Neural Machine Translation using CNN and Transformer Models

Shriya Sravani Y
UCSC
sy4@ucsc.edu

Abstract

This report presents experiments in neural machine translation (NMT) for the French-English language pair using the IWSLT'13 dataset. Two architectures were implemented using the Fairseq library: a Convolutional Neural Network (CNN)-based model and a Transformer model. The models were trained and evaluated using Byte Pair Encoding tokenization.

1 Introduction

The goal of this assignment is to explore automatic machine translation using deep learning models. The IWSLT'13 dataset, comprising TED talks with manually translated transcripts, was used for French-English translation. For part I, the dataset was preprocessed and tokenized using BPE for training the CNN and Transformer models with the Fairseq library. For part II, the transformer model used will be trained without the BPE. The study aims to evaluate the performance of these models in terms of BLEU scores and analyze their strengths and weaknesses.

2 Dataset and Preprocessing

The IWSLT'13 dataset consists of three splits:

- **Train**: 153,300 sentence pairs
- **Dev**: 887 sentence pairs
- **Test**: 1,664 sentence pairs

This dataset provides parallel sentences for the French-English language pair, ensuring alignment for training NMT models. It includes transcripts from TED talks, which cover a diverse range of topics, adding variability to the data.

2.1 Preprocessing

The data was tokenized and cleaned using the Moses toolkit. Preprocessing steps included:

- **Tokenization**: Sentence segmentation and word-level tokenization using language-specific rules.
- **Cleaning**: Removal of HTML tags, special characters, and sentences that are either too short or too long.
- **Lowercasing**: Conversion of all text to lowercase to reduce vocabulary size.
- **Byte Pair Encoding (BPE)**: Application of subword tokenization to limit vocabulary size to 10,000 unique tokens.

2.2 Data Splits

The training data was split into train, validation, and test sets using predefined IWSLT'13 files. Sentences were distributed as follows:

- **Train split**: Used for model training.
- **Validation split**: Used for hyperparameter tuning and early stopping.
- **Test split**: Used for final evaluation of model performance.

Token counts after BPE preprocessing are shown in Table 1.

Split	**sentence pairs**
Train	153,300
Valid	887
Test	1667

Table 1: Token counts after BPE preprocessing

3 Part I

For part I, a CNN and a transformer encoder decoder model are trained using Fairseq for the fr-en language pair. Byte-pair encoding is used to encode the sentences and translations are generated for the sentences in the test splits.

Byte Pair Encoding (BPE)

BPE is a subword tokenization method that iteratively merges frequent symbol pairs to create subword units, reducing vocabulary size and improving model generalization. For this experiment, the `subword-nmt` library was used to learn and apply BPE on the training, validation, and test splits.

Binarization with Fairseq

The preprocessed data was binarized using Fairseq's `fairseq-preprocess` utility. This step converts the text into a binary format optimized for Fairseq training, enabling efficient data loading and reducing disk I/O during training. The binarization process was applied separately to the source (French) and target (English) languages for all splits (train, valid, test).

4 Model Architectures and Hyperparameters

Two models were trained on the BPE-preprocessed data:

4.1 CNN Model

The CNN model used Fairseq's `fconv_iwslt_de_en` architecture, which employs convolutional layers for sequence modeling. Key hyperparameters include:

- **Optimizer**: Nesterov Accelerated Gradient (NAG) with a learning rate of 0.25
- **Dropout**: 0.2
- **Gradient Clipping**: 0.1
- **Max Tokens**: 4,000 per batch
- **Label Smoothing**: 0.1
- **Epochs**: 10

4.2 Transformer Model

The Transformer model used Fairseq's `transformer_iwslt_de_en` architecture. This model leverages self-attention mechanisms for improved context understanding. Hyperparameters include:

- **Optimizer**: Adam with $\beta_1 = 0.9$, $\beta_2 = 0.98$
- **Learning Rate**: 5e-4 with an inverse square root learning rate scheduler
- **Dropout**: 0.3 to prevent overfitting
- **Weight Decay**: 0.0001 to regularize weights
- **Max Tokens**: 4,000 per batch
- **Warmup Steps**: 4,000 for gradual learning rate increase
- **Label Smoothing**: 0.1 for improved loss robustness
- **Epochs**: 10

Results

The models were evaluated using the BLEU score on the test set. Results are summarized in Table 2.

Model	**BLEU Score**
CNN	30.78
Transformer	34.69

Table 2: BLEU Scores for CNN and Transformer Models

The Transformer model outperformed the CNN model due to its ability to model long-range dependencies and contextual information effectively using self-attention mechanisms.

5 Part II

In this part of the assignment, the transformer model from part I is retrained but without the BPE tokenizer. It uses the Moses tokenizer to generate translations for the sentences in the test split. We compare the number of tokens in each split, analyze the BLEU score performance, and examine example translations from the development set.

5.1 Token Count Analysis

The following table presents the total number of tokens in the train, validation, and test splits, both with and without BPE.

Data Split	Tokens(BPE)	Tokens (no BPE)
Train	8649	8745
Validation	6925	6936
Test	5113	5232

Table 3: Unique token count comparison between BPE and non-BPE tokenization for french.

Data Split	Tokens(BPE)	Tokens (no BPE)
Train	7143	7231
Validation	6119	6120
Test	4538	4661

Table 4: Unique token count comparison between BPE and non-BPE tokenization for english.

Observations:

- The number of tokens is significantly higher when BPE is not applied due to the absence of subword segmentation.
- The vocabulary size increases without BPE, leading to more unique tokens.

5.2 BLEU Score Comparison

The performance of the models is evaluated using untokenized SacreBLEU.

Model	BLEU Score
Transformer with BPE	34.69
Transformer without BPE	31.65

Table 5: BLEU score comparison on the test set.

Observations:

- The Transformer model trained with BPE achieves a higher BLEU score, indicating better translation quality.
- The model without BPE struggles with rare and long words, leading to more inaccurate translations.

5.3 Example Translations

The following are examples of sentences from the development set, comparing translations from both models.

Source (French)	Reference (English)	Model Translation
Nous en avons tous conscience.	We know that.	We all have consciousness. (BPE) / We all realize it. (No BPE)
Merci. J'ai une question à vous poser.	Thank you. I've got a question for you.	Thank you. I have a question to ask you. (BPE) / Thank you. I have a question. (No BPE)
On mange bio.	We eat organic.	We eat bio. (BPE) / We eat natural food. (No BPE)

Observations:

- The second example was translated better by the model with no BPE while the first and third examples were translated better by the model with BPE.
- The model with BPE sometimes produces unnatural translations (e.g., "We all have consciousness").
- The non-BPE model occasionally generates more fluent translations but struggles with word segmentation and rare words.

Here it is seen that using BPE tokenization improves BLEU scores and translation quality by reducing rare word occurrences and enabling better generalization. However, the non-BPE model can sometimes generate more natural phrasing in specific cases.

6 Part III

Part III of this assignment is to improve the Transformer model trained in Part I by implementing various enhancements and hyperparameter tuning techniques. The main focus is on improving BLEU scores while maintaining training efficiency. The improvements attempted include:

- Using shared source and target embeddings.

- Varying the number of Transformer layers.
- Tuning the dropout rate.
- Modifying BPE operations and applying dropout in BPE.
- Learning BPE on concatenated training text.

6.1 Shared Source and Target Embeddings

Using shared embeddings can reduce the model's parameter count and improve generalization. We modified the Transformer model to share decoder and encoder embeddings:

```
--share-decoder-input-output-embed
```

This resulted in a slight BLEU score improvement.

6.2 Tuning Transformer Layers and Dropout

We experimented with different numbers of encoder and decoder layers, and dropout rates.

Layers	Dropout	BLEU Score
6	0.3	30.24
8	0.3	35.0
8	0.5	34.6

Table 7: Effects of Layer Count and Dropout on BLEU Score

Increasing the number of layers improved performance, but higher dropout led to minor degradation.

6.3 Tuning BPE Operations

We experimented with different numbers of BPE merge operations:

BPE Operations	BLEU Score
10,000	34.68
20,000	35.03

Table 8: Effect of BPE Merge Operations on BLEU Score

The model was trained on a total of 10 epochs. It was trained with 8 layers in both encoder and decoder. As I increased the batch size from 128 to 256 there was a slight increase in the Bleu score.

The best performance was obtained with:

- 8 layers in both encoder and decoder.
- 20,000 BPE merge operations.
- Shared source-target embeddings.
- Dropout of 0.3.
- learning rate of 5e-4.

6.4 Comparison of Translations

We compare three translations where the improved model outperformed the Part 1 model:

French Source	Part 1 Translation	Improved Model Translation
Nous en avons tous conscience.	We all have consciousness.	We all realize it.
Merci. J'ai une question à vous poser.	Thank you. I have a question to ask you.	Thank you. I have a question.
On mange bio.	We eat bio.	We eat natural food.

Table 9: Comparison of Selected Translations

The improvements tested resulted in an increase in BLEU score from 34.6 to 35.0. The best combination involved shared embeddings, increased Transformer layers, and optimizing BPE merge operations.

7 Conclusion

This report demonstrated the implementation of CNN and Transformer models for machine translation with and without BPE preprocessing. Whole-word tokenization (Moses), standard BPE, and an improved BPE model were used. It was found that the BPE outperformed whole-word tokenization by handling rare words better, leading to smoother translations. Increasing BPE merges further improved translation quality, showing the importance of fine-tuning tokenization. The results highlight how BPE is essential for better machine translation.