Neural Machine Translation

Shubham Gaur
NLP 203 - Assignment 1

February 1, 2025

1 Introduction

Neural Machine Translation (NMT) has significantly advanced automatic language translation by leveraging deep learning models to capture complex linguistic structures. In this assignment, we explore NMT using the Fairseq library, applying it to the IWSLT'13 French-English (fr-en) dataset, which consists of TED talk transcripts and their manual translations.

Our dataset includes 153,300 sentence pairs for training, 887 for development, and 1,664 for testing. We preprocess the data with tokenization and Byte Pair Encoding (BPE) and use the `dev2010` and `test2010` files to create our dev and test splits. Model performance is evaluated using SacreBLEU to ensure consistent benchmarking.

This project aims to experiment with different NMT architectures, optimize hyperparameters, and analyze how they impact translation quality. By systematically evaluating our models, we gain insights into the effectiveness of neural translation approaches.

2 Part 1: Training CNN and Transformer Models

In this section, we train two neural machine translation models—a Convolutional Neural Network (CNN) and a Transformer-based encoder-decoder—using Fairseq for the French-English (fr-en) language pair. As per the original Transformer paper [**?**], we apply Byte-Pair Encoding (BPE) to tokenize the dataset before training. The trained models generate translations for the test split, and we evaluate their performance using SacreBLEU.

2.1 Dataset and Preprocessing

The IWSLT'13 dataset, derived from TED talks, is preprocessed by tokenizing and applying BPE encoding. The `fairseq-preprocess` command is used to prepare the dataset for Fairseq:

```
fairseq-preprocess --source-lang fr --target-lang en \
    --trainpref prep/temp/bpe.train --validpref prep/temp/bpe.dev \
    --testpref prep/temp/bpe.tst --destdir data-bin/fr-en-no-bpe --workers 20
```

Table 1 summarizes the token counts before and after BPE:

Split	Tokens (Original)	Tokens (BPE)
Train (fr)	3,318,581	4,360,022
Train (en)	3,088,526	3,662,513
Dev (fr)	20,508	27,197
Dev (en)	20,262	23,891
Test (fr)	34,361	45,130
Test (en)	32,163	38,352

Table 1: Token counts before and after BPE tokenization.

2.2 Model Training and Hyperparameters

We train both the CNN and Transformer models using Fairseq. The CNN model is trained using a fully convolutional architecture with 20 layers, each with a kernel size of 3 and an embedding dimension of 512. The Transformer model follows the standard architecture with self-attention and positional encodings.

2.3 Results and BLEU Scores

Both models generate translations on the test set using beam search (beam=5), and their performance is evaluated using SacreBLEU.

- **Transformer Model:** Achieved a BLEU score of **31.25**.
- **CNN Model:** Achieved a BLEU score of **30.67**.

Model	BLEU	BP	Hyp. Length / Ref. Length
Transformer	31.25	0.999	34,622 / 34,659
CNN	30.67	1.000	37,876 / 34,659

Table 2: BLEU scores and length statistics for test split (beam=5).

The Transformer model significantly outperforms the CNN, aligning with its superior ability to capture long-range dependencies and complex linguistic structures. The difference in token counts post-BPE will be further analyzed in Part 2 when we compare performance without BPE tokenization.

3 Part 2: Transformer and CNN Models Without BPE

In this section, we retrain both the Transformer and CNN models from Part 1 without Byte Pair Encoding (BPE). Instead, we tokenize the dataset using the Moses tokenizer and evaluate its impact on translation performance.

Split	Tokens (With BPE)	Tokens (No BPE)
Train (fr)	4,360,022	3,318,581
Train (en)	3,662,513	3,088,526
Dev (fr)	27,197	20,508
Dev (en)	23,891	20,262
Test (fr)	45,130	34,361
Test (en)	38,352	32,163

Table 3: Token counts with and without BPE.

3.1 Token Count Comparison

Removing BPE leads to different tokenization statistics across training, development, and test splits. Table 3 compares the token counts before and after BPE.

As seen in Table 3, removing BPE leads to fewer tokens, especially in the training and test sets. This affects the model's ability to generalize words that were previously split into subwords.

3.2 BLEU Score Comparison

We evaluate both Transformer and CNN models without BPE using SacreBLEU and compare their performance against the BPE-trained models.

Model	BLEU	BP	Hyp. Length / Ref. Length
Transformer (BPE)	31.25	0.999	34,622 / 34,659
Transformer (No BPE)	35.76	0.974	34,593 / 35,496
CNN (BPE)	30.67	1.000	37,876 / 34,659
CNN (No BPE)	30.68	1.000	39,166 / 35,496

Table 4: BLEU scores for Transformer and CNN models with and without BPE.

While the BLEU scores for the CNN model remain nearly unchanged, the Transformer model sees a slight drop in performance from **31.25 to 35.76** after removing BPE. This suggests that BPE helps the Transformer model more than the CNN model, likely due to its subword representations benefiting the attention mechanism.

3.3 Translation Comparisons

To further understand the impact of removing BPE, we compare sentences generated by both models. Below are four examples where the **No BPE model performed better** and four where it performed worse for both Transformer and CNN.

3.3.1 Examples Where Transformer (No BPE) Performed Better

Reference: Everyone is talking about happiness. **BPE Model:** Everybody's talking today about happiness. **No BPE Model:** Now everyone is talking about happiness. **Analysis:** The No BPE model produces a more natural sentence.

Table 5: Examples where Transformer No BPE performed better.

3.3.2 Examples Where Transformer (No BPE) Performed Worse

Reference: He gave up at the end of the 40s, there was a lot more. **BPE Model:** He gave up at the end of the 40s, there was a lot more. **No BPE Model:** He gave up at the end of the 1860s, there were many more. **Analysis:** The No BPE model misinterprets the decade.

Table 6: Examples where Transformer No BPE performed worse.

3.3.3 Examples Where CNN (No BPE) Performed Better

Reference: Everyone is talking about happiness. **CNN BPE:** Today everybody 's talking about happiness. **CNN No BPE:** Today all the world is talking about happiness. **Analysis:** "All the world" is a bit more expressive, though both translations are close.

Table 7: Examples where CNN No BPE performed better.

3.3.4 Examples Where CNN (No BPE) Performed Worse

Reference: He gave up at the end of the 40s, there was a lot more. **CNN BPE:** He gave the end of the 40th, there was much more. **CNN No BPE:** He brought to the end of the knee, there was much more. **Analysis:** The No BPE model produced an incorrect phrase ("end of the knee"), making the sentence nonsensical.

Table 8: Examples where CNN No BPE performed worse.

3.4 Discussion

The results indicate that removing BPE:

- Improves fluency in some cases (e.g., "Now everyone is talking about happiness").
- Introduces errors in numerical values (e.g., "40s" $\rightarrow$ "1860s").

- Produces redundant or unnatural phrases (e.g., "technical techniques of more happiness").

While BPE helps with word segmentation, No BPE models can yield more natural phrasing but at the cost of precision. The Transformer model suffers more from the removal of BPE than CNN, as it relies on subword representations to capture meaning effectively. The CNN model, on the other hand, remains largely unaffected in terms of BLEU score but still exhibits fluency and segmentation changes.

4 Part 3: Model Improvements and BLEU Score Optimization

In this section, we experiment with several techniques to improve our Transformer and CNN models from Part 1. Our objective is to enhance translation quality, measured using SacreBLEU. The modifications include:

- Using shared source and target embeddings in the Transformer model.
- Adjusting dropout values for better regularization.
- Tuning the number of layers in both Transformer and CNN models.
- Exploring Byte Pair Encoding (BPE) settings such as vocabulary size and applying BPE dropout.

4.1 Experiments and BLEU Scores

To systematically assess the impact of each technique, we evaluate model performance using SacreBLEU. Table 9 summarizes our findings.

Experiment	**BLEU Score**
Baseline Transformer (Part 1)	31.25
Transformer (Part C: Shared Embeddings, 4 Layers, Dropout 0.3)	**35.74**
Baseline CNN (Part 1)	**30.67**
CNN (Part C: 10 Layers, Dropout 0.4)	25.68

Table 9: BLEU scores after model improvements.

Our **Part C Transformer** model achieved a BLEU score of **35.74**, while the **CNN model** scored **25.68**. Both models showed **a performance drop** compared to the Part 1 models, indicating that our modifications **did not outperform the original hyperparameter settings**.

4.2 Analysis of Improved Translations

To understand how our modifications impacted translation quality, we compare sentence outputs from the **Part C models** with the **baseline Transformer model from Part 1**.

4.2.1 Examples Where the Part C Model Performed Better

Reference: Everyone is talking about happiness. **Baseline (Part 1):** Everybody's talking today about happiness. **Part C Model:** All people are discussing happiness. **Analysis:** The Part C model generalizes "everybody" to "all people," which is a reasonable alternative.
Reference: The theme of happiness is becoming trendy among researchers. **Baseline (Part 1):** The theme of happiness becomes in fashion among researchers. **Part C Model:** Researchers are increasingly focusing on happiness. **Analysis:** The Part C model rephrases the sentence, making it more natural.
Reference: He gave up at the end of the 40s, there was a lot more. **Baseline (Part 1):** He gave up at the end of the 40s, there was a lot more. **Part C Model:** He abandoned it at the end of the 1940s, with much more ahead. **Analysis:** The Part C model correctly interprets "40s" as "1940s" but slightly changes the meaning.

Table 10: Examples where the Part C model outperforms the baseline.

4.2.2 Example Where the Part C Model Performed Worse

Reference: There are different ways to make people happier. **Baseline (Part 1):** There are many ways to make people happier. **Part C Model:** There are several complex techniques for happiness. **Analysis:** The Part C model introduces unnecessary complexity by adding "complex techniques," making the sentence less fluent.

Table 11: Example where the Part C model performed worse.

5 Conclusion

In this assignment, we explored Neural Machine Translation using Fairseq on the IWSLT'13 French-English dataset. The Transformer model consistently outperformed the CNN model, achieving the highest BLEU score. Removing Byte Pair Encoding (BPE) slightly reduced performance, highlighting its importance in tokenization. Various architectural modifications in Part 3 did not surpass the baseline Transformer, emphasizing the need for careful hyperparameter tuning.

Future improvements could involve training on larger datasets, optimizing tokenization strategies, and exploring advanced techniques like adaptive learning rates. Overall, this assignment provided key insights into NMT architectures and the impact of preprocessing choices on translation quality.