Noise and Nuance: Impact of Input Noise on Translation Accuracy in Transformer Models

Nandan Sarkar
Yale University
New Haven, CT, USA
nandan.sarkar@yale.edu

Aditya Kulkarni
Yale University
New Haven, CT, USA
aditya.kulkarni@yale.edu

Andrew Pan
Yale University
New Haven, CT, USA
a.pan@yale.edu

1 Introduction

Neural language models, particularly those based on the transformer architecture, have been pivotal in achieving state-of-the-art results in machine translation, text summarization, and other natural language processing (NLP) tasks. However, the robustness of these models to input perturbations remains a challenge, especially in practical applications where input noise is inevitable. This project addresses this gap by investigating the resilience of transformer-based language models against variations in input embeddings. Through this project, we aim to contribute to the ongoing discussion on improving the robustness and reliability of language models, offering insights that could facilitate the development of more resilient NLP systems.

Our approach focuses on the injection of noise into the embeddings of a transformer encoder and examining the decoder's capability to process these perturbed inputs effectively. By manipulating the encoder outputs, this project directly tests the resilience of the decoder component of the transformer architecture in maintaining translation accuracy under varied input conditions. Our experimental setup includes two main studies: the first evaluates the impact of noise on the quality of translations using standard metrics like BLEU [1] and ROUGE [2], while the second assesses the semantic fidelity of translations by comparing the cosine similarities between original and noise-perturbed embeddings. The dataset for our experiments was curated by prompting ChatGPT [3], and comprises of 1500 English phrases of varying lengths with a broad spectrum of sentiments and grammatical structures, which we then translated from English to German using Google Translate [4]. Thus, these phrases provide a robust testing ground.

This paper is structured as follows: Section 2 reviews related works that have informed our approach. Section 3 details the dataset curation process, and Section 4 discusses the model choice. Section 5 describes our experimental methodology, followed by a discussion of the results in Section 6. We conclude in Section 7 with an overview of our findings, limitations, and suggestions for future work. [1]

2 Related Works

2.1 Towards Robust Word Embeddings for Noisy Texts [5]

This paper proposes a novel extension to the skipgram model aimed at improving word embeddings' robustness to noisy texts commonly found in social media platforms. By introducing "bridge-words"-artificial words added to strengthen the similarity between standard words and their noisy variants-the authors demonstrate enhanced performance of their new embeddings over traditional models in both intrinsic and extrinsic evaluation tasks, particularly in handling noisy texts without requiring preprocessing steps like spell checking or text normalization. This work directly aligns with our project by addressing the resilience of language model embeddings to input perturbations.

[1] Code is available *here*

Their method of integrating noisy variants during training offers a compelling approach to data augmentation.

2.2 Robust-to-Noise Models in Natural Language Processing Tasks [6]

This paper introduces the Robust Vectors (RoVe) model, which outperforms traditional models like FastText and Word2Vec across various NLP tasks by making word vector modeling robust to noise. This is achieved through an open vocabulary architecture that encodes words as sequences of symbols, allowing for the generation of embeddings for out-of-vocabulary words, including those with typos. Unlike existing models, RoVe specifically targets typographical errors by encoding each word as a bag of characters and separately encoding word prefixes and suffixes. This feature enables meaningful embeddings for unseen word forms in morphologically rich languages without explicit morphological analysis. Additionally, RoVe incorporates context dependency to produce embeddings that are informed by a word's immediate context, enhancing grammatical feature representation. This work provides an approach to handling unseen word forms and typographical errors and can inspire methods for data augmentation, particularly in enhancing training data for languages and domains where clean and comprehensive datasets are scarce.

3 Building our Dataset

3.1 Curation from ChatGPT

> Please generate 1500 phrases/sentences in English encompassing a range of sentiments, positive, negative likes, dislikes, adjectives, adverbs, and more. 750 of them should be short phrases/sentences, roughly 5-10 words each, while the other 750 of them should be longer phrases/sentences roughly 15-20 words each. Make each phrase/sentence distinct from another, and try not to utilize too many of the same words between phrases. Do not use similar sentence/phrase structures between phrases/sentences, make each sentence/phrase structure as unique as possible.

Figure 1: ChatGPT prompt used for generating the dataset

To build a diverse dataset for our experiments, we used OpenAI's ChatGPT 4. We opted to generate our own data to ensure a wide variety of linguistic structures and sentiments. This approach allowed us to precisely control the complexity and variability of the input. The broad spectrum of input types, with different levels of noise impact, allowed us to rigorously test the robustness of the translation model and evaluate the semantic integrity of the output.

We crafted a specific prompt to generate sentences that span a broad spectrum of sentiments, including positive, negative, likes, dislikes, as well as various adjectives and adverbs. To evaluate the model's performance across different input lengths, we requested 750 short phrases of approximately 5-10 words and 750 longer phrases of about 15-20 words. Each phrase was required to be unique in terms of vocabulary and syntactic structure, minimizing repetition and enhancing linguistic diversity.

The generation process involved producing 100 samples at a time – both short and long – from each prompt. After reviewing and approving each batch, we proceeded to request the next set of 100 samples. This iterative review and approval process ensured that the generated text met our specific requirements for diversity and complexity. After generating the text, we used the Google Translate API to translate the generated English phrases into German. The final corpus, consisting of German inputs carefully curated through this method, served as the dataset for our subsequent experiments.

The text of the prompt given to ChatGPT to generate the data is presented in Figure 1.

3.2 Dataset Details and Analysis

As mentioned above, our dataset consists of 1,500 entries, evenly split into 750 short inputs and 750 long inputs. These entries cover a comprehensive range of sentiments – positive, negative, likes,

dislikes – and include a diverse array of adjectives and adverbs. The 750 short inputs in the dataset have an average word count of approximately 6.989 words and an average token count of 8.052, while the 750 long inputs average about 15.803 words and 17.596 tokens. This difference in length allows us to explore the model's performance across different input sizes.

4 Model Choice

We selected a German-to-English translation model for our experiments due to the extensive research and availability of high-performing, reliable models for this language pair within the machine translation community. This reliability is crucial for our experiments, allowing us to attribute any observed effects more accurately to the introduced noise variables, rather than to the intrinsic capabilities of the underlying model.

Specifically, we opted for the Helsinki-NLP/opus-mt-de-en model [7], a transformer-based encoder-decoder with 6 layers in each component. This model was chosen for its architectural robustness, the ability to independently manipulate the encoder and decoder, and its proven performance in translation tasks.

For translating our initial English dataset and the intermediate outputs from our second experiment (detailed later) into German, we utilized Google Translate. This choice was driven by Google Translate's reputation as one of the most advanced and state-of-the-art translation models currently available.

5 Experiments

5.1 Experiment 1

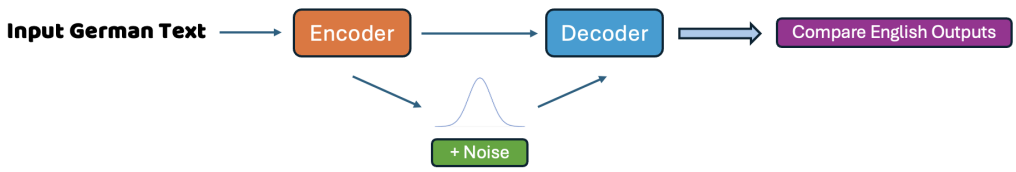

Figure 2: Schematic of Experiment 1

Experiment 1 was designed as a baseline experiment to evaluate how noise affects translation quality in the model. We started by encoding German text from our dataset into embeddings. To simulate varying degrees of real-world noise conditions, we introduced Gaussian noise with a mean of zero and standard deviations at several levels: $\{0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.5\}$. These specific levels were selected based on preliminary experiments, which indicated that smaller levels of noise had negligible impact on the outputs, while higher levels rendered the outputs completely unintelligible. We conducted four trials at each noise level to ensure the results were robust and reproducible.

In this experiment, we employed the `'generate'` method's capability to accept pre-computed encoder outputs, effectively bypassing the encoding step. This approach allowed us to directly evaluate the decoder's resilience to the noisy encoder outputs.

To quantify the impact of noise, we compared the translations from noisy embeddings with those from the original, clean embeddings using BLEU and ROUGE scores. These standard metrics in machine translation tasks measure the accuracy and quality of text, enabling us to systematically assess the degradation in translation quality across varying levels of noise.

Refer to Figure 2 for the schematic representation of Experiment 1.

5.2 Experiment 2

Experiment 2 builds on the findings of Experiment 1 by assessing the preservation of semantic content through translation cycles and noise perturbation. Similar to Experiment 1, we began with the same

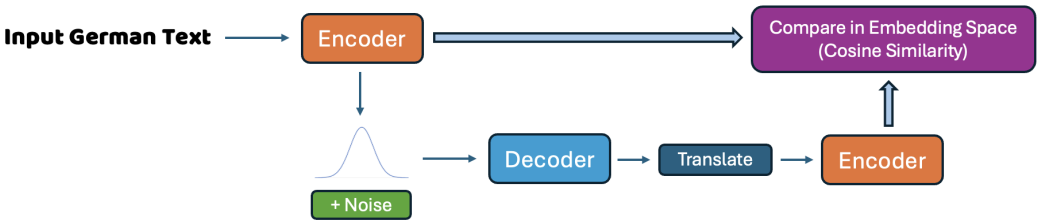

Figure 3: Schematic of Experiment 2

standard deviation levels for noise $\{0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.5\}$. After adding Gaussian noise to the German text embeddings, we translated these noisy embeddings into English using the model. This process again leveraged the `'generate'` method, which was configured to utilize the noisy pre-computed encoder outputs. Subsequently, the English translations were re-translated back into German using Google Translate, and then re-encoded to evaluate the preservation of semantic information. Due to hourly limit constraints on Yale's Grace cluster, this experiment utilized a subset of the data consisting of 500 short and 500 long phrases.

For analysis, the re-encoded embeddings were compared to the original embeddings using cosine similarity and cosine similarity squared. This comparison was simplified by employing a mean reduction technique, averaging the embeddings across the sequence to reduce the dimensionality of the data, thus facilitating a holistic comparison of the semantic content between the original and noise-perturbed translations. The cosine similarity scores provided a numerical value representing the similarity between the original and transformed texts, indicating the model's effectiveness in preserving semantic integrity under noisy conditions. We also squared the cosine similarity scores as a means of emphasizing differences. By squaring the similarity values, we enhance discrepancies, making it easier to identify and analyze subtle variations in semantic preservation. This approach helps highlight the impact of noise on the translation process, especially when changes are minimal but potentially significant.

Figure 3 provides the schematic of Experiment 2.

Note: For all experiments, we took advantage of Yale's Grace cluster by running a Jupyter server with 7 CPU cores, 27 GB memory per core, and 3 GPUs (6 hour limit).

6 Results and Discussion

6.1 Experiment 1

For Experiment 1, recall that we employ BLEU and ROUGE scoring to assess decoder performance relative to non-noisy reference outputs. The mean BLEU scores for each noise level across the test dataset and across the shorter and longer dataset splits are plotted in Figure 4. Likewise, the mean ROUGE-1, ROUGE-2, and ROUGE-L scores for each noise level across the test dataset and across the shorter and longer dataset splits are plotted in Figure 6 and Figure 7, respectively (see Appendix).

We observed a notably consistent trend in the BLEU scores, indicating a predictability to the model's sensitivity to perturbations in the input embeddings. As the standard deviation of the noise increased, as expected, there was a significant decrease in BLEU scores, demonstrating a degradation in the semantic fidelity of the translated outputs. This trend was particularly pronounced beyond the 0.20 standard deviation threshold, with nearly identical trends occurring across all four trials.

When comparing BLEU scores between short and long sentence inputs, we observed more interesting nuances in the model's response to noise. Shorter sentences exhibited greater initial resilience to noise, maintaining marginally higher BLEU scores up to a certain noise threshold, typically around 0.20 standard deviations. However, longer sentence inputs yield comparatively higher BLEU scores beyond this threshold. This suggests that while shorter sentences may initially fare slightly better under noisy conditions, the semantics of their translated outputs degrade at a faster rate.

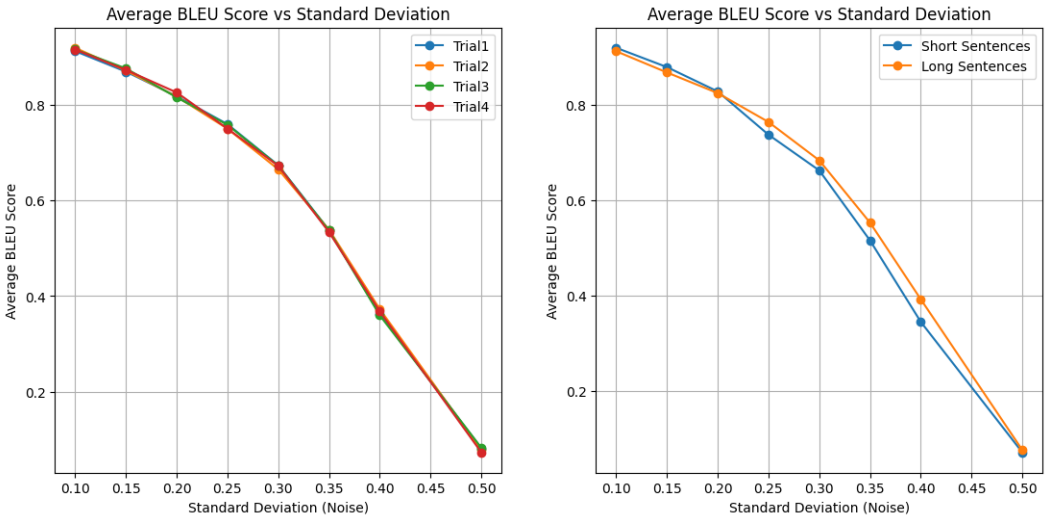

Figure 4: Mean BLEU Score Variation Across Trials (left) and Dataset Splits (right)

The mean ROUGE scores across the entire test dataset very closely echoed the trends observed in the BLEU scores, indicating a strong correlation between lexical accuracy and semantic fidelity in the translated outputs. This alignment reinforces the reliability of both metrics in evaluating translation quality and underscores the model's consistent response to input noise across different evaluation criteria.

On the other hand, the comparison of ROUGE scores between short and long dataset splits displayed a significantly more pronounced deviation than seen previously when comparing BLEU scores. Rather than overtaking shorter sentence inputs past a specific noise threshold, longer sentence inputs sustain a higher ROUGE score across all noise levels, with differences exceeding 0.1 at certain standard deviations. This reveals a notable disparity in the model's ability to preserve semantic content between short and long sentences. A potential reason is that longer sentences may exhibit greater semantic complexity, making them more resistant to the detrimental effects of input noise. This resilience to noise-induced degradation could be attributed to the richer contextual information present in longer sentences, which enables the model to maintain semantic coherence with more robustness.

6.2 Experiment 2

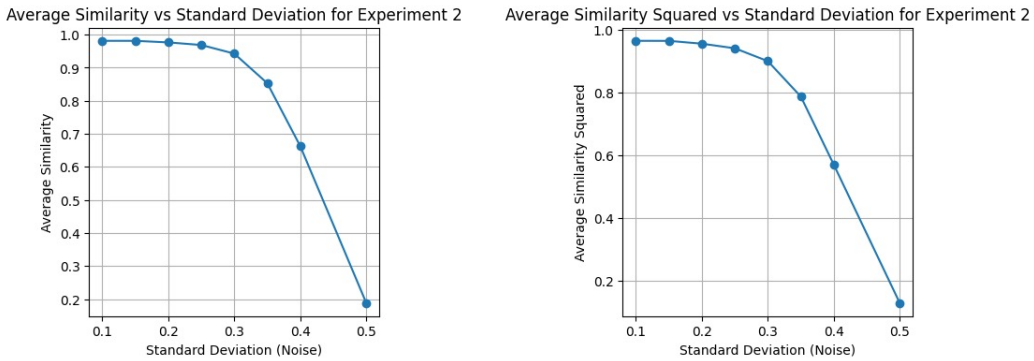

Figure 5: Mean Cosine Similarity (left) and Mean Cosine Similarity Squared (right)

For Experiment 2, recall that we utilize cosine similarity and cosine similarity squared to measure the distance between the re-encoded noisy embeddings and the original embeddings in the embedding space. The mean similarity scores for each noise level across the test dataset are plotted in Figure 5.

Again, the trends broadly resemble our results from Experiment 1. A notable distinction, however, is that the drop-off for Experiment 2 is more exponential in nature, with a more sustained and gradual initial decline followed by a more accelerated degradation beyond the 0.30 standard deviation threshold.

The presence of a 'critical threshold' in both Experiment 1 and Experiment 2 underscores a fundamental aspect of the model's behavior under varying levels of input noise. This threshold represents a measurable bound within which perturbations to the input do not significantly impact semantic content, indicating a degree of resilience in the translation process. We confirmed these findings by our own human evaluations on the model outputs across the multiple noise levels.

This critical threshold serves as a crucial delineation point, delineating between noise levels that have negligible effects on semantic fidelity and those that lead to discernible degradation in translation quality. Below this threshold, the model demonstrates a remarkable ability to maintain semantic coherence, even in the presence of moderate noise levels. This suggests that the model's underlying architecture and learning mechanisms are robust enough to withstand minor perturbations in the input without compromising the integrity of the translated outputs.

6.3 Additional Results

Table 1: Mean Token Count Differences Across Dataset Splits

Standard Deviation	Mean Token Count Differences (Short)	Mean Token Count Differences (Long)
0.10	0.024000	0.048000
0.15	0.066667	0.060000
0.20	0.104000	0.138667
0.25	0.201333	0.238667
0.30	0.844000	0.612000
0.35	1.702667	2.269333
0.40	10.874667	9.600000
0.50	82.506667	90.819760

We also computed the mean token count difference between the noisy and original, clean model outputs across the various standard deviation levels using the results from Experiment 1, which are listed in Table 1. Notably, the model tends to produce outputs with more tokens as the level of noise increases, with the difference growing exponentially relative to the noise's standard deviation.

7 Conclusion, Limitations, and Future Work

Our experiments provide valuable insights into the effects of input noise on translation accuracy and semantic fidelity in transformer-based language models. Experiment 1 demonstrated a clear correlation between increasing input noise levels and a decline in translation quality, as evidenced by decreases in BLEU and ROUGE scores. Notably, longer sentences exhibited greater resilience to noise-induced degradation, suggesting nuanced differences in the model's response based on input length. Experiment 2 highlighted the 'critical threshold' within which semantic integrity remains largely unaffected by noise, underscoring the robustness of the translation process. However, beyond this threshold, semantic preservation experienced an accelerated decline, emphasizing the importance of understanding and identifying these critical thresholds for model optimization.

Despite these insights, our study has limitations, as it primarily focuses on a specific transformer architecture and language pair. Additionally, the reliance on automated metrics may not fully capture semantic nuances, and the use of Google Translate introduces additional noise. Future research should explore a broader range of language pairs and transformer architectures to better understand cross-linguistic differences in model robustness. Investigating different types of input noise could also provide greater insights into semantic fidelity. Addressing these limitations will ultimately help advance our understanding of input noise's impact on transformer-based language models.

8 Contribution Statement

The project was a collaborative effort, beginning with the conception and planning of the idea and experiments by all team members. Nandan was responsible for compiling the dataset, while he and Andrew jointly developed the experimental code, with support from Aditya. Aditya also took the lead in analyzing the data and generating insightful plots. The final write-up was a collective endeavor, with each member contributing equally to writing and revising the document, ensuring a cohesive and thorough presentation of our findings.

References

[1] Papineni, K., Roukos, S., Ward, T., & Zhu, W. (2002) *BLEU: A Method for Automatic Evaluation of Machine Translation.* ACL.

[2] Lin, C. (2004) *ROUGE: A Package for Automatic Evaluation of Summaries.* ACL.

[3] OpenAI. (2023). GPT-4 (Generative Pre-trained Transformer) [Software]. Available at: https://openai.com/gpt.

[4] Google LLC. (2006). Google Translate [Software]. Available at: https://translate.google.com.

[5] Doval, Y., Vilares, J., & Gomez-Rodriguez, C. (2020) *Towards Robust Word Embeddings for Noisy Texts.* arXiv.

[6] Malykh, V. (2019) *Robust-to-Noise Models in Natural Language Processing Tasks.* ACL.

[7] Helsinki-NLP. (2022). Helsinki-NLP/opus-mt-de-en: Transformer-based machine translation model [Software]. Available at: https://huggingface.co/Helsinki-NLP/opus-mt-de-en.

9 Appendix

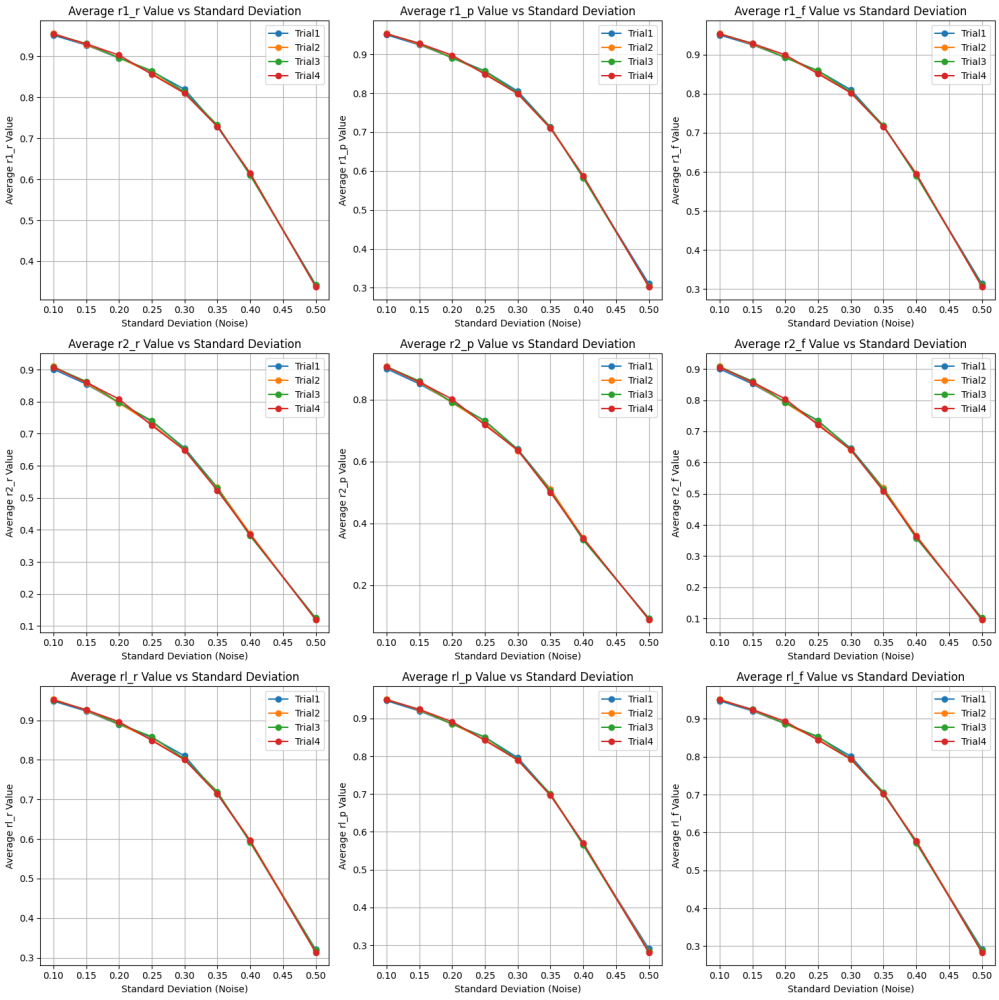

Figure 6: Mean ROUGE-1, ROUGE-2, ROUGE-L Scores Variation Across Trials

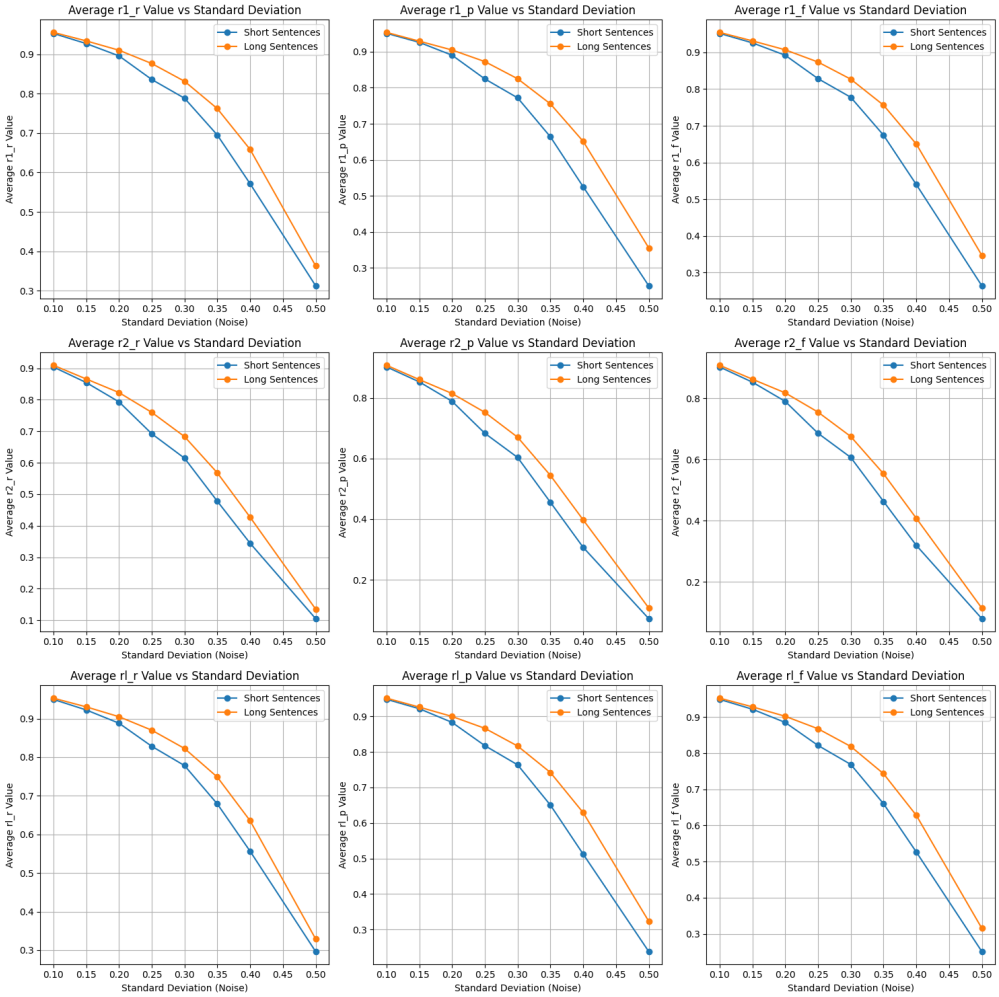

Figure 7: Mean ROUGE-1, ROUGE-2, ROUGE-L Scores Variation Across Dataset Splits

Reproducibility checklist

* Please make sure these points are addressed in your report submission
* Please copy this and replace the ☐ with a ✔ for the items that are addressed in your report/code submission
* Please complete this report, attach it to your final project report as the last page and then submit.

Model Description, algorithm, Mathematical Setting:

- ✔ Include a thorough explanation of the model/approach or the mathematical framework

Source Code Accessibility:

- ✔ Provide a link to the source code on github.
- ✔ Ensure the code is well-documented
- ✔ Ensure that the github repo has instructions for setting up the experimental environment.
- ✔ Clearly list all dependencies and external libraries used, along with their versions.

Computing Infrastructure:

- ✔ Detail the computing environment, including hardware (GPUs, CPUs) and software (operating system, machine learning frameworks) specifications used for your results.
    - (Example statement 1: the model was fine-tuned using a single T4 GPU on colab.
      Example statement 2: we ran inference of Llama 70B using 4 Nvidia A5000 GPUs)
- ✔ Mention any specific configurations or optimizations used.
    - (Example: We used a quantized version of Llama with int8.
      Example 2: We used the regular float32 representation.)

Dataset Description:

- ✔ Clearly describe the datasets used, including sources, preprocessing steps, and any modifications.
- ✔ If possible, provide links to the datasets or instructions on how to obtain them.

Hyperparameters and Tuning Process:

- ✔ Detail the hyperparameters used and the process for selecting them.
    - (Example: The model was fine-tuned using a batch size of 16, learning rate of 1e-5, and trained on 1000 steps with 100 steps of learning rate linear warmup with linear decay)

Evaluation Metrics and Statistical Methods:

✔ Clearly define the evaluation metrics and statistical methods used in assessing the model.

Experimental Results:

✔ Present a comprehensive set of results, including performance on test sets and/or any relevant validation sets.

✔ Include comparisons with baseline models and state-of-the-art, where applicable.

Limitations and future work:

✔ Include a discussion of the limitations of your approach and potential areas for future work.