Report on Disfluent to Fluent Question Conversion

Prepared by: Ritesh
Email: riteshsharma.here@gmail.com
Phone: +1(426)821-3552
Submitted to: Chata.ai
Position: NLP Engineer

Contents

Overview

The goal of this project is to transform disfluent questions—those containing hesitations, repetitions, and grammatical inconsistencies—into fluent, coherent, and grammatically correct questions using natural language processing (NLP) models. By leveraging pre-trained transformer-based models such as BART and T5, this project explores data analysis, model training, and evaluation, with a focus on improving model performance through data augmentation and fine-tuning.

The use case for this technology spans various real-world applications, including voice assistants, customer service automation, and conversational AI, where converting disfluent input into fluent text is critical for seamless communication. This report provides a detailed overview of the data analysis, model comparison, and performance evaluation

Dataset

Disfl-QA is a targeted dataset for contextual disfluencies in an information seeking setting, namely question answering over Wikipedia passages. Disfl-QA builds upon the SQuAD-v2 (Rajpurkar et al., 2018) dataset, where each question in the dev set is annotated to add a contextual disfluency using the paragraph as a source of distractors.

The final dataset consists of ~12k (disfluent question, answer) pairs. Over 90% of the disfluencies are corrections or restarts, making it a much harder test set for disfluency correction. Disfl-QA aims to fill a major gap between speech and NLP research community. We hope the dataset can serve as a benchmark dataset for testing robustness of models against disfluent inputs.

Our expriments reveal that the state-of-the-art models are brittle when subjected to disfluent inputs from Disfl-QA. Detailed experiments and analyses can be found in our paper.

Dataset Description

Disfl-QA consists of ~12k disfluent questions with the following train/dev/test splits:

File	Questions
train.json	7182
dev.json	1000
test.json	3643

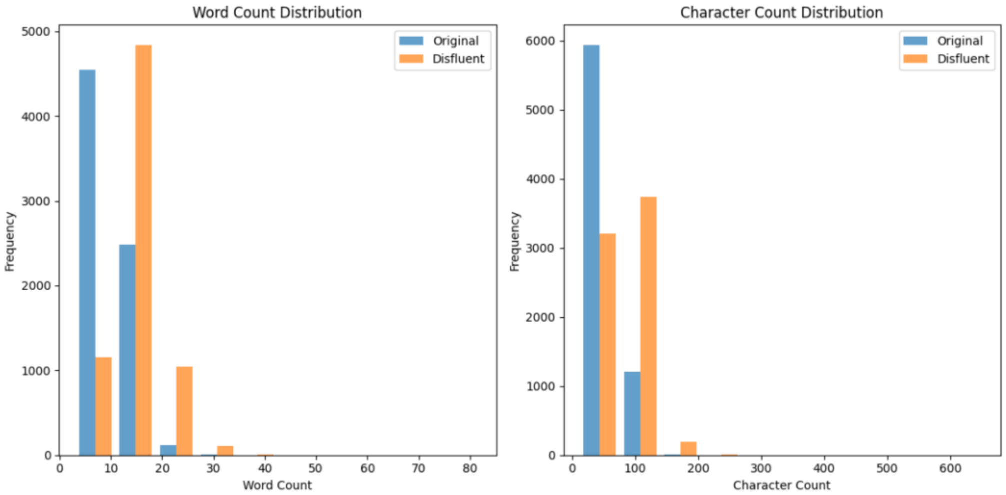
Exploratory Data Analysis (EDA)

Before commencing model training and data augmentation, a comprehensive exploratory data analysis (EDA) was conducted on the training set.

Word and Character Count Count Distribution: Original vs. Disfluent Texts

Analyzing the word count distribution revealed that disfluent texts generally had more words due to fillers and hesitations (e.g., "uh", "um"), while fluent texts were more concise. This comparison was crucial for understanding sentence complexity in disfluent vs. fluent speech.

Similarly, the disfluent texts had higher character counts, further confirming that disfluencies contribute to increased sentence length. This insight helped assess the structure and complexity of both types of text.



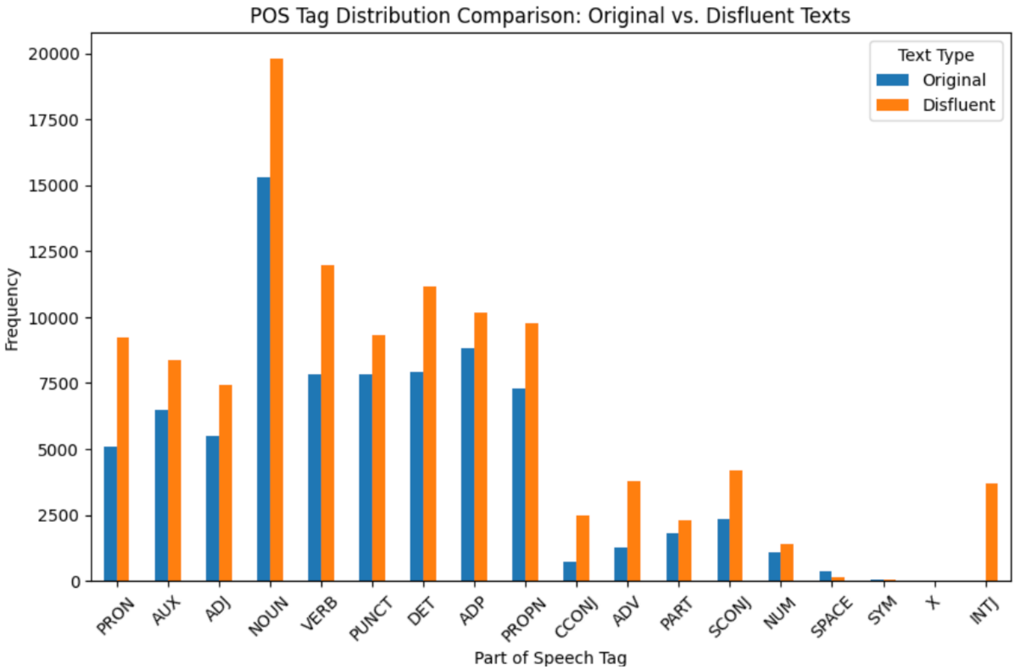
Top Unigrams, Bigrams, and Trigrams in Disfluent Texts

A frequency analysis identified the most common unigrams, bigrams, and trigrams in the disfluent texts, highlighting frequent fillers like "um" and "uh". Understanding these patterns informed our approach to data augmentation by pinpointing typical disfluencies.

n-grams	Disfluency contributors
Unigram	sorry, rather mean, uh, er, um, wait, actually, oh, instead
Bigram	er uh, sorry mean, umm rather, oh sorry, uh instead, er instead, umm mean, meant say
Trigram	sorry want know, wait want know, oh sorry mean, ah actually tell

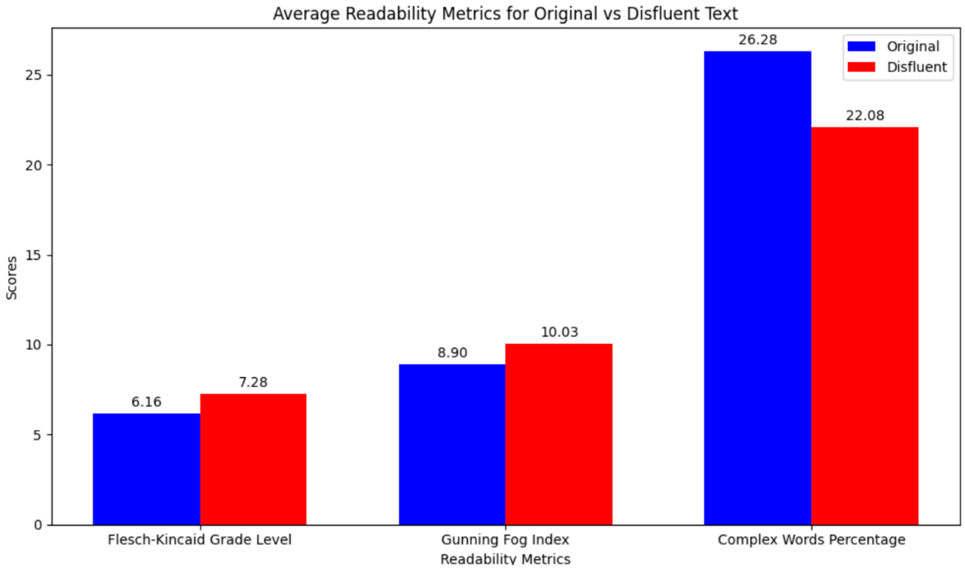
POS Tag Distribution: Original vs. Disfluent Texts

Part-of-speech (POS) tagging analysis revealed that disfluent texts had higher frequencies of pronouns, interjections, and auxiliary verbs—indicative of spontaneous, informal speech—while fluent texts were more content-heavy, utilizing more nouns and proper nouns.



Readability Metrics Comparison

Readability metrics such as Flesch-Kincaid Grade Level and Gunning Fog Index confirmed that disfluent texts were more difficult to read. This reinforced the need for transformation into fluent text to enhance clarity and comprehension.



Levenshtein Distance

Levenshtein Distance is a metric for measuring the difference between two sequences (usually strings). It calculates the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one string into another.

The **Average Levenshtein Distance** of 26.585 suggests that, on average, about 26–27 character edits are needed to transform one string into another (from disfluent to fluent).

Average Levenshtein Distance: 26.585222502099075

Data Cleaning

During EDA, several records containing fewer than 25 characters were identified. These records were deemed uninformative and potentially detrimental to model performance. As a result, they were removed to ensure that only meaningful data was used for training.

Data Augmentation

To enhance the model's ability to handle disfluent text, data augmentation was employed, doubling the size of the dataset. Two key methods were used:

- **Back Translation**: Fluent text was translated into another language (French) and then back into English, introducing slight variations in phrasing while retaining the original meaning.
- **Filler Insertion**: Contextual embeddings were used to insert fillers such as "um" and "uh" into the text, simulating conversational disfluencies.

This process enriched the diversity of disfluent texts, enabling the models to generalize better during training.

Dataset after augmentation

Dataset	**Rows**
Train	14292
Validation	1990
Test	7258

Model Training and Evaluation

Model Selection: Why T5 and BART?

For the task of converting disfluent questions to fluent questions, we selected the T5 (Text-to-Text Transfer Transformer) and BART (Bidirectional and Auto-Regressive Transformers) models. Both models are known for their strong performance in text generation and sequence-to-sequence tasks. Specifically, we chose these models for the following reasons:

- **T5**: T5 treats all tasks as text-to-text problems, making it highly flexible for tasks that involve transforming or generating text. Its architecture allows for seamless handling of both disfluency and fluency transformations, ensuring strong contextual understanding and generation accuracy.
- **BART**: BART is particularly effective for text generation and denoising tasks, which makes it well-suited for handling noisy input, such as disfluent questions. Its ability to reconstruct original text from corrupted input makes it an ideal choice for transforming disfluent text into its fluent equivalent.

Evaluation Approach

To ensure comprehensive model evaluation, we will focus on several key metrics that capture both the surface-level and semantic quality of the generated fluent questions.

Evaluation Metrics:

1. **BLEU**: Measures n-gram overlap between generated and reference questions. This is crucial for assessing fluency and the grammatical correctness of the output.
2. **ROUGE**: Evaluates recall and precision by focusing on the overlap of phrases between the generated and reference texts, ensuring structural and content alignment.
3. **BERTScore (F1)**: Leverages BERT embeddings to measure semantic similarity, which is critical for ensuring the meaning is preserved in the generated fluent questions.

Metric Explanations:

- **ROUGE-L**: Focuses on the longest common subsequence between generated and reference questions, ensuring structural integrity.
- **BLEU Score**: Captures n-gram overlap to assess the accuracy of the generated fluent questions at various levels (unigrams, bigrams, etc.).
- **BERTScore (F1)**: Evaluates the semantic similarity between generated and reference questions by comparing their embeddings, ensuring the essence of the disfluent input is preserved.

Fine Tuning models

BART Fine-Tuning (Original Dataset)

We first fine-tuned the BART model on the original dataset. The model performed well, achieving high scores across all evaluation metrics, though there was slight overfitting. This was evident from its performance on the test set, where it consistently generated fluent and grammatically correct questions.

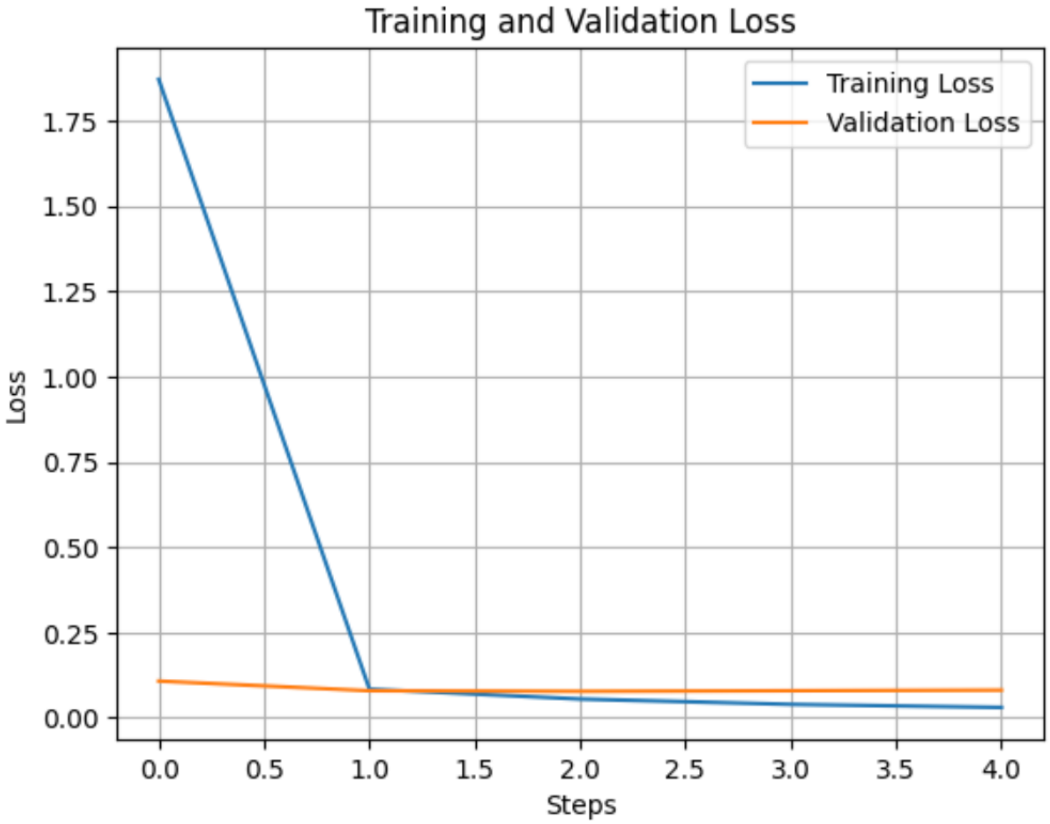


Figure 1: Training/Validation Loss Curve

Evaluation Scores:

Evaluation	Score
Bleu	0.890
Rouge	0.939
Bert_f1	0.932

BART Fine-Tuning (Augmented Dataset)

We then fine-tuned the BART model using an augmented dataset to increase data diversity and improve generalization. However, this introduced overfitting challenges, and despite several mitigation strategies, the performance dropped compared to the original dataset.

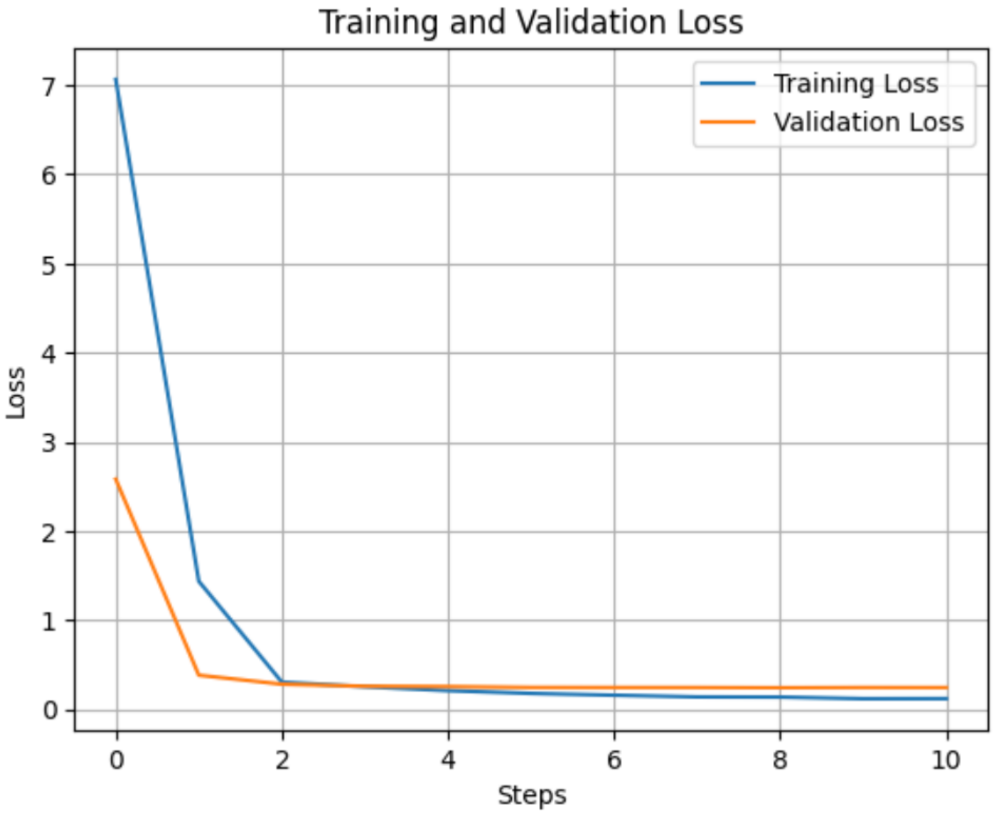


Figure 2: Training/Validation Loss Curve

Evaluation Scores:

Evaluation	Score
Bleu	0.644
Rouge	0.817
Bert_f1	0.821

T5 Fine-Tuning (Original Dataset)

Similarly, the T5 model was fine-tuned on the original dataset, and like BART, it performed very well. It showed a slight tendency to overfit, but the results were strong across all evaluation metrics, indicating that the model was effective in generating fluent and semantically accurate questions.

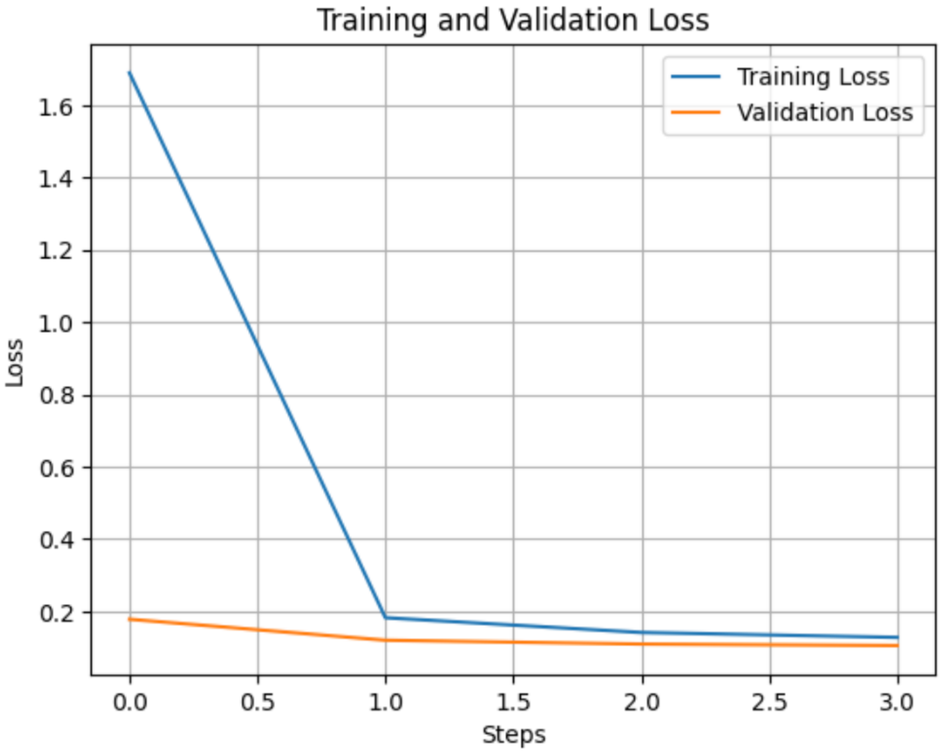


Figure 3: Training/Validation Loss Curve

Evaluation Scores:

Evaluation	Score
Bleu	0.797
Rouge	0.891
Bert_f1	0.870

T5 Fine-Tuning (Augmented Dataset)

The T5 model was then fine-tuned using the augmented dataset to test whether additional data could improve generalization. Similar to BART, T5's performance also dipped but not as much as BART when using the augmented data, showing the same overfitting challenges despite our efforts to mitigate them.

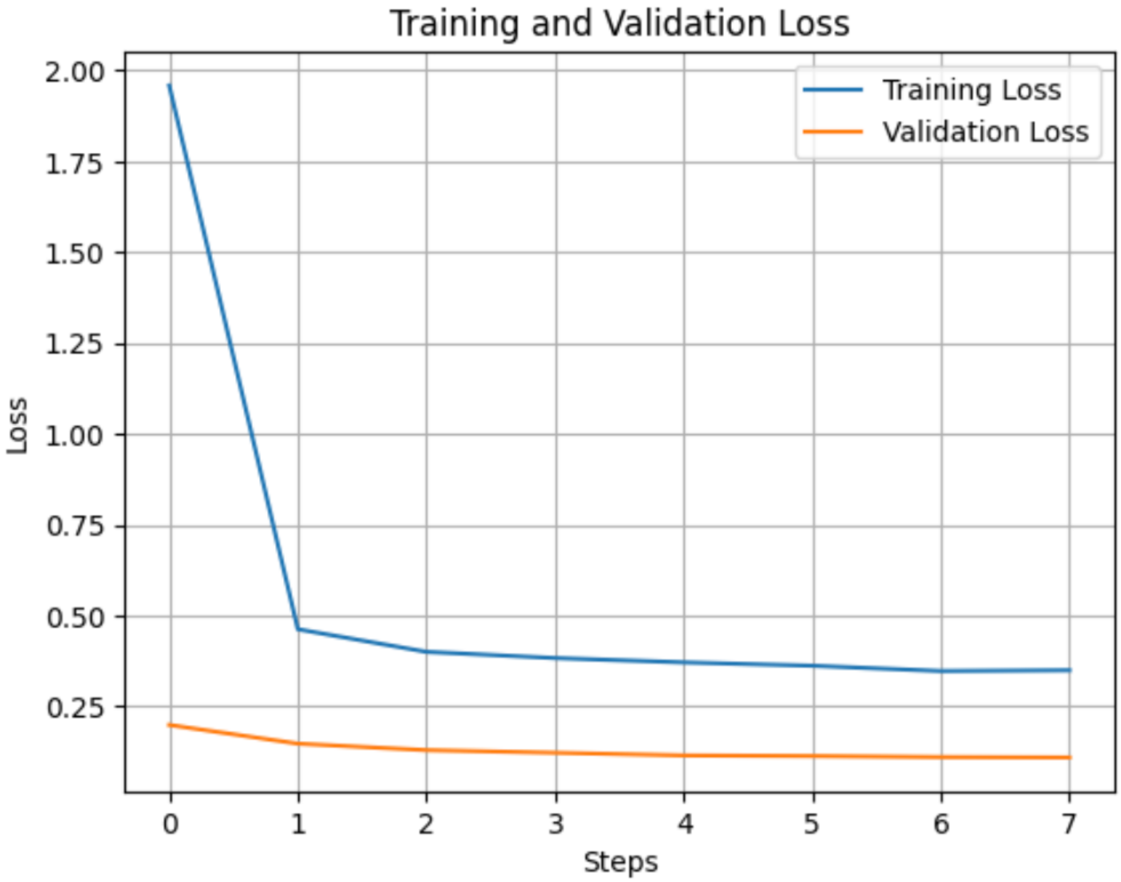


Figure 4: Training/Validation Loss Curve

Evaluation Scores:

Evaluation	Score
Bleu	0.752
Rouge	0.878
Bert_f1	0.855

Comparison

After training the models on different datasets (original and augmented), I compared their performance across various evaluation metrics. This comparison allowed me to assess how well each model performed in converting disfluent questions to fluent questions under different conditions.

Model	*Rouge Score*	*BERT Score*	*Bleu Score*
BART (fine-tuned)	0. 939	0. 932	0. 890
BART (fine-tuned w/ augmented data)	0.817	0.821	0. 644
T5(fine-tuned)	0.891	0. 870	0. 797
T5(fine-tuned w/ augmented data)	0. 878	0. 855	0.752

Based on the results outlined above, we can proceed by selecting either the BART or T5 model, both of which have been fine-tuned on the original dataset, to replicate the outcomes. For the training process, the following training arguments should be applied:

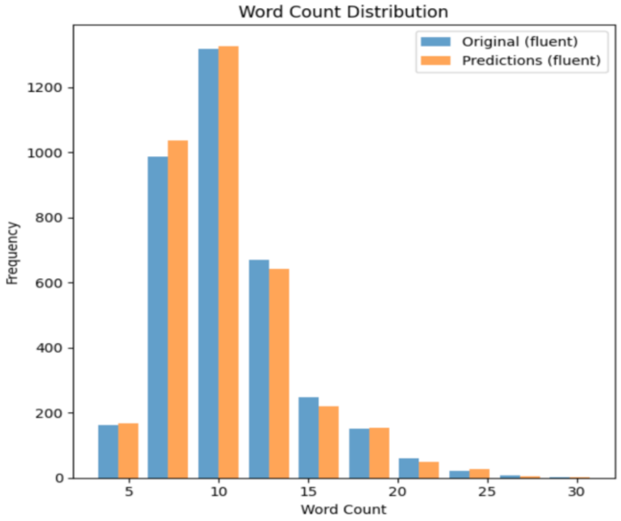
Parameter	**Value**
output_dir	./results
num_train_epochs	5
per_device_train_batch_size	64
per_device_eval_batch_size	64
eval_strategy	steps
eval_steps	100
logging_dir	./logs
logging_steps	100
save_steps	200
save_total_limit	2
report_to	none
seed	42

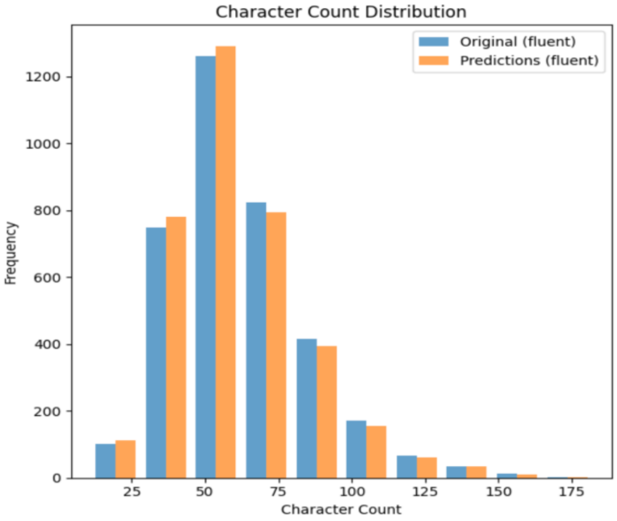
Prediction Comparison and Visualization

Below are the visualization to confirm how perfectly the prediction aligns with the original text using BART fine tuned on the original data

Word and Character Count Distribution: Original vs. Predicted Texts

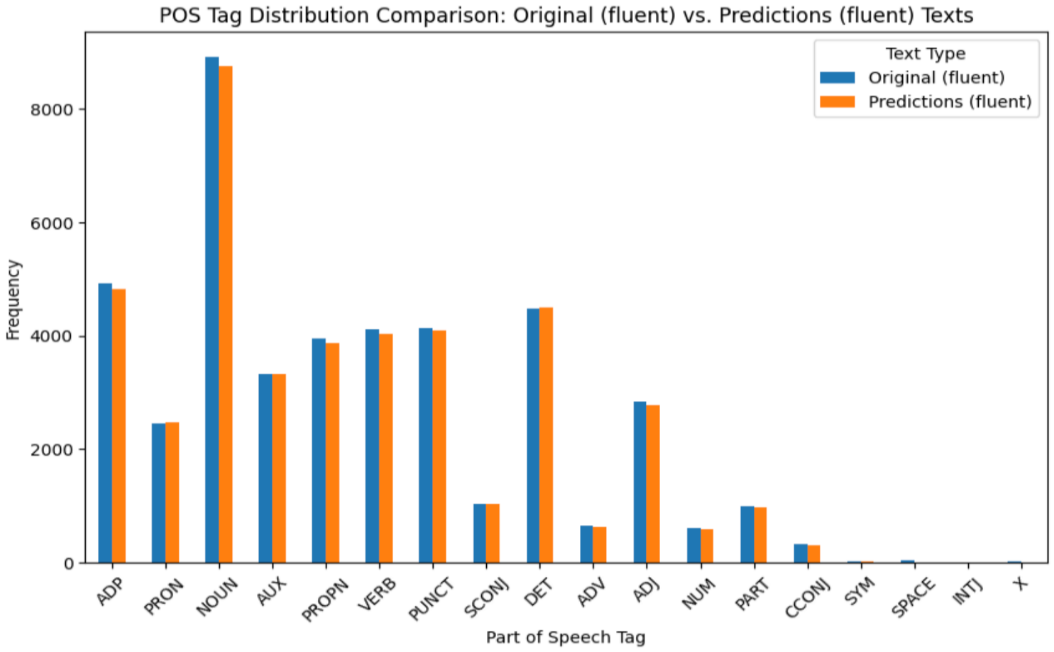
The BART-generated texts closely matched the original fluent texts, although they were slightly longer, indicating some over-generation. Despite the increase in length, the character count analysis revealed that the predicted texts remained highly aligned with the structure and content of the original fluent texts, demonstrating BART's strong ability to generate accurate and fluent outputs.


Word Count Distribution


Character Count Distribution

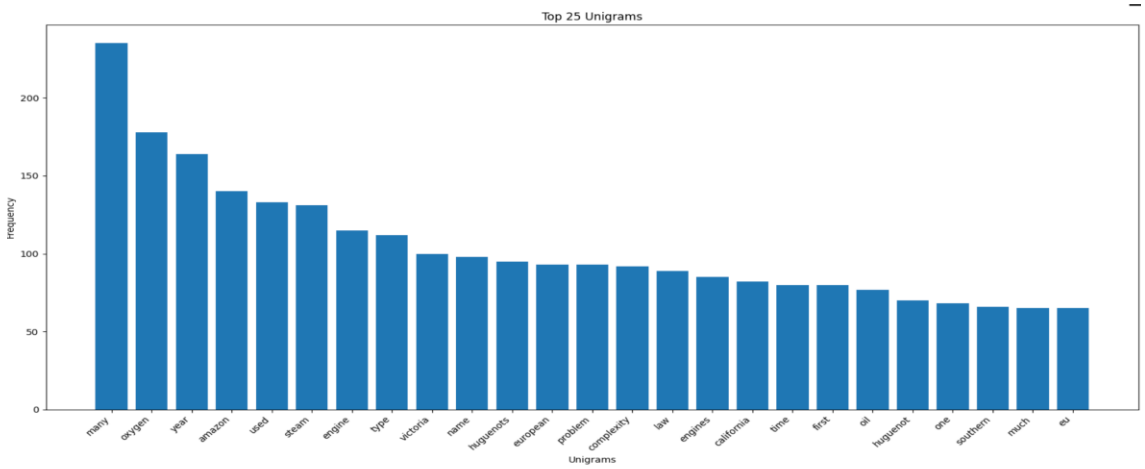
POS Tag Distribution: Original vs. Predicted Texts

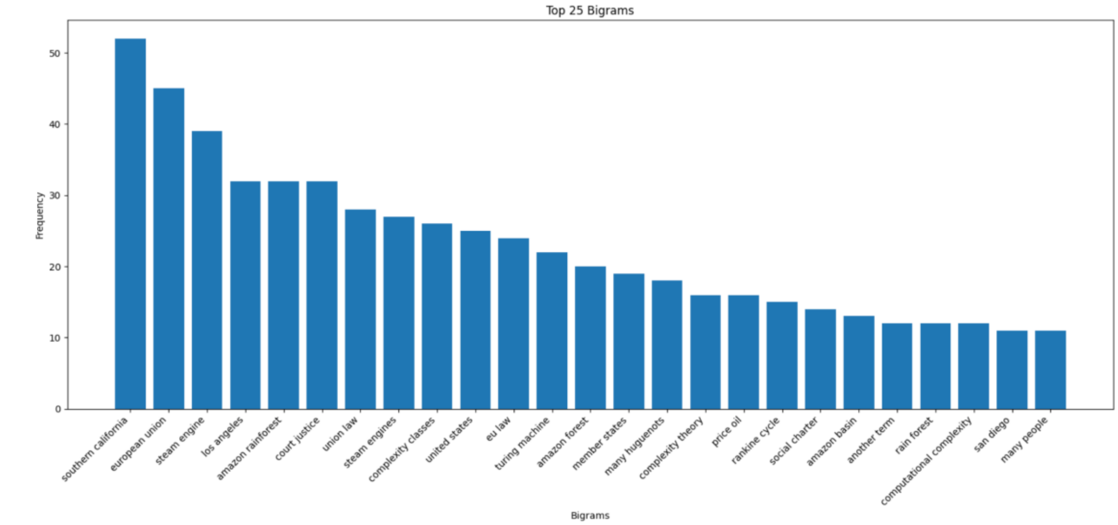
The predicted texts exhibited fewer pronouns and interjections than the disfluent texts, highlighting the model's ability to reduce disfluencies and streamline the text.

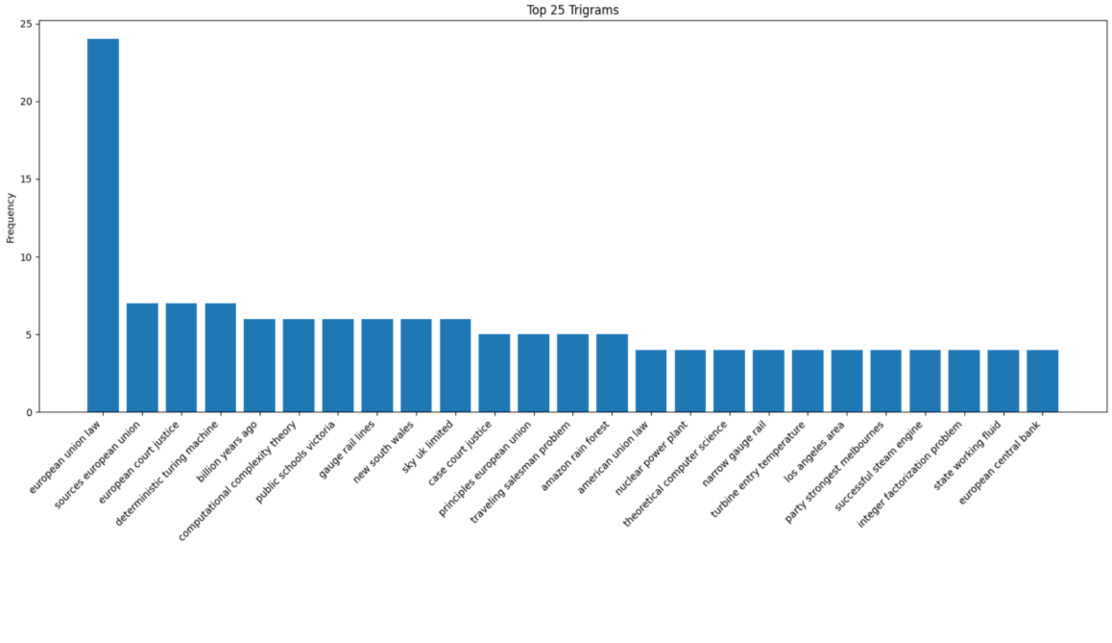

POS Tag Distribution Comparison: Original (fluent) vs. Predictions (fluent) Texts

Unigrams, Bigrams, and Trigrams in Predicted Texts

The model effectively captured key content words while successfully minimizing filler words typically associated with disfluent speech.

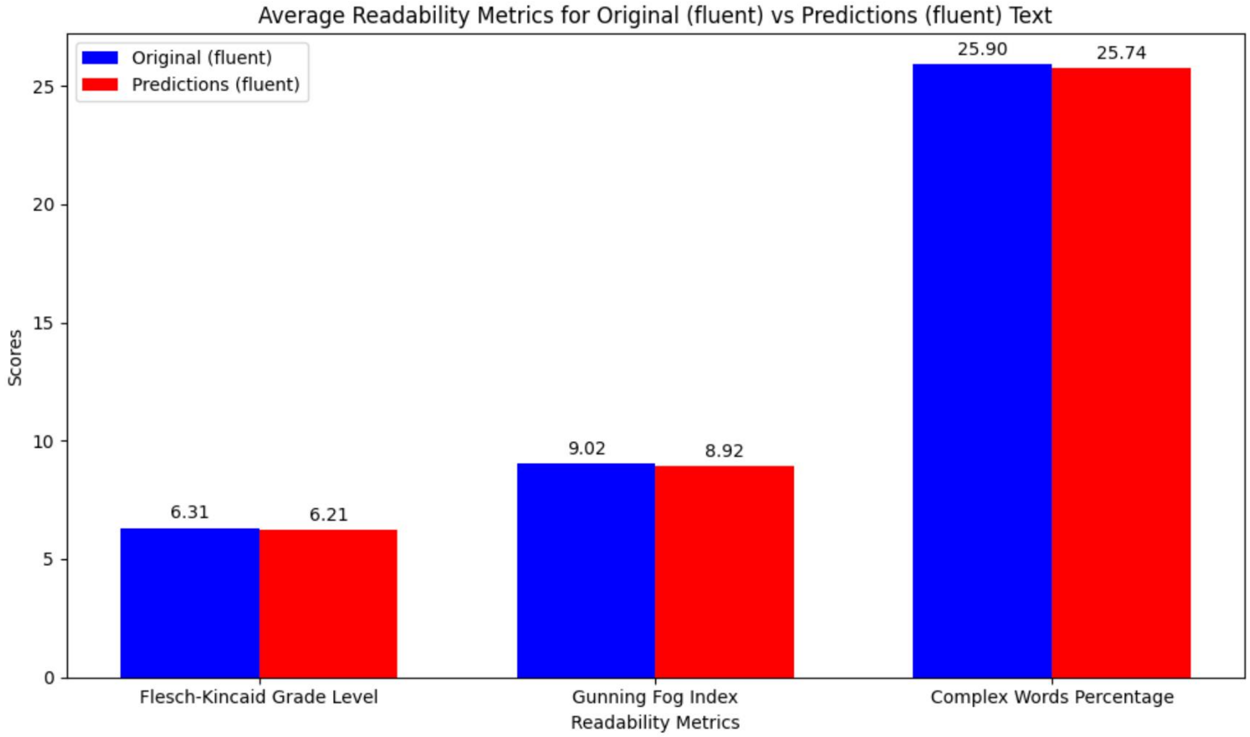






Readability Metrics Comparison

The readability metrics showed that the predicted texts closely matched the original fluent texts, demonstrating an improvement in sentence structure and overall fluency.



Levenshtein Distance

The **Average Levenshtein Distance** of 4.72 suggests that, on average, about 5 character edits are needed to transform one string into another (from disfluent to fluent).

Average Levenshtein Distance: 4.726

Conclusion

In this NLP project focused on converting disfluent text to fluent text, we evaluated two pre-trained models: BART and T5. Both models demonstrated strong performance when fine-tuned on our original dataset, showing that they are effective for disfluency correction tasks. However, there are key distinctions between the two models that should inform the choice depending on project goals.

If achieving the highest accuracy is the primary objective, BART is the preferred model, as it consistently outperforms T5 in terms of fluency and grammatical correction accuracy. BART's architecture is designed to handle sequence generation tasks more robustly, making it highly suitable for tasks that prioritize the quality of the output.

On the other hand, if fast inference time and computational efficiency are more critical, T5, especially the T5-small variant, becomes an attractive choice. Although there is a slight decrease in accuracy compared to BART, the difference is minimal and may be acceptable in applications where speed and lower resource consumption are more important. Given T5-small's significantly smaller size compared to BART, it offers a good balance between accuracy and efficiency for real-time applications or resource-constrained environments.

Thus, both models have distinct advantages: BART for scenarios where accuracy is paramount, and T5 for environments that prioritize faster inference with slightly reduced accuracy. The decision should be guided by the specific needs of the project, whether it's accuracy or efficiency.