.962	0.652	172.177
Mistral-7B	107.466	0.056	1.375	2.323	0.652	114.004
Mistral-7B-Instruct	102.654	0.100	1.192	16.965	0.629	225.177
Openchat-3.5-7B	95.955	1.367	1.599	3.411	0.644	531.673
Starling-LM-7B	66.350	7.813	7.132	1.823	0.573	5100.092
GPT-3.5-0613	80.366	6.560	4.636	1.877	0.651	204.042
GPT-3.5-1106	80.508	4.976	4.528	1.715	0.652	150.992
GPT-4-0613	80.493	5.217	4.444	1.775	0.656	157.319
GPT-4-1106	77.535	7.843	5.283	2.417	0.643	261.388

Table 12: TSCC Setting (Mean)

Models	FRE	SMOG	FKGL	NERF	Length
Human	25.878	3.526	4.575	2.895	1243.833
Llama-2-7B	97.339	4.577	18.853	10.719	4457.702
Llama-2-13B	251.946	5.414	44.864	38.081	3510.847
Llama-2-70B	97.945	6.168	18.376	10.124	3295.110
Llama-2-7B-chat	19.359	2.191	3.481	3.416	974.536
Llama-2-13B-chat	68.190	2.039	10.141	3.332	1061.472
Llama-2-70B-chat	18.181	2.097	3.363	3.051	933.708
Vicuna-13B	20.587	2.463	3.858	3.082	891.109
Orca-2-7B	49.525	4.575	8.502	4.573	1042.146
Orca-2-13B	49.349	4.434	8.499	4.459	948.356
CodeLlama-7B	67.783	3.993	15.805	15.995	4586.738
CodeLlama-13B	126.915	5.378	22.443	11.425	4037.748
CodeLlama-34B	70.241	4.720	13.151	8.122	2716.919
CodeLlama-70B	102.019	5.252	20.763	11.766	4692.574
CodeLlama-7B-Instruct	49.437	3.539	10.081	8.102	4469.528
CodeLlama-13B-Instruct	45.858	3.205	8.467	6.084	3802.495
CodeLlama-34B-Instruct	41.123	3.533	7.449	5.756	2915.341
CodeLlama-70B-Instruct	46.992	3.611	9.029	6.011	4658.893
Mistral-7B	53.374	3.621	11.932	8.225	3890.356
Mistral-7B-Instruct	22.860	2.292	4.252	4.147	1199.339
Openchat-3.5-7B	38.451	2.515	6.748	5.220	4447.172
Starling-LM-7B	47.231	2.325	7.845	4.779	5483.139
GPT-3.5-0613	20.233	2.195	3.522	2.353	980.324
GPT-3.5-1106	20.130	2.380	3.598	2.468	971.184
GPT-4-0613	20.491	2.085	3.516	2.684	962.822
GPT-4-1106	20.978	2.271	3.659	2.264	1423.798

Table 13: Stack-Overflow Normal Setting (Mean Absolute Error)

Models	FRE	SMOG	FKGL	NERF	Length
Human	25.878	3.526	4.575	2.895	1243.833
Llama-2-7B	137.635	4.339	29.784	17.822	4708.711
Llama-2-13B	139.464	5.575	35.837	63.870	4300.091
Llama-2-70B	123.341	6.373	25.888	20.171	3743.874
Llama-2-7B-chat	17.229	1.953	3.131	3.396	1043.307
Llama-2-13B-chat	27.461	2.029	4.525	3.793	1031.519
Llama-2-70B-chat	16.896	2.020	3.125	3.024	952.276
Vicuna-13B	229.641	2.655	33.083	4.292	1006.485
Orca-2-7B	72.674	5.964	12.163	6.059	1260.131
Orca-2-13B	48.830	4.415	8.382	4.767	1097.196
CodeLlama-7B	80.792	3.526	16.978	12.604	4844.746
CodeLlama-13B	91.405	4.852	17.564	8.963	3837.530
CodeLlama-34B	85.187	5.751	15.574	15.681	2673.364
CodeLlama-70B	114.259	5.776	23.174	12.558	4337.944
CodeLlama-7B-Instruct	41.407	2.943	8.409	8.134	5047.529
CodeLlama-13B-Instruct	54.775	3.954	10.138	7.931	4306.888
CodeLlama-34B-Instruct	29.485	2.501	4.989	4.976	1845.032
CodeLlama-70B-Instruct	40.356	3.577	7.526	6.220	5672.056
Mistral-7B	42.017	3.016	9.444	7.830	3644.151
Mistral-7B-Instruct	21.777	2.086	3.847	4.080	1067.657
Openchat-3.5-7B	19.613	1.878	3.464	3.450	2332.179
Starling-LM-7B	28.696	2.450	4.505	4.134	6512.476
GPT-3.5-0613	21.340	2.627	3.721	2.558	981.147
GPT-3.5-1106	25.569	3.976	4.691	2.625	934.750
GPT-4-0613	23.365	3.316	4.194	2.435	979.349
GPT-4-1106	25.152	4.068	4.766	2.576	1658.239

Table 14: Stack-Overflow Complex Setting (Mean Absolute Error)

Models	FRE	SMOG	FKGL	NERF	Length
Human	24.704	0.730	4.247	1.664	47.958
Llama-2-7B	52.819	0.262	10.880	1.957	116.385
Llama-2-13B	46.716	0.201	9.104	8.925	169.677
Llama-2-70B	34.875	0.273	5.945	15.910	88.654
Llama-2-7B-chat	30.699	0.968	5.071	1.913	67.696
Llama-2-13B-chat	87.097	1.992	13.059	2.426	345.292
Llama-2-70B-chat	29.152	1.489	4.882	8.678	104.558
Vicuna-13B	37.263	0.627	5.802	11.423	78.492
Orca-2-7B	32.298	0.517	5.305	1.902	50.550
Orca-2-13B	43.802	0.709	6.763	1.770	72.942
CodeLlama-7B	69.643	0.249	13.698	4.857	95.962
CodeLlama-13B	77.903	0.276	12.222	5.135	104.465
CodeLlama-34B	62.395	0.271	9.858	25.323	114.354
CodeLlama-70B	67.085	0.249	12.484	19.832	197.304
CodeLlama-7B-Instruct	36.872	0.378	5.936	1.624	66.681
CodeLlama-13B-Instruct	54.860	0.259	11.146	13.875	126.096
CodeLlama-34B-Instruct	39.097	0.273	6.500	45.192	222.504
CodeLlama-70B-Instruct	47.679	0.299	10.115	16.811	173.119
Mistral-7B	47.280	0.205	9.545	5.077	119.508
Mistral-7B-Instruct	34.043	0.277	6.084	18.761	207.588
Openchat-3.5-7B	30.908	1.580	5.260	5.334	525.646
Starling-LM-7B	34.648	7.653	6.012	3.222	5062.912
GPT-3.5-0613	23.879	6.412	4.047	2.916	160.192
GPT-3.5-1106	24.334	4.758	4.170	2.740	108.650
GPT-4-0613	24.354	4.991	4.170	2.807	111.985
GPT-4-1106	24.288	7.606	4.270	3.371	212.377

Table 15: TSCC Setting (Mean Absolute Error)

Models	Stack-Overflow			TSCC
Settings	normal	simple	complex	–
Human	0	0	0	0
Llama-2-7B	16	15	14	0
Llama-2-13B	7	6	6	4
Llama-2-70B	16	16	16	3
Llama-2-7B-chat	0	0	0	0
Llama-2-13B-chat	0	0	0	0
Llama-2-70B-chat	0	0	0	2
Vicuna-13B	0	0	0	5
Orca-2-7B	0	3	0	1
Orca-2-13B	0	0	0	1
CodeLlama-7B	16	16	16	4
CodeLlama-13B	16	16	16	5
CodeLlama-34B	16	16	16	5
CodeLlama-70B	16	16	16	5
CodeLlama-7B-Instruct	15	13	14	4
CodeLlama-13B-Instruct	15	16	16	4
CodeLlama-34B-Instruct	16	16	16	4
CodeLlama-70B-Instruct	13	15	16	2
Mistral-7B	15	15	15	1
Mistral-7B-Instruct	1	0	0	4
Openchat-3.5-7B	0	0	0	0
Starling-LM-7B	0	0	0	0
GPT-3.5-0613	0	0	0	0
GPT-3.5-1106	0	0	0	0
GPT-4-0613	0	0	0	0
GPT-4-1106	0	0	0	0

Table 16: Skip rows

Models	Base Models	Parameter Size	Datasets	Tuning Methods	versions	Author
Llama-2	(Base)	7B, 13B, 70B	Publicly available sources	(Base)	Llama-2-{7,13,70}b-hf	(Touvron et al., 2023b)
Llama-2-chat	Llama-2	7B, 13B, 70B	Publicly available sources	SFT + RLFT	Llama-2-{7,13,70}b-chat-hf	(Touvron et al., 2023b)
Vicuna	Llama-2	13B	ShareGPT	SFT	vicuna-13b-v1.5	(Zheng et al., 2023)
Orca	Llama-2	7B, 13B	Publicly available sources	SFT	microsoft/Orca-2-{7,13}b	(Mitra et al., 2023)
CodeLlama	Llama-2	7B, 13B, 34B, 70B	Publicly available sources	SFT + RLFT	CodeLlama-{7,13,34,70}b-hf	(Roziere et al., 2023)
CodeLlama-Instruct	Llama-2	7B, 13B, 34B, 70B	Publicly available sources	SFT + RLFT	CodeLlama-{7,13,34,70}b-Instruct-hf	(Roziere et al., 2023)
Mistral	(Base)	7B	–	(Base)	Mistral-7B-v0.1	(Jiang et al., 2023)
Mistral-Instruct	Mistral	7B	–	SFT	Mistral-7B-Instruct-v0.1	(Jiang et al., 2023)
Openchat	Mistral	7B	ShareGPT	C-RLFT	openchat_3.5	(Wang et al., 2024a)
Starling	Openchat	7B	Nector	C-RLFT+ APA	Starling-LM-7B-alpha	(Zhu et al., 2023a)
GPT3.5-Turbo	–	–	–	SFT + RLFT	gpt-3.5-turbo-{0613, 1106}	(Ouyang et al., 2022)
GPT4	–	–	–	SFT + RLFT	gpt-4-0613	(OpenAI et al., 2023)
GPT4-Turbo	–	–	–	–	gpt-4-1106-preview	(OpenAI et al., 2023)

Table 17: Models Description