A comparative study of the effects of BPE on SMT and NMT performance as applied on Inuktitut

Viktorija Buzaitė, Rafał Černiavski, Laura Janulevičiūtė, Eva Elžbieta Sventickaitė

October 29, 2021

1 Introduction

1.1 Byte Pair Encoding

Byte Pair Encoding (BPE) is a data compression method introduced by Gage (1994). It is commonly used in modern NLP for sub-word tokenization, wherein words with lower frequency are broken into sub-words. This tends to significantly reduce the number of out-of-vocabulary words, which is a considerable challenge in Machine Translation (MT) (Provilkov et al., 2020). Furthermore, BPE is fairly simple to implement, as it has only one parameter and can be done using just the source and target corpora. The single parameter used in BPE is the number of merge operations. It sets a limit to the amount of merges performed on the data that is initially broken down into characters. As a result, the frequent combination of characters are merged into morphemes, which may then be joined with other morphemes and roots, etc. A common practice is to use a number of merges that equals the size of the vocabulary. Nevertheless, the choice of an optimal number of merges still remains a somewhat open question. Seeking to address this, Ding et al. (2019) compiled the 44 papers accepted for Conference of Machine Translation in 2017 and 2018. They evinced that the most common choice of merges was 30k-40k; however, they concluded that the optimal number of merges for most settings was yet to be studied. According to Ding et al. (2019), the choice of an optimal number of merges can improve performance of MT models by up to 4 BLUE score. They carried out a systematic study of a range of merges (from 0 - character level - to 32k) to see how it would affect NMT models for English, Czech, German, French, and Arabic. In their study, the authors trained LSTM and Transformer based models. It became evident that in the case of Transformers, 0-4k merges was the most optimal choice across all language pairs. LSTM-based models, on the other hand, did not appear to demonstrate preference for a specific range of merge operations across languages, thus indicating that it highly depends on the language pair. In this experiment, we seek to extend the research of Ding et al. (2019) by investigating the effects of BPE on two additional models, namely SMT and CNN. Furthermore, we assess he optimal BPE merge range for both the LSTM architecture and the Transformer on Inuktitut to English MT models. As such, we have chosen to evaluate the effects of 1,000, 2,000, 5,000, 10,000, 20,000, and 30,000 merge operations on the chosen models.

Our early hypotheses were the following:

- Phrase based SMT model will perform better with more than 5,000 merge operations used by Joanis et al. (2020) on Inuktitut to English translation direction.
- LSTM architecture will exhibit no significant differences for the number of BPE merges, as found by Ding et al. (2019).
- CNN model will achieve best results with 30.000 BPE merges as it would retain the largest amount of characteristics, which would facilitate feature learning via feature maps (Gehring et al., 2017).
- Transformer architecture will perform best with 1,000 and 2,000 BPE merges, as was found by Ding et al. (2019).

1.2 The Inuktitut language

The chosen language pair for the experiments regarding BPE effects is Inuktitut and English. Inuktitut is a morphologically complex language spoken in North of Canada with as many as 40k native speakers. The language uses its own alphabet, titirausiq nutaaq, which is typographically different from English as well as any other languages that use the Latin script. It is considered a low-resource language which, in combination with preprocessing hindrances, makes the automatisation of the translation tasks of this language rather challenging. Nevertheless, due to its distinction from other, more popular languages, Inuktitut offers valuable insight into the shortcomings of modern models when used on low-resource languages. As such, we believe that the models trained on Inuktitut without BPE would perform lower compared to those trained on BPE data, as BPE segments the sequences it would facilitate the learning of morphological features specific to Inuktitut.

2 Data preparation

In all of our experiments, we used the Nunavut Hansard Inuktitut–English Parallel Corpus 3.0 (Joanis et al., 2020). The corpus consists of sentence-aligned proceedings of the Legislative Assembly of Nunavut. For our project, we used the normalized to the Latin alphabet version of the Inuktitut corpus provided by the authors. After manual analysis of the data sets, we observed that the original corpora contained multiple empty lines, leading to flawed alignment. Hence, we removed the empty lines along with their corresponding lines in the parallel data set. We then used the Moses toolkit to tokenize, lowercase, and remove the sentences with more than 35 words. The sizes of the data sets after additional cleanup can be seen below in Table 1.

	Inuktitut	**English**
Train	337,729	337,729
Development	84,432	84,432
Test	99,936	99,936

Table 1: The Nunavut Hansard Inuktitut–English Parallel Corpus 3.0 after manual cleanup

Afterwards, we used SubwordNMT to produce byte pair encodings with 1,000, 2,000, 5,000, 10,000, 20,000, and 30,000 merge operations. We used only joint BPE, that is, trained on both languages simultaneously, as previous research has found no difference in the performance between joint and separate BPE (Ding et al., 2019).

3 Experimental set-up

We trained 4 model architectures: SMT, LSTM, CNN and Transformer with joint BPE data with 1,000, 2,000, 5,000, 10,000, 20,000, and 30,000 merge operations. In addition, we trained the models with word-level tokenization to compare the performance against BPE. We trained the models in one translate direction - from Inuktitut to English. The NMT models were trained on 50 epochs each, unless stated differently in the separate sections below. The number for separate models could have been raised in the process of fine-tuning. Each model which showed the best performance with the particular number of BPE merges was fine-tuned in order to improve the performance. BLEU score was used for evaluation (Papineni et al., 2002).

4 SMT

4.1 Theory and Methods

Though mostly used in NMT models, it has been proven that BPE can boost the performance of SMT models (Oudah et al., 2019). BPE is especially useful for language pairs that use different scripts, such as Hindi and Urdu, where it outperforms other sub-word and word-level models (Kunchukuttan and Bhattacharyya, 2021). However, to the best of our knowledge, previous research in SMT either studied a relatively limited range of merge operations, or opted for one particular number, which was not always reported. Kunchukuttan and Bhattacharyya (2017)

have investigated a small range of merge operations, namely 1,000-4,000, on a variety of languages. The result showed that the choice of appropriate parameter affects the performance of an SMT model by 1.6% on average. Kunchukuttan and Bhattacharyya (2021) further compared the performance of BPE as opposed to characters, morphemes, and orthographic syllables in pivot-based SMT. In most cases, BPE yielded best results; however, the authors did not disclose the number of merge operations used in order to produce the sub-word encodings. Lastly, BPE was also used in the Joanis et al. (2020) baseline SMT model, where the authors chose 5,000 merge operations for the Inuktitut to English model and 30,000 merge operations for English to Inuktitut. Even though Joanis et al. (2020) have not addressed their choice of merge of operations for the mentioned translation directions, we believe that bigger number of merge operations can result in better accuracy of a Inuktitut to English phrase-based SMT model. We believe that this might be the case since 5,000 merge operations on a language as morphologically rich as Inuktitut leaves numerous morphemes *unmerged*, thus possibly affecting the alignment of the corpora.

4.2 Setup

We firstly trained a 6-gram English Language Model (LM) on the Train set using the SRILM toolkit (Stolcke, 2002). Unlike the 10-gram LM for BPE in Kunchukuttan and Bhattacharyya (2017), we had to settle for a 6-gram LM due to system limitations. We then used the Moses toolkit (Koehn et al., 2007) to train phrase-based SMT models with the *grow-diag-final-and* heuristic parameter following Kunchukuttan and Bhattacharyya (2017). We used the trained model to translate the Development set and compared the produced translation against the target Development set to determine the performance of the model using the BLUE score metric. We lastly tuned the best performing model trained on BPE-encoded data for 8 [1] epochs using Batch MIRA (Cherry and Foster, 2012). For comparison, we repeated the same procedure with word-level tokenization. We report the results in the following section.

4.3 Results and Discussion

The performance of phrase-based SMT models trained on BPE-encoded data with a range of merge operations can be seen below in 2.

BPE merges	1,000	2,000	5,000	10,000	20,000	30,000
BLEU score	9.97	11.96	14.23	17.11	18.24	18.25

Table 2: BLEU scores on validation set of SMT trained with different BPE merges

The results seem to indicate a clear trend, as a higher number of merge operations appears to lead to overall better performance of the model. The best performance was observed with the highest number of merges, although the performance between the models trained on 20,000 and 30,000 merges BPE differs by only 0.01 BLEU score. Furthermore, neither BPE-based model outperformed the word-based model, which achieved 23.77 BLUE score on the Development set with no additional tuning. We nevertheless attempted tuning the best performing BPE model to learn whether this could at least partially narrow the gap in the performance of BPE and word-based models. We only managed to get the BLEU score up to 19.68. It can hardly be considered a statistically significant improvement. Lastly, we observe that the performance of the word-based model outperforms the Joanis et al. (2020) SMT baseline with BPE by 3.06 BLEU points, as the former model achieved 30.66 BLEU score without fine tuning. Nevertheless, our word-based model falls short from the aforementioned baseline on the Development set by almost 10 BLEU points. We believe that such difference stem from the lack of fine tuning the word-based model on our part, as it would have likely improved the performance on the Development set, yet, due to overfitting, lowered the performance on the Test set.

It appears that our initial hypothesis was confirmed: a higher number of merge operations seems to improve the performance of a Inuktitut to English phrase-based SMT model. However, we also observe that in phrase-based SMT, word-level tokenization outperforms BPE overall, which we find counter intuitive. We believed that BPE would significantly improve the performance of an SMT model when applied on morphologically rich languages.

[1] Note that a smaller number of epochs was chosen since the SMT experiments were conducted on CPU, as to not occupy GPU on Uppmax

Nevertheless, we believe that there might be several plausible explanations of these results. Firstly, it is possible that tokenization using BPE misleads the produced alignments, which are the underlying component of a phrase-based SMT model. As words are broken into sub-word units, the alignments may suffer from too much noice. Secondly, we suspect that despite being used in a BPE-based setup by Kunchukuttan and Bhattacharyya (2017), Moses might lack in terms of robustness in a BPE-based setting, as BPE is never appears in the Moses documentation, and only appears in the source code a handful of times. Perhaps another SMT toolkit, such as Portage[2] used by Joanis et al. (2020), could yield completely different results. We would like to further explore the two possible causes in our future work.

5 Seq2Seq: LSTM

5.1 Theory and methods

LSTM model (Hochreiter and Schmidhuber, 1997) is one of the sequence-to-sequence architectures in NLP which is a simple yet effective architecture for sequence tasks such as machine translation (Sutskever et al., 2014). It is based on Neural Machine Translation architecture which is built of encoder and decoder, with encoder creating a vector from which decoder constructs a translation (Bahdanau et al., 2015). Nevertheless, it is not the most novel architecture and it has been replaced by CNN models and Transformers nowadays. We still trained it for comparison and in order to replicate the study of Ding et al. (2019), as well as test our hypothesis that the number of BPE merges does not have a significant difference for the outcome of the translation while using LSTM architecture. Even in the age of Transformers, LSTM can still be useful as a word segmentation tool to morphologically rich languages like Inuktitut, before performing machine translation with the help of Transformers as done by Le and Sadat (2020). Another important aspect is that for the Transformer architecture to be successful, the dataset has to contain a lot of data and it might not be as successful for low-resource and morphologically rich languages (Pramodya et al., 2020) so it was not clear whether it would outperform LSTM model in our case and it was interesting to test. Regarding the research on LSTM with different merges of BPE, Denkowski and Neubig (2017) argued that 32,000 BPE merges might be most suitable for LSTM architectures, whereas Cherry et al. (2018) found that the bigger number of BPE merges influences the model in a negative way and makes it perform worse - 32,000 BPE merges performed the worst in comparison to the lesser number of merges. With these contradictory findings in mind, we found it useful to train the model ourselves and see if our results would be any different. Finally, one study found that vanilla LSTMs actually match the performance of Transformers when trained on millions of examples in the dataset, besides having a 20 times smaller training time (Wang et al., 2020), so their existence should not be discarded altogether.

5.2 Setup

The model was implement using *Fairseq* toolkit (Ott et al., 2019)[3], as was the CNN as well as the Transformer. It was first trained with the original parameters used in the paper on the dataset with different BPE merges applied in turn as indicated previously. Once the best performing model was found according to the produced BLEU score (Papineni et al., 2002), the model in question was fine-tuned further to increase the performance. The hyperparameters and the respective results can be found below in the Results and Discussion section.

5.3 Results and Discussion

As was found by Ding et al. (2019) and supporting our hypothesis, different BPE merges did not significantly impact the performance of LSTM architecture. The reported BLEU scores can be seen in the Table 3 below:

[2]Despite our best efforts, we were unable to find any available APIs, repositories, or documentation of Portage
[3]https://github.com/pytorch/fairseq

BPE merges	1,000	2,000	5,000	10,000	20,000	30,000
BLEU score	31.65	31.29	31.50	32.55	31.99	32.67

Table 3: BLEU scores on validation set of LSTM trained with different BPE merges

The biggest number of BPE merges, 30,000 respectively, performed best but 10,000 BPE merges followed closely afterwards. For comparison, a vanilla LSTM model was trained on pre-processed data with word-level tokenization and the model produced 27.06 BLEU score. Therefore, it is apparent that while there is no clear trend which number of BPE merges should be used, it is important to use BPE in general due to the fact that it increases the BLEU score by more than 4 BLEU score units even while comparing with the poorest performance of data with applied BPE merges.

After this the best performing model with 30,000 BPE merges was fine-tuned by changing or adding one parameter and tracking the performance. If the hyperparameter increased the BLEU score, it was kept and another parameter was changed or added. Making the LSTM bi-directional instead of unidirectional as in the original setup, increased the BLEU score to 33.32. This finding is not surprising as bi-directional entails that both the preceding and succeeding context for the current prediction of the word is taken into consideration (Sundermeyer et al., 2014). The number of embedding dimensions was also increased from 128 in the original setting to 512 to determine whether it would boost the performance, considering that larger dimensionality of vectors representing words would result in better representation of translated sentences. It did not increase the score and mirrored the finding of Britz et al. (2017) who determined that even small dimensional vector embeddings like in the original LSTM architecture produce good results and converges two times sooner than 512-dimensional embeddings. To further test the hyperparameters, 4 encoder and 4 decoder layers were used instead of the initial 1 layer in both the encoder and decoder, regarding the depth of the network. It also increased the training time by 4 times but sadly did not improve the performance and produced a 33.16 BLEU score. In addition, Fan et al. (2020) determined that the dropout impacts the BLEU score significantly (in their research it was by 4 BLEU points) and therefore the optimal dropout should be searched. To our best knowledge, there is no optimal way to guess the dropout correctly and most research determines the dropout rate by trial-and-error method. In this light, with the model being trained in 0.2 dropout value in the original setting, it was changed to 0.4 and 0.1 dropout rates during the fine-tuning stage. The received BLEU scores were 32.38 and 32.89 respectively which did not improve the performance. Further dropout rates were not explored due to the lack of computational resources. Also, the optimizer used in the original setting was Adam (Kingma and Ba, 2015) and it was not experimented to change it due do it being computationally efficient and used a lot by research in the same field.

All in all, the best found architecture was of bi-directional LSTM model trained on data with 30,000 BPE merges. As the last resort to outperform it, the number of epochs were raised from 50 (as in the original setting of all models) to 100 epochs. With the still decreasing loss the model did not converge and produced a BLEU score of 33.35 which was the best result so far. In order to outperform it, the number of epochs can be raised even more in the future research.

6 Seq2Seq: CNN

6.1 Theory and methods

Due to the lack of literature regarding the Seq2Seq CNN architecture for NMT with BPE, the comparison of previous research with our experiments is rather challenging. As an NMT model, the CNN architecture was introduced in (Gehring et al., 2017), outperforming the incumbent models at the time, such as LSTM (Luong et al., 2015) and GNMT (Wu et al., 2016) by margins of 1.6-5 BLEU score points. In the original paper the authors used 40,000 BPE merges, yet did not specify the reasons behind the choice of the setup or the possible effects on the quality of the translations. The models were trained and evaluated on three language pairs, namely, English to Romanian, to French and to German. It was established that CNN is much more computationally efficient compared to the classical RNN model, as it allows for parallelisation. Although showing promising results at the time, CNN was later replaced by the Transformer architecture and was not as extensively researched for NMT tasks. As the

CNN model learns the most prominent features of a language via feature maps, theoretically BPE would facilitate this process as it segments the features beforehand.

6.2 Setup

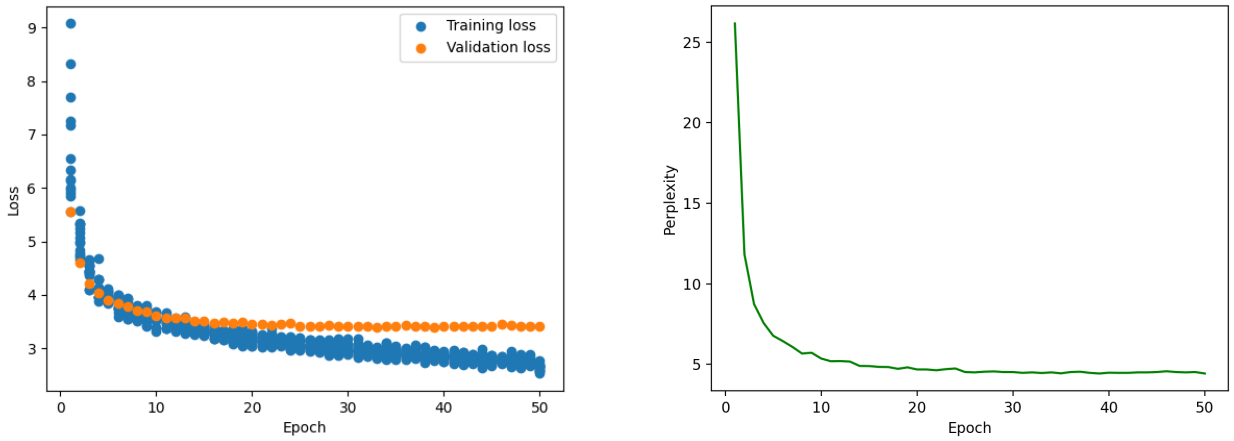
The model[4] was initialised using *Fairseq* toolkit and was first trained on the setup as defined in the original paper. The architecture consists of an encoder and decoder, both 19-layer CNNs. CNN also uses attention mechanisms, called *multi-step attention*, yet unlike Transformer attention, in CNN it is implemented as a dot product between the context from decoder and encoder. It was later fine-tuned by the method of trial and error on varying sets of hyperparameters, such as differing learning rate (from 0.5 to 0.0001), dropout (0.1, 0.2, 0.5 and 0.8), gradient clipping (0.2, 0.5 and 0.8), optimization algorithms (Adam and Nesterov's accelerated gradient method (Sutskever et al., 2013)) as well as label smoothing (0.1 and 0.5). Different training times were also assessed, set to 10, 50 and 100 epochs. The fine-tuning process was done on the 10.000 BPE merges dataset.

6.3 Results and Discussion

In terms of BPE merge effect, the best performance was achieved by using 10,000 merge set, which disproved the hypothesis. Unlike the SMT, the CNN architecture did not present a clear pattern of improvement regarding the merges, as illustrated in Table 4. BLEU scores were worst for 2,000 and 30,000, slightly better for 1,000 and 20,000 with the best BLEU score achieved with 5,000 and 10,000, differing only by 0.11 points. As such the fine-tuning phase was conducted on 10.000 due to the best score and 30.000 merges following the hypothesis. Additionally, it was found that the most optimal set of hyperparameters was as proposed in the original paper Gehring et al. (2017). In general, the model was rather parameter sensitive, responding the most to changes in learning parameter and

BPE merges	1,000	2,000	5,000	10,000	20,000	30,000
BLEU score	33.19	31.25	34.04	34.15	33.89	31.39

Table 4: BLEU scores on validation set of CNN trained with different BPE merges



(a) Training and validation loss

(b) Perplexity

Figure 1: CNN trained on BPE with 10 000 merges

optimisation algorithm change and least to dropout and label smoothing. The experimentation with fine-tuning did not result in better BLEU scores, only slightly lower, in cases of small increase in dropout or label-smoothing, much lower, in cases of lowering learning rate, or ended in gradient vanishing, in cases of increase in clip-normalisation. Furthermore, the model was not affected by the additional training epochs as it converged at around 48 epochs,

[4]The architecture can be found here: https://github.com/pytorch/fairseq/blob/main/fairseq/models/fconv.py

with the results improving only slightly after 20 epochs, as illustrated in Figure 1. Although the training loss improves after the 20th epoch, the validation loss stays stagnant. Similar case can be seen in terms of perplexity, which indicates that the model is not learning new features after the 35th epoch. As such, the training time for an optimal model takes less than 3 hours, yet because it is possible to parallelize the training on several machines, as done in Ott et al. (2018), it could be improved to under 40 minutes. It has to be noted that during the fine-tuning phase, even though the worst models have achieved 10-15 BLEU scores, the translation themselves were not eligible English, with sentences like "*I would like to say hello to everyone and it is nice to see you again*" translated as "*we would like to have to the bring , we have to the bring , we have to the brere , we have to the bring .*" The translation is less than acceptable, however, the model performance is measured at 15.61 BLEU. This suggests that the classic BLEU score is not the most optimal metric for evaluating the results of NMT tasks. As such, we suggest Joanis et al. (2020) YiSi-0 (Lo, 2019) metric to provide a different view of the translations, which is more in-line with human judgements.

7 Transformer

7.1 Theory and Methods

Transformer based models are useful for various NLP tasks and often tend to perform better than alternative models such as CNNs and RNNs (LSTM and GRU) (Vaswani et al., 2017). Transformers are often also used for machine translation tasks. For instance, Wang et al. (2019) investigate deep transformer models used for machine translation, Araabi and Monz (2020) analyse the optimization problem of transformers for machine translation involving a low-resource language setup. Among other parameters, Araabi and Monz (2020) point out the importance of BPE merge operations for improving transformer based model results. Therefore, finding the right number of BPE merge operations is an important focus of this project.

7.2 Setup

It was decided to use an open-source *Fairseq* toolkit (Ott et al., 2019) and train a transformer model following IWSLT'14 German to English (Transformer)[5] example. However, it is important to note that the model architecture in this project is `transformer` instead of `transformer_iwslt_de_en`. Originally it was planned to train transformer models without using BPE and as well train on datasets with different numbers of BPE merge operations. However, due to time constraints and considering additional computational resources needed for that, it was decided to focus on investigating transformer models trained only on datasets with different BPE merges. Moreover, as Araabi and Monz (2020) point out, using BPE for neural machine translation is a standard approach because BPE excels at rare or unseen words.

In order to find the best BPE merges for this model, several models were trained on different BPE datasets. Each model was trained with the same parameters for 11 hours. The best performing model was selected based on the highest BLEU score on the validation set. After selecting the best performing model, it was trained until epoch 50 before doing any fine-tuning. The main reason behind this is to make the transformer based model more comparable to other models used in this project. The best model was fine-tuned by changing the dropout rate and optimizer. All fine-tuning was done by using the best checkpoint of the model trained on the dataset with BPE 10,000 merge operations.

7.3 Results and Discussion

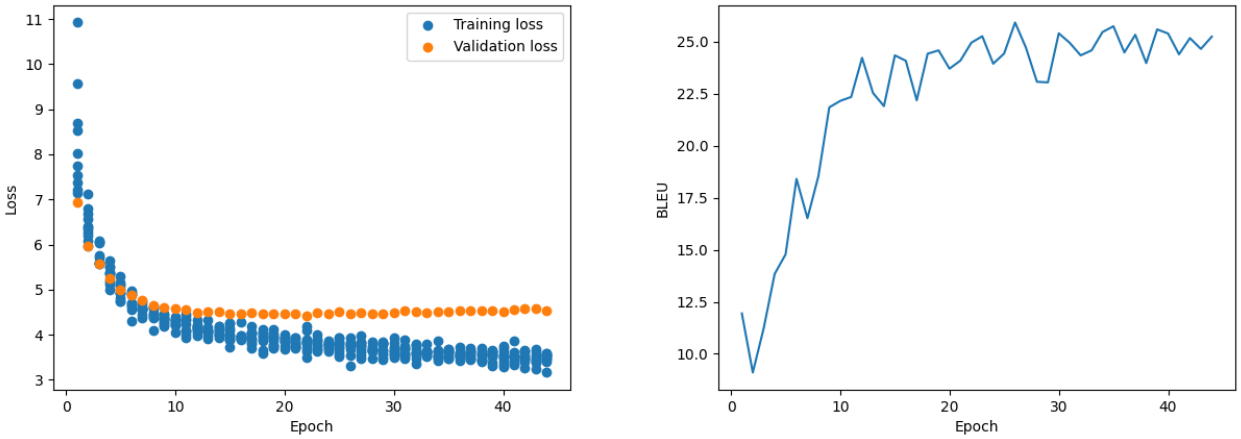
Table 5 shows validation scores of transformer models trained with different BPE merges and the highest BLEU score (model with BPE 10,000 merges) is highlighted. Based on these results, it is recommended using range 5,000 - 20,000 for BPE merges for this transformer. It is likely that the optimal number of merges is in this range. However, it is important to note that the recommendation applies specifically to the Fairseq transformer on this dataset, as different implementations might perform better with varying number of merges. For instance, Joanis et al. (2020)

[5] IWSLT'14 German to English (Transformer) example on Github.

show that transformer implemented in Sockeye performs better with 5,000 merges. In this project, the Fairseq model trained with BPE 10,000 merges was chosen as the best one.

BPE merges	1,000	2,000	5,000	10,000	20,000	30,000
BLEU score	10.88	19.27	21.68	25.93	24.52	23.87

Table 5: BLEU scores on validation set of Fairseq transformers trained with different BPE merges



(a) Training and validation loss

(b) BLEU score on validation set

Figure 2: Fairseq transformer trained on BPE with 10 000 merges

When analysing the results of this model (after training for 11 hours) in detail, it is noticed that the model started to slightly overfit. The gap that is increasing between validation loss and train loss illustrates this (see Figure 2a). In addition, Figure 2b suggests that there is a slight possibility for further improvement as the BLEU score is still showing a small increase. This was confirmed because a slight improvement was reached after traning the model until epoch 50 – 26.07 BLEU on the validation set. As the model showed a tendency to overfit, fine-tuning was focused on tackling this issue. A useful technique to begin with is changing the dropout rate. Dropout randomly drops some nodes together with their connections in each layer during the training phase (Jurafsky and Martin, 2018), meaning that the model is forced to improve the performance of the other nodes. Therefore, dropout was changed from 0.3 (original setting in Fairseq example) to 0.5 and 0.7. However, changing the dropout rate did not improve the model and the best BLEU on the validation set stayed the same (26.07). A possible reason to explain it is that the local minimum could have been reached. Another experiment to improve the score was changing the optimizer from *Adam* to others, such as *Adadelta, Adamax* and *SGD*. It is important to emphasize, that fine-tuning was done by using the best checkpoint of the model, i.e., not the last one. In addition, the optimizer of the model was reset to a new one and the impact of that was investigated. Only one optimizer, *Adadelta*, improved the BLEU score to 26.09 after tuning for 2 hours. Nonetheless, this increase is negligible and is likely to occur due to a random chance. However, it is possible that training the model with this optimizer from the very beginning might boost the BLEU score. It is also important to note that the BLEU score on the test set was 25.81.

As Araabi and Monz (2020) emphasize, finding the optimal parameters for a transformer model is a computationally expensive task. Due to this reason, the model used in this project likely has not reached its full potential. However, further work with transformer based model on Inuktitut language could include changing the model parameters based on Araabi and Monz (2020) study. Some suggested experiments include reducing the number of attention heads, reducing the amount of layers or changing label smoothing.

8 Qualitative analysis

To better evaluate the performance of each model, we also conducted some qualitative analysis. Table 6 shows the differences between translations of the models used in this project. The sentence is taken from test set, the formatting, such as lowercase or tokenisation unchanged. First, it is evident that the SMT, when encountering unknown words, left them unchanged, whereas other models suggest the most likely word in given context. Furthermore, both sequence models were able to retain the general structure of the sentence and indicate the main entities, such as "*constituents*" and "*area*" as "*land*" or "*Bathurst Inlet*" (a location in Canada inhabited by the Inuit people). Nonetheless, the sentences are nonsensical even though the models achieve best BLEU scores compared to other models. Regarding the translation of the Transformer, although failing in retaining the meaning of the sentence, it does portray general fluency unlike SMT or Seq2Seq. As such, although Seq2Seq models outperformed Transformer in regards to BLEU score, qualitatively we evaluate Transformer as a better model in regards to translation readability and usefulness.

Source	kingulliqpaamik uqaqatigillugit niruaqtausimavigijakka arraagulimaaruq utaqijariaqalaurmata qinngurmingaqtit nunamik utaqqijaulutik .
Target	in the last talk i had with the constituents that they had to wait a year , at least , for the surveyors to come in to identify a certain area for their purpose .
SMT	i talked to the final niruaqtausimavigijakka arraagulimaaruq utaqijariaqalaurmata qinngurmingaqtit land utaqqijaulutik .
LSTM	lastly to talk with my constituents in my constituency , they have to wait for a year to wait for a musk ox hunt in the land to wait .
CNN	lastly , in conjunction with my constituents in the next fiscal year they had to wait until bathurst inlet is waiting for a waiting list .
Transformer	during the last session , i have heard from many people that there will be a lot of tourists going out on the land to visit my community .

Table 6: Translations made by SMT, LSTM, CNN and Transformer in comparison to the target sentence

9 Conclusion

The main purpose of this study was to examine the impact of different BPE merges on the performance of four MT models: SMT, two Seq2Seq models, namely LSTM and CNN, as well as the Transformer. For this purpose, the morphologically rich Inuktitut language has been chosen. The results show that no BPE was the best solution for SMT, LSTM performed best with 30,000 BPE merges, whereas CNN and Transformer scored the highest with 10,000 merges. Additionally, unlike SMT and the Transformer, both Seq2Seq models did not show a clear pattern of improvement in regards to the number of BPE merges. Furthermore, the best BLEU score was achieved by the Seq2Seq models, first CNN with LSTM a close second. Nonetheless, the qualitative analysis suggested that, firstly, the Transformer produces more eligible translations, and, secondly, different evaluation metric could be used in future work. It should be added that although the Transformer-based model did not perform particularly well, a possible solution could be reducing model complexity and size. Finally, even though the Transformer is currently regarded in the community as the most efficient architecture, our study has also indicated the value of investigating older models as well.

References

Ali Araabi and Christof Monz. Optimizing transformer for low-resource neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3429–3435, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.304. URL https://aclanthology.org/2020.coling-main.304.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473, 2015.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. Massive Exploration of Neural Machine Translation Architectures. *ArXiv*, abs/1703.03906, 2017.

Colin Cherry and George Foster. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montréal, Canada, June 2012. Association for Computational Linguistics. URL https://aclanthology.org/N12-1047.

Colin Cherry, George F. Foster, Ankur Bapna, Orhan Firat, and Wolfgang Macherey. Revisiting Character-Based Neural Machine Translation with Capacity and Compression. In *EMNLP*, 2018.

Michael J. Denkowski and Graham Neubig. Stronger Baselines for Trustable Results in Neural Machine Translation. In *NMT@ACL*, 2017.

Shuoyang Ding, Adithya Renduchintala, and Kevin Duh. A call for prudent choice of subword merge operations in neural machine translation. In *MTSummit*, 2019.

Yang Fan, Fei Tian, Yingce Xia, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Searching Better Architectures for Neural Machine Translation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1574–1585, 2020.

Philip Gage. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38, 1994.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252. PMLR, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9:1735–1780, 1997.

Eric Joanis, Rebecca Knowles, Roland Kuhn, Samuel Larkin, Patrick Littell, Chi-kiu Lo, Darlene Stewart, and Jeffrey Micher. The nunavut hansard inuktitut–english parallel corpus 3.0 with preliminary machine translation results. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2562–2572, 2020.

Daniel Jurafsky and James H Martin. Speech and language processing (draft). *preparation [cited 2020 June 1] Available from: https://web. stanford. edu/~ jurafsky/slp3*, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980, 2015.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://aclanthology.org/P07-2045.

Anoop Kunchukuttan and Pushpak Bhattacharyya. Learning variable length units for SMT between related languages via byte pair encoding. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 14–24, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4102. URL https://aclanthology.org/W17-4102.

Anoop Kunchukuttan and Pushpak Bhattacharyya. *Machine Translation and Transliteration Involving Related and Low-resource Languages*. 06 2021. ISBN 9781003096771. doi: 10.1201/9781003096771.

Tan Ngoc Le and Fatiha Sadat. Revitalization of Indigenous Languages through Pre-processing and Neural Machine Translation: The case of Inuktitut. In *COLING*, 2020.

Chi-kiu Lo. YiSi - a unified semantic MT quality evaluation and estimation metric for languages with different levels of available resources. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 507–513, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5358. URL https://aclanthology.org/W19-5358.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. *arXiv preprint arXiv:1806.00187*, 2018.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

Mai Oudah, Amjad Almahairi, and Nizar Habash. The impact of preprocessing on Arabic-English statistical and neural machine translation. In *Proceedings of Machine Translation Summit XVII: Research Track*, pages 214–221, Dublin, Ireland, August 2019. European Association for Machine Translation. URL https://aclanthology.org/W19-6621.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Ashmari Pramodya, Randil Pushpananda, and Ruvan Weerasinghe. A Comparison of Transformer, Recurrent Neural Networks and SMT in Tamil to Sinhala mt. *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 155–160, 2020.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. Bpe-dropout: Simple and effective subword regularization. *ArXiv*, abs/1910.13267, 2020.

Andreas Stolcke. Srilm - an extensible language modeling toolkit. In *INTERSPEECH*, 2002.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. Translation Modeling with Bidirectional Recurrent Neural Networks. In *EMNLP*, 2014.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *NIPS*, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1176. URL https://aclanthology.org/P19-1176.

Sinong Wang, Madian Khabsa, and Hao Ma. To Pretrain or Not to Pretrain: Examining the Benefits of Pretrainng on Resource Rich Tasks. In *ACL*, 2020.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.