PROSE: Prompt Refinement and Optimization via Scored Evaluation

Praveen Bandla* praveen.bandla@nyu.edu
Bella Chang* ic2664@nyu.edu
Darren Jian* darren.jian@nyu.edu
Rebecca Rinehart* r.rinehart@nyu.edu

*Center for Data Science, New York University

Abstract

Prompt design plays a critical role in the performance of Large Language Models (LLMs). In this work, we introduce PROSE (Prompt Refinement and Optimization via Scored Evaluation), a task-generalizable framework that automates prompt refinement through synthetic data generation and rubric-based evaluation. PROSE generates diverse prompt variations, evaluates their effectiveness using LLMs as rubric-based validators, and trains a lightweight Regression Head Model to predict output quality based solely on prompt structure. Using the task of generating learning guides for students in third grade as a case study, we analyze over 12,000 prompt variations across 300 base prompts, demonstrating that minor structural changes significantly influence model output quality. Our results show that structured variations, such as adding task emphasis or correcting typographical errors, lead to measurable improvements, while arbitrary changes like full capitalization degrade performance. The Regression Head Model achieves strong predictive accuracy with minimal overfitting, offering an efficient alternative to direct LLM fine-tuning. This work lays the foundation for fully automating prompt optimization and highlights the potential of rubric-guided evaluation in refining prompt engineering strategies.

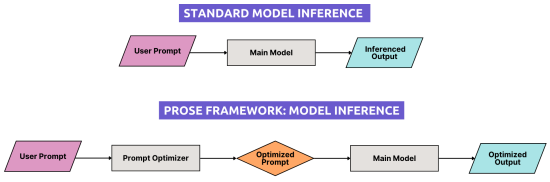

Figure 1: Our PROSE framework for LLM inference

1 Introduction

As LLMs become more integrated into daily tasks, differences in prompt structure — even minute — have been shown to significantly affect output quality (Sclar et al., 2024). Research on prompt phrasing sensitivity has investigated heuristic-driven techniques — such as clarifying instructions, specifying output formats, and incorporating few-shot examples — revealing how structure, syntax, and context shape LLM behavior. However, these manual approaches often rely on trial-and-error and lack scalability. Meanwhile, black-box methods, which optimize prompts without model internals or handcrafted heuristics, remain underexplored but offer a path toward more systematic and scalable improvements.

Prompt Optimization Modern prompt optimization approaches reduce computational cost but still face major limitations. Techniques like Model-Adaptive Prompt Optimization (MAPO) (Chen et al., 2024) tailor prompts to specific models, sacrificing generalizability, while Black-Box Prompt Optimization (BPO) (Cheng et al., 2024) addresses intent alignment without retraining but depends on costly manual annotations. Since strong prompts can outperform fine-tuning in complex tasks (Zhou et al., 2022; McDonald et al., 2024), we shift toward dataset-driven optimization, using a rubric-based validation system grounded in evidence that LLMs can reliably serve as graders (Hashemi et al., 2024).

Model Evaluation with LLMs Recent work in the model evaluation space shows that LLMs prompted with manually-defined rubrics can match human grading performance (Shakil et al., 2024; Chiang and yi Lee, 2023), while traditional metrics like ROUGE, BLEU, and BERTScore often correlate poorly with human evaluations (Nguyen et al., 2024). Based on analyses of LLM-graded rubrics (Shakil et al., 2024; Nguyen et al., 2024), we selected five qualitatively grounded categories to define our rubric, detailed in Validator Score.

Prompt Engineering Furthermore, prompt engineering has also emerged as a strategy for models to generalize out-of-distribution and improve model reliability. Recent work in understanding prompt engineering includes perturbing prompts with minor typos, resulting in incorrect LLM responses for math and sentiment analysis problems (Zhu et al., 2024). Encouraging an LLM to think "step-by-step" improves model performance on complex reasoning tasks (Wei et al., 2023). Furthermore, LLMs trained in languages that reflect cultural norms and values have exhibited alignment with the culture of that language (Cao et al., 2023). As a result, polite or impolite language can affect an LLM's performance on tasks like summarization, language understanding, and bias detection (Yin et al., 2024). Motivated by these findings, we examine how specific prompt modifications influence rubric scores and model behavior under the PROSE framework.

2 PROSE Framework: Conceptual Overview

We propose PROSE (Prompt Refinement and Optimization via Scored Evaluation), a task-generalizable framework for prompt optimization. PROSE introduces a pipeline for generating synthetic data related to any task, along with a procedure for training a dedicated Prompt Optimizer. The framework proceeds as follows:

- We collect a list of base prompts and generate multiple prompt variations for each base prompt.
- For each prompt variation, we produce sample model outputs.
- We employ LLMs-as-a-judge (validator models) to assign scores to each generated output, based on a pre-defined rubric.
- We train a Prompt Optimizer, which learns to take any base prompt related to our task to generate an optimized prompt after learning from the synthetic data.

Key features of the PROSE framework include:

Topic Adaptability: To adapt the framework to a new domain, only the modules responsible for generating prompt variations, model outputs, and the evaluation rubric need to be updated.

Custom Rubric Integration: An evaluation rubric is provided to guide the scoring process, ensuring that prompt optimization is aligned with domain knowledge and task requirements.

Two-Stage Optimization Approach: As shown in Figure 2, the Regression Head Model is trained on Validator Model scores from a synthetic dataset to predict output quality from prompt variations. A Prompt Generator then creates variations and uses Regression Head feedback, applying Proximal Policy Optimization (PPO) to iteratively refine them. The Regression Head quantitatively assesses which edits improve or degrade output performance, while the Prompt Generator automates prompt refinement based on predicted gains.

3 Hypothesis

We train the Regression Head Model to predict the anticipated validation scores of model outputs based on different prompt variations. We first posit that by doing so, we can reasonably estimate model output quality using only the base prompt and a corresponding prompt variation. After training the model, we generate targeted prompt variations and evaluate their predicted scores relative to the original base prompts. Specifically, we test variations focused on *Improvement*, *Typos*, *Capitalization*, *Structural Cues*, *Task Emphasis*, and *Partial Sentences* (see Experimenting with Prompt Variation Quality for details).

We hypothesize that *Improvement-Focused Variations*, *Structural Cues*, and *Task Emphasis* lead to higher predicted quality scores compared to the base prompts, reflecting improved structure and relevance to base prompt. Conversely, we anticipate that introducing *Typos* and applying the *Partial Sentences* condition will degrade performance, resulting in lower predicted scores. For variations involving *Politeness* and *Capitalization*, we predict minimal to no change in predicted quality, as we posit that these modifications are unlikely to materially alter the substance of the prompts.

4 Methodology and Experimentation Setup

In this section, we explain our experimentation setup for generating our synthetic data, training our model in line with our PROSE framework.

4.1 Task: Generation of Learning Guides

The task we chose for our synthetic data is creating learning guides for third-graders. To this end,

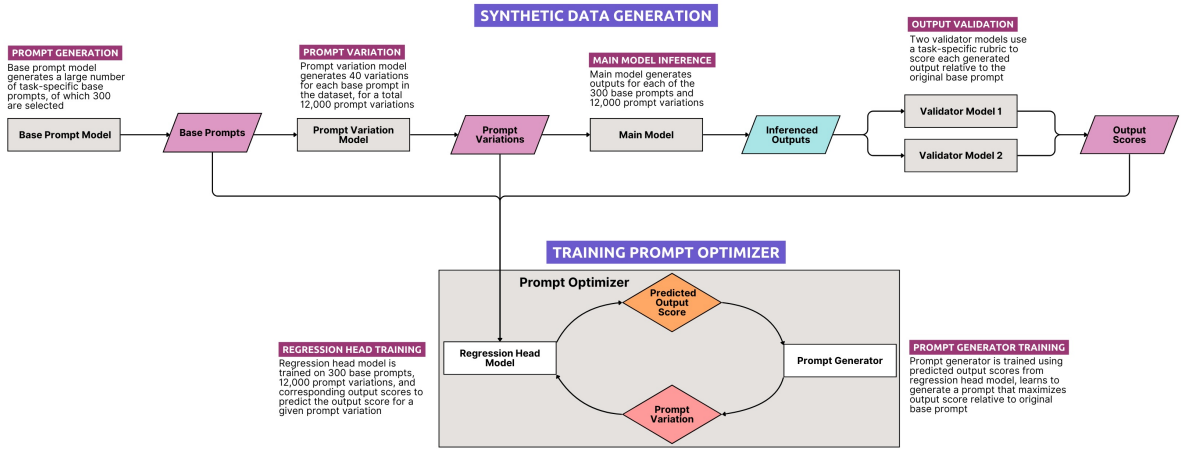

Figure 2: Our project workflow for building the PROSE framework

our base prompts and prompt variations all equate to instructions provided to LLMs on generating learning guides for third-graders on a variety of topics. Our model outputs correspond exactly to the learning guides themselves, while our evaluation rubric is tailored to specific criteria related to grading learning-guides.

4.2 Synthetic Data Generation

Here, we outline the specific procedures used to generate the various sub-components of the synthetic data. All inference instructions can be found in Appendix D. An example of each subcomponent can be found in Figure 9.

4.2.1 Base Prompt

Topic Generation. Using Llama-3.1-8B-Instruct, we generated a diverse set of 50 different topics. A sample list of topics can be found in Appendix B.

Base Prompt Generation. We randomly sampled from these 50 topics to generate the base prompts. For each selected topic, we then generated around 40 base prompts. The number of base prompts per topic was not exactly 40 due to inference inconsistencies, and we discuss this in Limitations. To ensure consistency for comparison, we asked the model to ensure that every base prompt includes the phrases "guide" and "third-grade," and after filtering for this criteria, the final base prompt set consisted of 1,766 samples. From this, we sampled 300 base prompts for training and 600 base prompts for testing. For further details, see Appendix C.

4.2.2 Prompt Variation

For each of the 600 base prompts in the training set, we instructed the Prompt Variation Model, Mistral-7B-Instruct-v0.3, to generate 50 alternative LLM prompt phrasings.

4.2.3 Model Output

For each of our prompt variations, we used the Main Model, Phi-3.5-mini-instruct, to generate exactly one model output to the given prompt variation. In the context of the task, these model outputs are the text of the learning guides.

4.2.4 Validator Score

We evaluated the learning guides generated for each prompt variation using a pre-defined, task-specific rubric that we designed to evaluate the quality of each learning guide across five categories (Figure 8). These categories include *Readability* and *Structure*, *Clarity* and *Directness*, *Conceptual Depth*, *Relevance to Prompt*, and *Engagement*, with specific weights for each. By evaluating and weighing across these individual categories, we present a method for more granular understanding of the framework's strengths and limitations. Furthermore, a user-defined rubric allows for greater flexibility and adaptability of the framework to new tasks or evaluation criteria.

The main model output, the rubric, and the inference instruction were provided to each of the Validator Models. For this, we used two models, Falcon3-3B-Instruct and Gemma-2-9b-it. For each learning guide, each of the Validator Models generated a singular score per category, which we averaged across all Validator Models. These averages of the sections were then combined with the

weights of the sections to calculate a total score.

4.3 Prompt Optimizer

The Prompt Optimizer is a combination of two different models. As outlined in PROSE Framework: Conceptual Overview, we first introduce the Regression Head Model, a model that learns to predict model output quality based on a given prompt variation of a base prompt. We then introduce our Prompt Generator, a model that learns to generate optimized prompt variations for a given base prompt. In this section we walk through the implementation of both.

4.3.1 Regression Head Model

The Regression Head Model predicts prompt variation output scores based on alignment with the base prompt. We initialize a pre-trained Llama-3.2-1B model, adding a fully connected regression head to map hidden states to a scalar score.

Training data includes base prompts, variations, and synthetic validator scores. The base prompt and variation are concatenated, tokenized, and processed by the model, with final hidden states pooled into a fixed-size vector for score prediction.

We then fine-tuned using Low-Rank Adaptation (LoRA) to efficiently update a small set of parameters while freezing most weights. The model minimizes mean squared error (MSE) between predicted and true scores. We conducted 10 trials, varying *LoRA Rank* (4–16), *Alpha* (16–64), *Dropout* (0.1–0.5), *Learning Rate* (1e-5–1e-2), and *Batch Size* (8–32). After training, we saved the model, tokenizer, and regression head for inference.

4.3.2 Prompt Generator

The Prompt Generator is designed to output optimized prompt variations based on a given base prompt, with training guided by the Regression Head Model. While we developed the training framework and reward mechanism for the Prompt Generator, we leave the full implementation and empirical evaluation to Future Work. Additional implementation details of the Prompt Generator are provided in Prompt Generator Implementation Details.

4.4 Experimenting with Prompt Variation Quality

Once the Regression Head Model is trained, we use it to assess how targeted variations of base prompts affect expected output quality. To do this, we provide our prompt generator with specific instructions during inference and evaluate the results using the Regression Head Model. In this paper, we explore the following types of prompt variations.

Improvement-Focused Variation: We instruct the prompt generator to revise base prompts in ways anticipated to yield higher quality outputs.

Typos: We test the impact of typos by introducing spelling errors while preserving the original structure and wording.

Capitalization: We also examine the impact of capitalization by converting all letters to uppercase, again keeping the base structure intact.

Structural Cues: We append short instructions to the end of the base prompt suggesting how the content should be structured, without altering the core content itself.

Politeness: We modify the base prompts to include polite phrases, such as "Please" and "Thank you."

Task Emphasis: We add an additional sentence highlighting key aspects of the prompt, such as emphasizing the topic, without modification of any of the existing words in the base prompt.

Partial Sentence Condition: We remove filler words to condense the prompt to its essential content, while preserving the original key words and base structure.

Detailed instructions for how each type of prompt variation was generated are provided in Appendix E.

5 Analysis

5.1 Analysis on Synthetic Data Generation

Figure 3 evaluates the synthetic data generated from 300 base prompts. It compares the base prompt scores, sorted from lowest to highest, against the distribution of scores from their corresponding prompt variations.

We observe that a portion of base prompts, particularly those with lower scores, were significantly outperformed by their variations, as indicated by the mean variation scores and IQRs being higher than the base scores. In contrast, higher-scoring base prompts tended to have prompt variations with means and IQRs below the base prompt scores.

[0]One trial was omitted due to failure to produce non-NaN values during training. We suspect this was a result of exploding gradients, characteristic of LoRA fine-tuning.

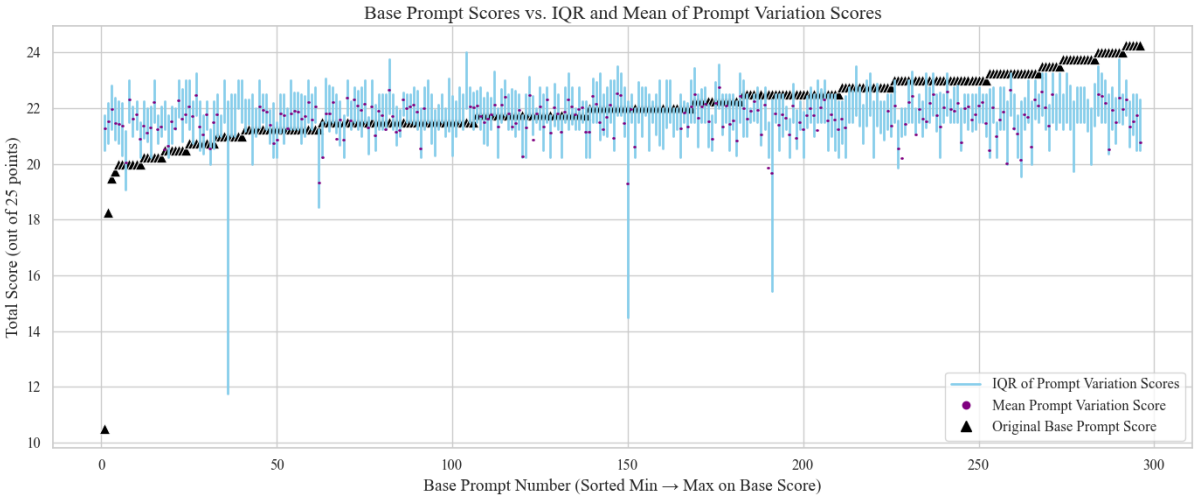

Figure 3: Comparing base prompt scores with inter-quartile range and mean of prompt variations

The graph suggests that no matter the quality or topic of the original base prompt, their corresponding prompt variations generally produce outputs that score within a consistent band. Therefore, our findings related to prompt variations are generalizable over all our base prompts.

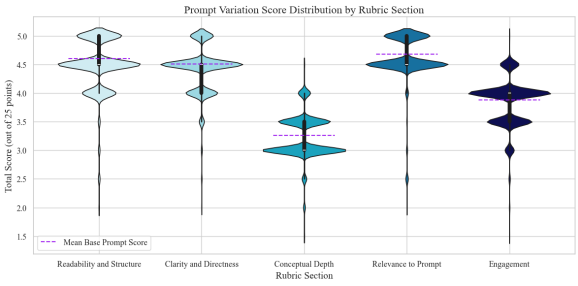

Figure 4: Mean scores for each rubric section

In addition, Figure 4 illustrates the distribution of prompt variation scores across the five rubric sections.

In the *Engagement* category, we observe that the majority of variation scores were higher than the mean base prompt score, which indicates that many prompt variations may have positively modified prompts to be more engaging for third-graders.

Moreover, the fact that a substantial number of prompt variations scored lower than the mean base prompt in both the *Relevance to Prompt* and *Readability and Structure* sections suggests that varying prompts may cause outputs to stray away from readability and base prompt focus. This observation motivated us to explore the impacts of *Emphasis* and *Structural Queues* in our prompt variation experimentation (see Experimenting with Prompt Variation Quality).

Lastly, our analysis reveals that *Conceptual Depth* exhibits lower scores overall compared to the other sections, but this may be due to our specific task, which focused on explaining educational concepts to third-graders.

5.2 Analysis on Model Training

In this section, we examine the results of our Regression Head Model training and report our results in Figure 5. Note that here, the RMSE is reported after n epochs, as represented by the x-axis. Thus, epoch = 0 represents the baseline RMSE before any training is applied.

Progression of Training: We observe that across all 9 trials, the training starts at around 22–23 RMSE and rapidly decreases to around 1.1–1.3. Considering that our rubric scores range between 9 and 25 points (out of 25 total), this final RMSE represents a small error relative to the range. We conclude that the Regression Head Model is able to strongly predict validator scores.

Impact of Hyperparameters: We note that while hyperparameter choices (LoRA rank, alpha, dropout, learning rate, and batch size) influence the rate at which the RMSE falls, particularly during the first epoch, they do not cause substantial differences in final model quality. Almost all trials achieve a final RMSE between 1.1 and 1.3, suggesting that the Regression Head Model is robust to a range of hyperparameter settings within the tested ranges.

Lack of Overfitting: Across trials, the Train, Validation, and Test RMSE curves closely track

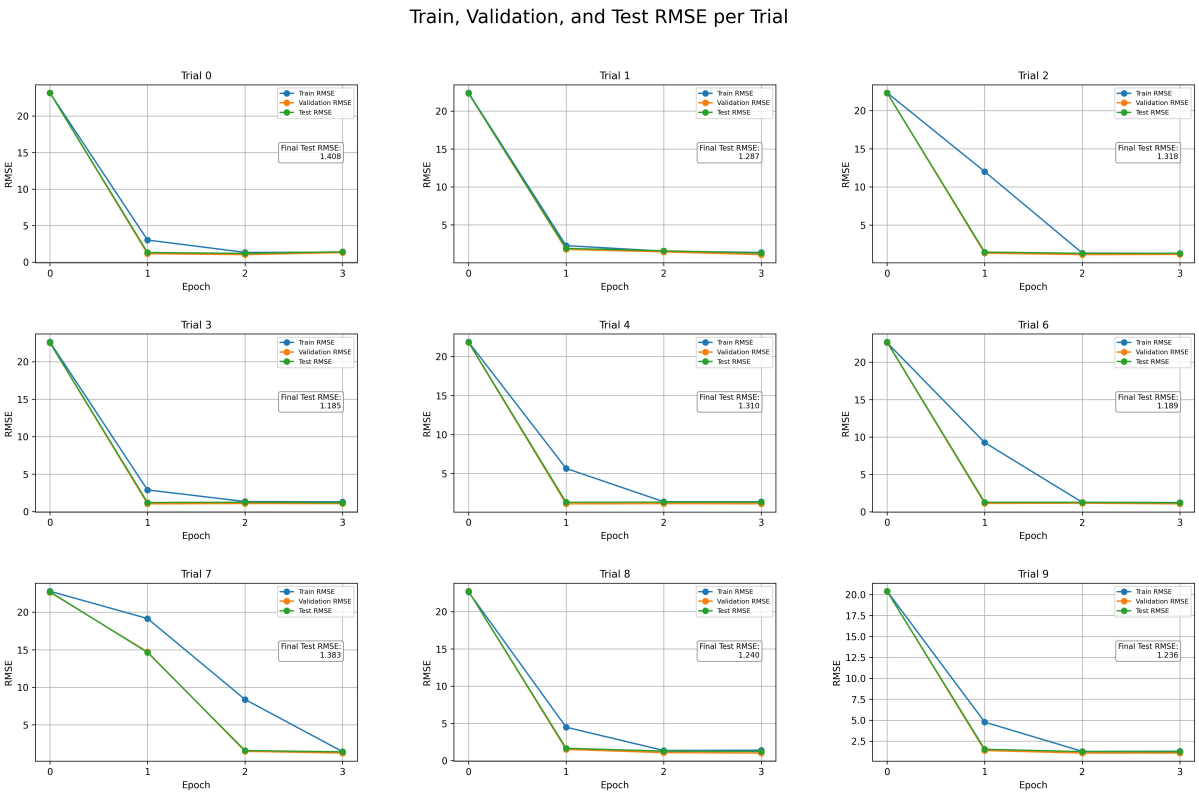

Figure 5: Results from Training the Regression Head [1]

one another throughout training. There is no significant divergence between training and evaluation losses, even after multiple epochs. This indicates that the model generalizes well to unseen data and that overfitting is minimal under the current training setup.

Epoch Convergence: In almost all trials, major improvements in RMSE occur within the first epoch. By the end of the second epoch, RMSE values plateau, and further training beyond two epochs yields minimal additional gains. This suggests that the Regression Head Model converges rapidly, and extended training is unlikely to provide meaningful improvements in performance.

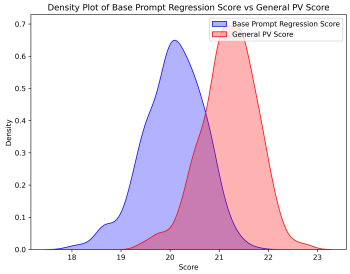

Figure 6: Density distribution of base prompt regression scores and general prompt variation (PV) scores.

5.3 Analysis on Prompt Variations

On our held-out test set of 600 base prompts, we generated an Improvement-Focused Variation for each and compared their Regression Head Model scores (Figure 6, Figure 10). In Figure 6, the prompt variation distribution shifts rightward, suggesting overall quality improvement. Figure 10 shows that lower-scoring base prompts benefited more from variations than higher-scoring ones.

We further analyze score distributions (Figure 7) and ranked comparisons (Figure 11), and summarize our observations below:

Typos: The distribution of typo-riddled prompts is shifted left and less peaked than that of the base prompt score distribution. The range of scores for prompt variations is relatively wide, with a large shift downwards in scores for higher-scored base prompts, implying that typos have a negative impact on the prompt quality based on our rubric with a stronger impact on originally well-scored prompts.

Capitalization: The distribution of capitalized prompts shifted significantly left from that of base prompts. The mean of the prompt variations distribution is approximately 3 points less. We observe that regardless of base prompt, the negative impact

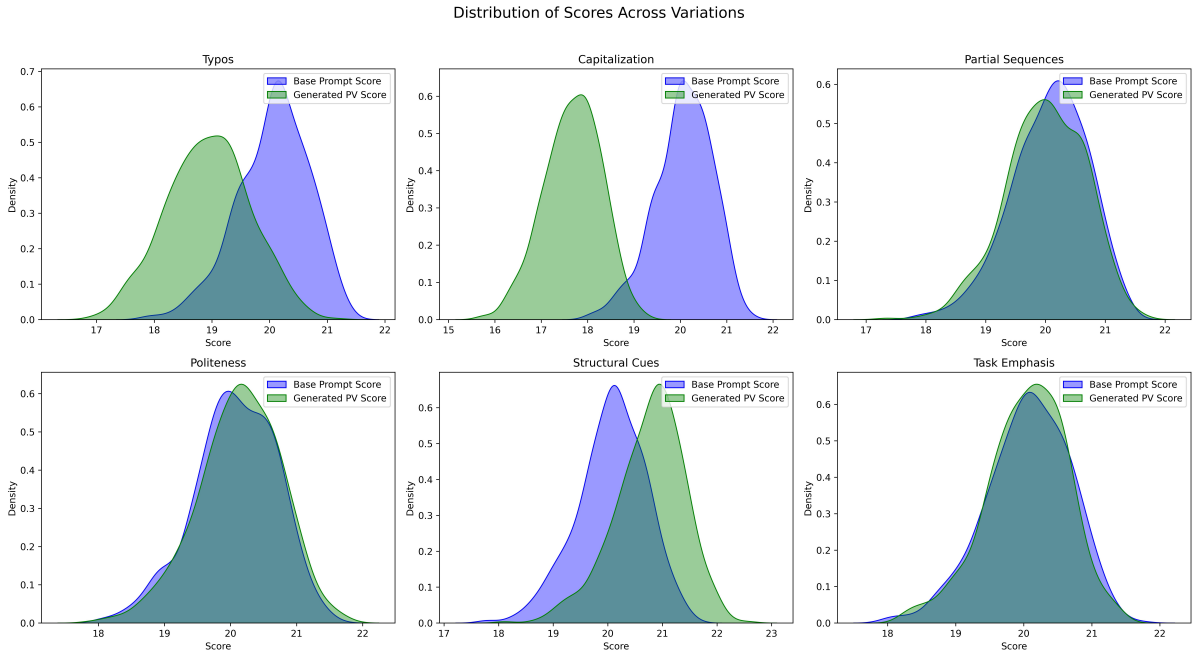

Figure 7: Comparative analysis of our different variation score distributions to base prompt score distributions.

is approximately the same.

Partial Sequences: While the distribution of prompt variations structured as partial sequences is very similar to that of the base prompt scores, we see that it is slightly shorter and shifted slightly to the left. Their means are relatively the same with a slight decrease for prompt variations. This suggests that using partial sequences does not have significant impact on prompt quality.

Politeness: We see that the distribution of polite prompt variation scores is slightly shifted to the right of the base prompt score distribution, implying a slight positive impact of polite tone in prompting. Also, the range of prompt variation scores is slightly wider than other neutral changes.

Structural Cues: The distribution of prompt variations for structural cues changes significantly to the right of the base prompt scores, with the mean increasing by around 0.7 points. Moreover, its general trendline of scores seems to impact the base prompt scores all relatively equally regardless of quality, showing structural cues have a consistently positive but generally smaller impact.

Task Emphasis: The distribution of prompts that show an emphasis on the task is very similar to the base prompt score distribution, with a bit of a larger peak with a shift to the right. It seems that as base prompt scores increase, the range of prompt variation scores seem to widen. It seems then that the impact of task emphasis is not significant and relatively unpredictable, particularly for already well-made prompts.

6 Limitations

Our team had trouble outputting the exact amount of base prompts and prompt variations we wanted due to constraints highlighted in Appendix C. Even though we enabled early stopping, the LLMs did not always generate the exact number of prompts we specified. Therefore, the training dataset for the Regression Head Model did not always consist 40 prompt variations for every base prompt, which may have introduced some bias in model training.

Despite generating 6 kinds of variations, the Prompt Generator may not fully capture the diversity of possible high or low-quality prompts that could impact model performance.

Only having one task which we tested PROSE on reduces the generalizability of our results. So far we have only tested PROSE in the context of our chosen task (learning guides for third-graders).

Finally, in our current implementation, we only use 2 Validator Models, but would ideally like to scale up to 3 or more to avoid any bias in the scoring of our model outputs.

7 Discussion

We now discuss our main takeaways from our current work, and present other takeaways in the Ex-

tended Discussion.

Prompt variation quality is generalizable across topics. Regardless of an original base prompt's quality or topic, corresponding variations produced consistent scores, suggesting that variation quality is largely independent of prompt structure or content and supporting aggregated analysis.

Most predicted hypotheses hold, with exceptions in capitalization and partial sequences. As hypothesized, prompt variations involving *Structural Cues* and *Task Emphasis* improve output scores, while *Typos* cause a negative shift and *Politeness* remains neutral. However, *Capitalization* unexpectedly shows a negative impact, and *Partial Sequences* remains neutral despite predictions of a decline.

Prompt variations have learnable features which allow a model to infer main model output quality. Analysis reveals that prompt variations exhibit learnable and predictive features that allow the Regression Head Model to effectively infer output quality changes. The model was able to distinguish between different prompt variations based on subtle semantic or structural cues, suggesting that variations are not random noise but contain informative patterns.

Validator Models are rubric-sensitive. We noticed that there were differences in scoring across rubric sections upon generating prompt variations. Thus, the quality of the synthetic dataset is heavily reliant on the credibility of the rubric provided as well as the section weights.

8 Future Work

Building on our work in this paper, we would like to complete the PROSE framework; we will train the Prompt Generator to automatically refine any given user prompt into an optimized base prompt.

We would also like to experiment with other kinds of prompt variations, evaluating the effects of changes on the predicted output score. This could include ablations such as variation of syntax, introducing errors, and exploration of punctuation.

Moreover, we would like to add an additional Validator Model to ensure that there is no bias present in implementing the rubric scoring system.

Lastly, expanding our framework to domains outside of third-grade learning guides and to tasks like text summarization would allow us to demonstrate the effectiveness of PROSE as a new method for prompt optimization.

9 Conclusion

In this work, we propose PROSE, a task-generalizable framework for prompt refinement and optimization through scored evaluation. By generating a synthetic dataset, training a Regression Head Model, and analyzing the effects of targeted prompt variations, we demonstrate that prompt structure significantly impacts model output quality. Our results show that simple prompt modifications, such as adding structural cues or correcting typographical errors, can significantly enhance or reduce output quality. Furthermore, we find that lightweight fine-tuning with LoRA enables effective regression modeling of prompt quality with minimal overfitting. Looking ahead, we aim to complete the PROSE pipeline by training a Prompt Generator, expanding our framework across tasks, and further refining automated prompt optimization strategies.

10 Division of Labor

Over the course of our project, we distributed the work evenly across our team, with each member taking lead on certain tasks and the other members supporting where needed.

Bandla took the lead on organizing the initial code base, building the Github structure, and designing data structures. Rinehart took the lead on brainstorming our framework and project flow based on background research. Jian led the writing of the abstract report and creating visualizations for the PROSE methodology. Jian wrote the scripts for base prompt generation, Chang wrote the prompt variation generation script, Bandla wrote the Main Model inferencing script, and Rinehart wrote the validator scoring script. Bandla and Rinehart collaborated on the Regression Head Model training scripts. Jian and Chang were responsible for generating the entire training and test datasets. Jian designed the figures and analyzed the generated data. Bandla trained and tested the Regression Head Model with the full synthetic dataset. Chang designed the figures and analyzed the regression head results. Bandla and Chang perfomed comparative analysis of the different prompt variation score distributions.

References

Yong Cao, Li Zhou, Seolhwa Lee, Laura Cabello, Min Chen, and Daniel Hershcovich. 2023. Assessing cross-cultural alignment between chatgpt and human societies: An empirical study.

Yuyan Chen, Zhihao Wen, Zhengyu Chen Ge Fan, Wei Wu, Dayiheng Liu, Zhixu Li, Bang Liu, and Yanghua Xiao. 2024. Mapo: Boosting large language model performance with model-adaptive prompt optimization.

Jiale Cheng, Xiao Liu, Pei Ke Kehan Zheng, Hongning Wang, Yuxiao Dong, Jie Tang, and Minlie Huang. 2024. Black-box prompt optimization: Aligning large language models without model training.

Cheng-Han Chiang and Hung yi Lee. 2023. Can large language models be an alternative to human evaluations?

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. 2024. LLM-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13806–13834, Bangkok, Thailand. Association for Computational Linguistics.

Tyler McDonald, Anthony Colosimo, and Ali Emami Yifeng Li. 2024. Can we afford the perfect prompt? balancing cost and accuracy with the economical prompting index.

Huyen Nguyen, Haihua Chen, Lavanya Pobbathi, and Junhua Ding. 2024. A comparative study of quality evaluation methods for text summarization.

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

Hassan Shakil, Atqiya Munawara Mahi, Phuoc Nguyen, Zeydy Ortiz, and Mamoun T. Mardini. 2024. Evaluating text summaries generated by large language models using openai's gpt.

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.

Ziqi Yin, Hao Wang, Kaito Horio, Daisuke Kawahara, and Satoshi Sekine. 2024. Should we respect llms? a cross-lingual study on the influence of prompt politeness on llm performance.

Yongchao Zhou, Andrei Ioan Muresanu adn Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Zhenqiang Gong, and Xing Xie. 2024. Promptrobust: Towards evaluating the robustness of large language models on adversarial prompts.

A Terminology

In this section, we define key terminology relevant to our work, which we will reference throughout the paper:

- **Prompt Topic Model**: A model that generates 50 different educational topics to use for base prompt generation.
- **Base Prompt Model**: A model that generates a large set of semantically equivalent but different base prompts for many topics.
- **Base Prompt**: The original prompt before any variations or optimizations.
- **Prompt Variation**: A modified version of the base prompt that differs in structure, wording, or emphasis, while preserving the original intent.
- **Prompt Variation Model**: A model that is inferenced to generate Prompt Variations of a given base prompt.
- **Main Model**: The target language model that generates responses based on a given base prompt or prompt variation. We fix our Main Model during our research as an experimental variable.
- **Prompt Optimization Model**: A model trained to generate optimized variations of a given base prompt.
- **Validator Model**: A model that uses the **??** to evaluate the quality of the Main Model's output.

B Base Prompt Topics

We generated 40 base prompts for each of the 50 topics produced through our topic generation process, oriented around Example Task. Below is a sample of 10 topics:

1. Weather and Climate
2. Basic First Aid
3. Basic Computer Skills
4. Graphs and Charts

5. Design and Prototyping
6. Energy and Motion, Health and Wellness
7. Earth's Resources
8. Measurement Units
9. Language Arts
10. Basic Map Reading

All 50 topics can be found in the GitHub repository provided.

C Base Prompt Batching

We started off with the intent to generate a base prompt dataset significantly larger than our target of 900 base prompts so that we would have spare prompts in case we needed more data to train the prompt optimization model. Due to HPC constraints, we decided to batch our base prompts. To execute this, we first tried 20 batches of 100 prompts each to ultimately produce 2000 base prompts total, but faced many bottlenecks, including numbers of prompts produced as well as prompt quality and semantic variation. We ultimately decided on an approach where we generated 50 topics from the Base Prompt Model, and then generated 40 prompts within each topic. This approach allowed us to effectively generate topically varied and consistent numbers of Base Prompts.

However, over multiple trial runs, we noticed that the Base Prompt Model continuously struggled to output the 40 prompts per batch we requested. As a result, we ended up with a raw dataset of 1,790 base prompts. To ensure consistency for comparison, we asked the model to ensure that every base prompt include the phrases "guide" and "third-grade," and after filtering for this criteria, we were left with a master dataset of 1,766 base prompts, of which we selected 300 for training through prompt variations and a further 600 for testing.

D Model Instructions: Synthetic Data Generation

Data Component	Model Inference Instruction
Base Prompt Topics	**System Role:** "You are an expert in elementary education. Generate broad, age-appropriate topic areas for third-grade students (ages 8–9)." **Content Template:** "Generate EXACTLY 50 broad educational topic areas suitable for third-graders. Each topic should be general enough to allow for many sub-topics and should be relevant worldwide. Avoid country-specific topics like U.S. history, flags, or holidays, and avoid overly narrow or repetitive topics. Example topics: 'Math Fundamentals', 'Animal Habitats', 'Reading Skills', 'Weather and Climate', 'Healthy Living'. Output a single JSON array of double-quoted strings, no explanations. Format: ["topic1", "topic2", ...]"
Base Prompts	**System Role:** "You are an expert in prompt engineering. Your task is to generate high-quality prompts for a language model that will be used to create learning guides for third-grade students." **Content Template:** "Generate EXACTLY {num_prompt} different prompts that ask a language model to generate a learning guide for a third-grade student, all around the topic of {topic}. Each prompt must: (1) include the word 'guide' and either 'third-grade' or 'third-grader' and (2) be between 5–25 words long. Keep the structure of the prompts diverse and include verbs in each prompt. All topics must be specific and age-appropriate for third-grade students in a U.S. school setting. Output should be a single JSON array of double-quoted strings with no explanation or commentary. Format: ["prompt1", "prompt2", ...]"
Prompt Variations	**System Role:** "You are an expert in prompt engineering. Your task is to write alternative LLM prompt phrasings for the base instruction: '{bp_str}'. Your variations should instruct a language model (not a child) to generate learning guides for third-grade students." **Content Template:** "Return exactly one JSON array of {num_prompt_variations} prompt variations for the base prompt input. Each prompt must be a instruction to an LLM for generating a learning guide for third-grade students. In each variation, you must include the exact word "guide", and include either "third-grade" or "third-grader". Do **not** use synonyms like 'manual', 'walkthrough', 'tutorial', or 'primer'. Every prompt must contain the literal word "guide". IMPORTANT: Do not add any explanations or formatting outside the JSON array. Output format: ["prompt_variation_1", "prompt_variation_2", ...] Make sure the output is formatted as a list (no headers)."
Main Model Output	**System Role:** "You are generating learning guides for third-graders. Make sure to keep the response between 200-400 words." **Content Template:** "pv_str_template"
Validation Scores	**System Role:** "You are an expert evaluator tasked with scoring third-grade learning guides generated by a large language model using a strict rubric. You must judge the quality of the output solely based on the rubric criteria and provide scores with no generosity or assumptions." **Content Template:** "The base prompt was: {base_prompt_str}.\nThis is the output: {main_model_output}.\nGrade the output in relation to the base prompt. Use the following rubric. Assign a numerical score for each of the categories based on specific criteria. Return ONLY a JSON array of [section1_score, section2_score, section3_score, section4_score, section5_score] corresponding to the number of sections. Do not return anything else.\nRubric: {rubric_text}"

E Model Instructions: Analysis on Prompt Variation

E.1 Improvement-Focused Variation

```
# Example of the desired behavior
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
example_variation = "Createe a leawrning guide for a third-grader on the congcept of grapvity."

# Instruction with an example
instruction = (
    f"Rewrite the following prompt by introducing 3-5 typos. Everything about the rewritten prompt should be the same as the original prompt, except for the typos. "
    f"The rewritten prompt MUST include the EXACT phrase 'learning guide' and the word 'third-grade' or 'third-grader'. "
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
    f"Example Base Instruction: {example_base_prompt}\n"
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

E.2 Typos

```
# Example of the desired behavior
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
example_variation = "Createe a leawrning guide for a third-grader on the congcept of grapvity."

# Instruction with an example
instruction = (
    f"Rewrite the following prompt by introducing 3-5 typos. Everything about the rewritten prompt should be the same as the original prompt, except for the typos. "
    f"The rewritten prompt MUST include the EXACT phrase 'learning guide' and the word 'third-grade' or 'third-grader'. "
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
    f"Example Base Instruction: {example_base_prompt}\n"
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

E.3 Capitalization

```
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
    example_variation = "CREATE A LEARNING GUIDE FOR A THIRD-GRADER ON THE CONCEPT OF GRAVITY."

instruction = (
    f"Make every letter of the following base prompt capitalized. Everything about the rewritten prompt should be the same as the original prompt, except for every letter being capitalized."
    f"The rewritten prompt MUST include the EXACT phrases 'LEARNING GUIDE' and the word 'THIRD-GRADE' or 'THIRD-GRADER'. "
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
    f"Example Base Instruction: {example_base_prompt}\n"
```

```
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

E.4 Partial Sequences

```
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
example_variation = "Create learning guide for third-grader gravity."

instruction = (
    f"Rewrite the prompt by removing unnecessary function words (such as articles, conjunctions, and prepositions) while preserving key phrases and essential meaning."
    f"Make sure to use the same key words and phrases as the original prompt."
    f"The rewritten prompt MUST include the EXACT phrase 'learning guide' and the word 'third-grade' or 'third-grader'."
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
    f"Example Base Instruction: {example_base_prompt}\n"
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

E.5 Politeness

```
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
example_variation = "Please create a learning guide for a third-grader on the concept of gravity. Thank you so much!"

instruction = (
    f"Rewrite the following prompt with a polite tone, retaining the same semantics of the base prompt."
    f"The rewritten prompt MUST include the EXACT phrase 'learning guide' and the word 'third-grade' or 'third-grader'."
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
    f"Example Base Instruction: {example_base_prompt}\n"
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

E.6 Structural Cues

```
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
example_variation = "Create a simple, step-by-step learning guide for a third-grader on the concept of gravity."

instruction = (
    f"Rewrite the following prompt and add instructions for specific structural tips typical in creating instructive and helpful learning guides. Everything about the rewritten prompt should be the same as the original prompt, expect for the added words that provide cues"
    f"The rewritten prompt MUST include the EXACT phrase 'learning guide' and the word 'third-grade' or 'third-grader'. "
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
```

```
    f"Example Base Instruction: {example_base_prompt}\n"
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

E.7 Task Emphasis

```
example_base_prompt = "Create a learning guide for a third-grader on the concept of gravity."
example_variation = "Create a learning guide for a third-grader on the concept of gravity. Important: make sure to talk only about gravity"

instruction = (
    f"Rewrite the following prompt by re-emphasizing the topic at hand. Everything about the rewritten prompt should be the same as the original prompt, except for the new words expressing added emphasis."
    f"The rewritten prompt MUST include the EXACT phrase 'learning guide' and the word 'third-grade' or 'third-grader'."
    f"Do not include any additional text or explanation. Return only the rewritten prompt.\n\n"
    f"Example Base Instruction: {example_base_prompt}\n"
    f"Example Rewritten Prompt: {example_variation}\n\n"
    f"Now, rewrite the following base instruction:\n"
    f"Base Instruction: {base_prompt}\n"
    f"Rewritten Prompt:"
)
```

F Rubric

Category / Score	Readability/ Structure (20%)	Clarity/ Directness (20%)	Conceptual Depth (10%)	Relevance to Prompt (40%)	Engagement (10%)
5	Exceptionally well-structured, logical progression, and highly readable. Paragraphs and sections flow seamlessly.	Exceptionally clear, precise, and free of unnecessary complexity. Ideas are conveyed with straightforward and unambiguous language.	Provides a deep, nuanced, and accurate explanation. Covers all key aspects while adding insightful details.	Fully addresses the prompt with a precise, highly relevant explanation. No unnecessary tangents.	Exceptionally engaging, dynamic, and thought-provoking. Captures and holds attention effortlessly.
4	Very well-organized with clear transitions, but minor areas could be refined for even better readability.	Very clear and easy to understand, with only minor areas that could be slightly more direct.	Strong conceptual depth with minor areas that could be slightly expanded or refined.	Strong relevance, with only minor deviations or extra details that don't distract from the main focus.	Very engaging with strong explanations and examples, though minor improvements could enhance it.
3	Generally well-structured, but occasional awkward transitions or uneven paragraphing.	Mostly clear, though some sentences could be refined for conciseness or improved clarity.	Covers the main ideas thoroughly but may lack deeper insights or explanations in some areas.	Mostly on-topic, but includes some unnecessary information or slight deviations.	Generally engaging, but may lack variety in phrasing or examples that would make it more compelling.
2	Mostly organized, but some parts feel choppy or slightly disjointed.	Generally understandable but contains occasional vague or overly complex phrasing.	Mostly correct and informative, but some points feel slightly underdeveloped or oversimplified.	Generally relevant, but includes noticeable sections that stray from the core question.	Moderately engaging, though some sections feel dry or could use better storytelling.
1	Some structural weaknesses make it harder to follow, but the overall message is still discernible.	Somewhat clear, but requires extra effort to interpret certain sections.	Covers basic aspects but lacks depth in key areas. Some minor conceptual gaps.	Somewhat relevant, but parts of the response feel off-topic or too generalized.	Somewhat engaging, but requires additional effort to maintain reader interest.

Figure 8: The rubric used for the learning guide generation task

G Example of Synthetic Data

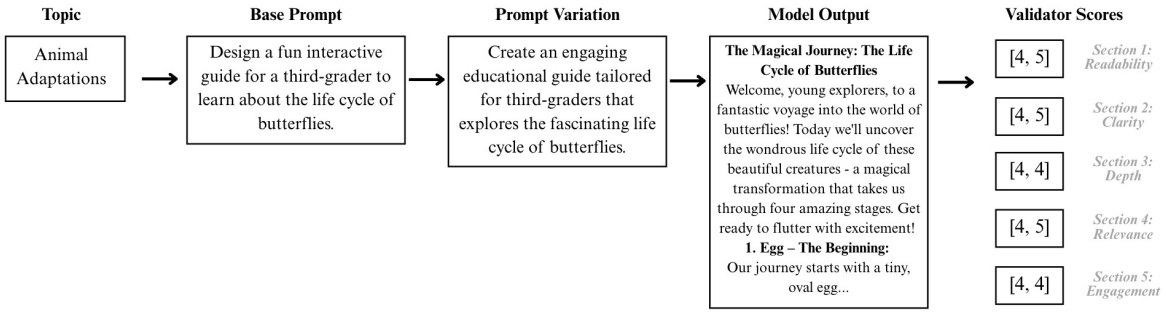

Figure 9: An example of the synthetic data generated

H Additional Figures

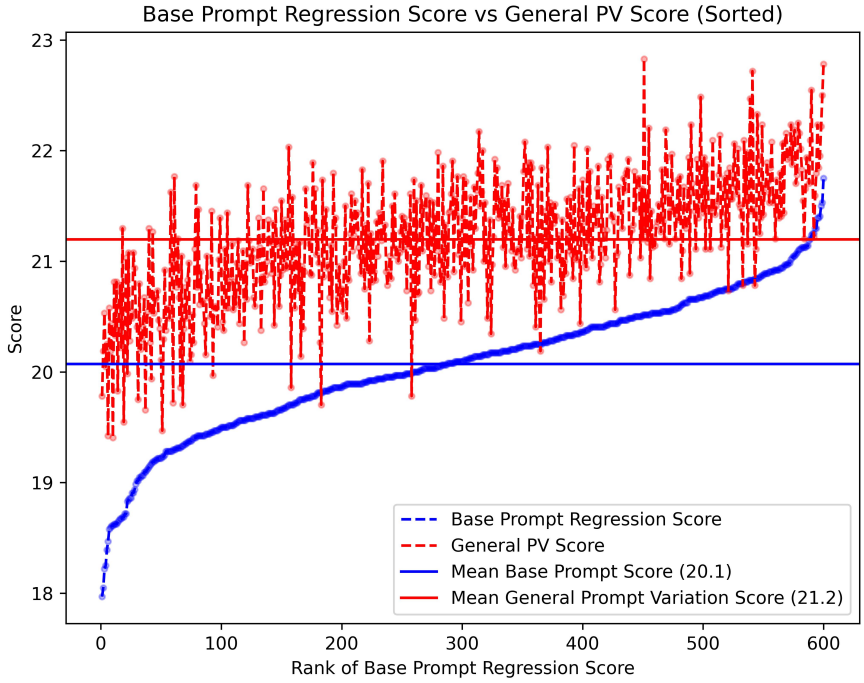

Figure 10: Comparison of sorted base prompt regression scores and corresponding general prompt variation (PV) scores.

I Prompt Generator Implementation Details

The Prompt Generator is designed to output optimized prompt variations based on a given base prompt. For each base prompt, the Prompt Generator generates n prompt variations, which are then scored individually by the trained Regression Head Model. To calculate a reward for each variation, we compute the difference between the Regression Head Model's predicted score of the prompt variation and the predicted score of the original base prompt. This delta serves as the reward signal: positive deltas indicate an improvement over the base prompt, while negative deltas suggest a degradation in prompt quality.

We conceptualize the training procedure as a reinforcement learning problem, where the Prompt Generator is optimized using Proximal Policy Optimization (PPO). PPO allows for stable training by constraining updates to the generator model. Similar to the Regression Head Model, we plan to apply Low-Rank Adaptation (LoRA) during training to enable parameter-efficient fine-tuning while preserving most of the underlying pre-trained model weights.

J Extended Discussion

Prompt variations generally improved engagement overall. Our prompt variations frequently improved the *Engagement* rubric scores on model outputs compared to their respective base prompts.

Simple variations in prompts can create a difference in quality. Through prompting the Prompt Variation Model to improve the base prompt in domain- and task-agnostic ways, we were able to consistently improve the Regression Head Model's predicted scores. Thus, there exist simple and straightforward heuristics and prompting best-practices to marginally improve prompt quality.

Hyperparameter tuning does not seem to have significant impact. Our experiments suggest that hyperparameter tuning, within the explored search space, does not significantly alter model performance.

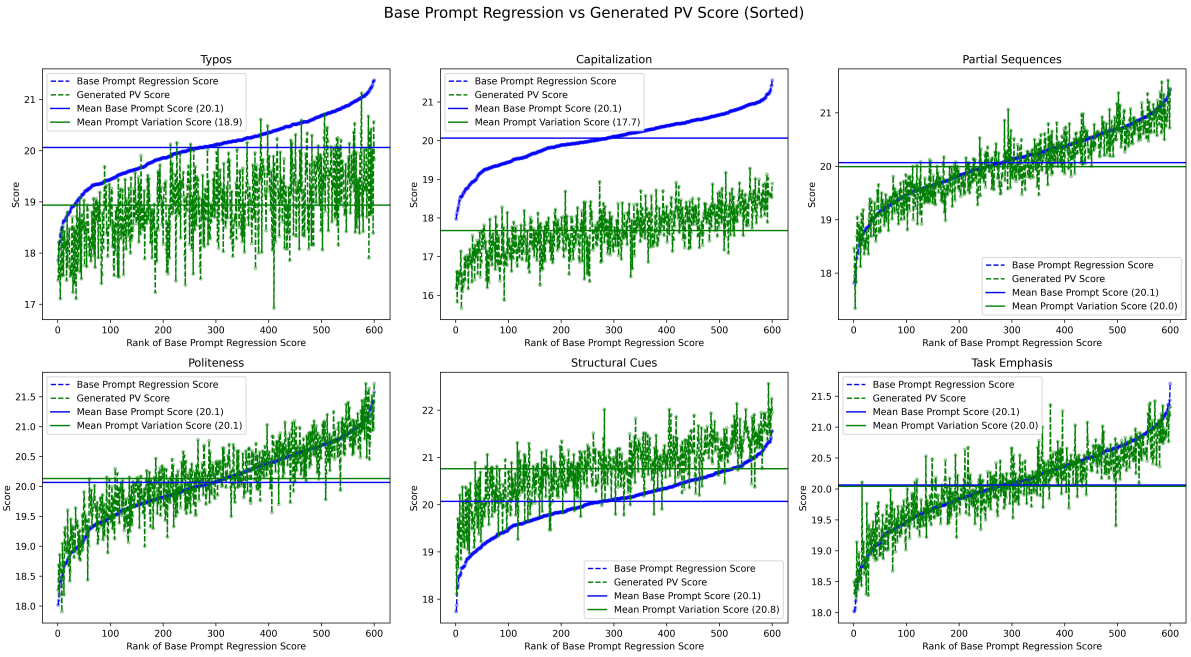

Figure 11: Comparison of base prompt regression scores and associated Prompt Variation scores across different variation types. For each variation type, prompts are ranked by their base prompt regression score (x-axis, ascending order), and corresponding prompt variation scores are plotted. Mean scores for base prompts and prompt variations are shown as horizontal lines for reference.

Across multiple trials, variations in learning rate, batch size, dropout rate, and LoRA parameters resulted in only minor fluctuations in validation and test RMSE.

Using a trained regression head, scoring can reasonably be accomplished for different prompts. After fine-tuning, the model is able to reduce regression scores that correlate well with the expected quality of both base prompts and prompt variations. This shows that even lightweight adaptation via LoRA is sufficient for the model to capture meaningful distinctions in prompt quality.

Synthetic data generation focused on prompt variations is robust. Despite the inherent variability in the quality and structure of generated variations, the Regression Head Model was able to learn meaningful scoring functions without significant performance degradation. This indicates that even when training on artificially created variations — rather than manually curated datasets — the model is able to generalize and extract useful signal.

The Regression Head Model may reasonably be applied to other score-based language tasks. Training on synthetically generated data leads to significant improvements over baseline pre-trained models, demonstrating the model's flexibility and the value of task-specific adaptation.