Enhancing Cherokee-English Translation System

Stanford CS224N Custom Project

Bhagirath Mehta
Department of Computer Science
Stanford University
bmehta18@stanford.edu

David Tran
Department of Computer Science
Stanford University
dtran24@stanford.edu

Abstract

Cherokee is an extremely low-resource language. We propose using transfer learning with NMT models with Inuktitut, a language with similar properties to Cherokee, to enhance BLEU scores. We find that using transfer learning with subword-level embeddings help while transfer-learning with character-level embeddings do not. We aim to optimize our current techniques while exploring other methods for improving BLEU scores, such as data augmentation and hyperparameter tuning. With all of our improvements, we find an improvement of 0.96 BLEU.

1 Mentorship

Our mentor is Professor Christopher Manning.

2 Introduction

Cherokee is an extremely low-resource language with very little literature available in Cherokee. As a result, the parallel Cherokee $\leftrightarrow$ English datasets are also small. This presents multiple difficulties. One difficulty is finding or creating models that are data-efficient enough to translate Cherokee-English data well. Another difficulty is finding useful data augmentation methods that can help expand the parallel data available or present information to a given model in a useful representation.

Zhang et al. helped to create part of the dataset that we analyze here, and they perform machine translation experiments of their own on the dataset [1]. Their best in-domain result is 15.8 BLEU for the Chr-En direction. Different methods they use are SMT (phrase-based), NMT (RNN-based and Transformer-based), supervised and semi-supervised learning, transfer learning, and multilingual joint learning. While it's evident that Zhang et al. try to use various techniques, it's less clear how much they tinkered with each technique to adapt to a low-resource setting.

There seems to be room to explore some of the ideas presented by Zhang et al. in more depth. For example, the languages chosen in the multilingual setting do not seem to be especially pertinent to Cherokee, so using human prior information to select a language could provide benefits. Additionally, Zhang et al. do not specify sets of hyperparameters explored with their models. It seems that more value can be found here by hyperparameter tuning on important factors like vocabulary size.

In our approach, we aim to address these issues. More specifically, we use transfer learning with an Inuktitut-English parent model because we believe Inuktitut may have useful similarities with Cherokee. Additionally, we attempt to enhance our transfer learning system by using hyperparameter tuning and different types of model embeddings.

3 Related Work

3.1 Transfer Learning for Low-Resource Neural Machine Translation

Applying transfer learning to baseline NMT models has been shown to improve BLEU scores in low-resource settings [2]. The crux of transfer learning is to train a parent model on high resource parallel datasets and use parameters from the parent model to initialize the child model that will be trained on low resource parallel datasets. Zoph et al. use French as the parent source language and experiment on Huausa, Turkish, Uzbek, and Urdu for the child source languages. The target language for all the models is English. The addition of transfer learning to baseline NMT models show an average improvement of 5.6 BLEU. Moreover, unknown word replacement and ensembling techniques push the average BLEU improvement to 7.5.

3.2 Adapting to Low-Resource Neural Machine Translation Settings

Results have shown that neural machine translation models underperform phrase-based statistical machine translation or unsupervised methods in low-resource settings [3]. Sennrich et al. revisit past NMT experiments, adapt these NMT models for low-resource settings, and show significant BLEU score improvements [4]. Sennrich et al. attempt four different kinds of adaptations. The first set of adaptations are concerned with model architecture and "training tricks". This includes label smoothing, word dropout, normalization, and tied embeddings. The second set of adaptations manage language representation and size of vocabulary. Thirdly, the adaptations consist of hyperparameter tuning. Fourthly, the adaptation uses a lexical model. More specifically, this adaptation trains a feed-forward network (the lexical model) together with the original neural machine translation model. In aggregate, the adaptations lead to a BLEU improvement of 9.4. Reducing vocabulary size (+4.9 BLEU) and aggressive word dropout (+3.4 BLEU) lead to the biggest improvements.

3.3 Copied Monolingual Data

To help with neural machine translation in low-resource conditions, using copied monolingual data can help to augment a smaller parallel dataset. More specifically, for a given parallel dataset, monolingual data can be added such that the new sentences are the same on the source and target sides [5]. Currey et al. use this technique for Turkish $\leftrightarrow$ English translation and Romanian $\leftrightarrow$ English translation, both representing low-resource settings. Improvements of up to 1.2 BLEU are procured. Currey et al. note that using larger-sized monolingual datasets are helpful, helping provide 0.2-0.6 BLEU.

4 Approach

We utilized a transformer for neural machine translation (NMT), using self-attention in order to allow each token to be considered in terms of the context of the tokens before and after it. Self-attention solves issues with gradients vanishing for longer sentences, and has performed better than recurrent neural networks (RNNs).

As shown in the picture above, we take a source sentence, for example, in Cherokee, and split the sentence up into subwords. Using a mapping of subwords to unique indices, we can create embeddings that are $\mathcal{R}^{1024x1}$ dimensions. As these embeddings pass through the bidirectional encoder, we generate hidden states and cell states. The final hidden state and cell states is linearly projected to form the decoder's initial state. We have 1024 hidden states and m is the length of the longest sentence in the training set for each of our experiments.

$h_i^{enc} = [\overleftarrow{h_i^{enc}}; \overrightarrow{h_i^{enc}}]$ where $h_i^{enc} \in \mathcal{R}^{2048x1}, [\overleftarrow{h_i^{enc}}, \overrightarrow{h_i^{enc}}] \in \mathcal{R}^{1024x1}$ from $1 \leq i \leq m$

$c_i^{enc} = [\overleftarrow{c_i^{enc}}; \overrightarrow{c_i^{enc}}]$ where $c_i^{enc} \in \mathcal{R}^{2048x1}, [\overleftarrow{c_i^{enc}}, \overrightarrow{c_i^{enc}}] \in \mathcal{R}^{1024x1}$ from $1 \leq i \leq m$

We provide the decoder with a target sentence, looking up the embedding for each subword and concatenating it with our combined out-put vector (described later) from our previous step and feed this vector into our decoder, along with our previous hidden and cell states to obtain the next hidden and cell states.

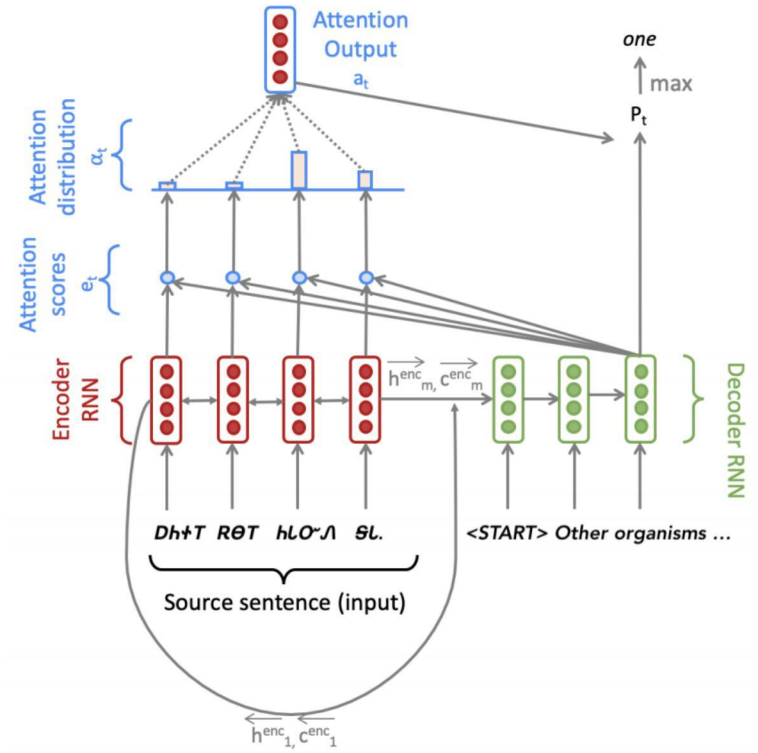


Figure 1: Encoder-decoder framework for NMT [6]

We use the initial decoding state for multiplicative attention over each of the hidden states for each of the t subwords. $e_{t,i} = h_t^{dec^T} W_{attProj} h_i^{enc}$ where $e_t \in \mathcal{R}^{mx1}, W_{attProj} \in \mathcal{R}^{hx2h}$ and $1 \leq i \leq m$

After obtaining e, we run the softmax function over it and take the dot product of this result with each of the hidden states, giving us the final attention.

Concatenating the attention output with the decoder hidden state and passing it through a tanh linear layer and dropout give us the combined-output vector o.

$P_t = softmax(W_{vocab} o_t)$ where $P_t =\in \mathcal{R}^{V_t x1}, W_{vocab} \in \mathcal{R}^{V_t xh}$

V_t is the size of the target vocabulary. We finally take the cross-entropy loss between each subword for P and g (the one-hot vector of the target subword) at each time t as described below:

$J_t(\theta) = CrossEntropy(P_t, g_t)$

Using our loss function, we perform backpropagation in order to accomplish gradient descent.

5 Experiments

5.1 Data

We used four sets of data - a Cherokee to English dataset, an Inuktitut to English dataset, an English to English dataset using the King James Bible from 1611 and an English to English dataset using classics.

Dataset 1: For our Cherokee to English dataset, we used the Chr-En [1] dataset compiled from which came from a series of meeting minutes. We cleaned and extracted the main training set, dev set and test sets provided in the dataset. For most of our models, we pretrained on the Inuktitut to English dataset in order to better prime our model and harness the benefits of transfer learning. In particular,

because Inuktitut is polysynthetic, have a freely free word order and use a syllabary writing system, like Cherokee, we believed that this would give us positive results. We had 1,287,230 pairs of lines in the dataset, which was much more sizable than our Cherokee dataset, but still makes Inuktitut a low-resource language.

Dataset 2: For our Inuktitut to English dataset, we used the Nunavut Hansard Inuktitut-English Parallel Corpus 3.0 [7] which came from a series of meeting minutes. We cleaned and extracted the main training set, dev set and test sets provided in the dataset. For most of our models, we pretrained on the Inuktitut to English dataset in order to better prime our model and harness the benefits of transfer learning. In particular, because Inuktitut is polysynthetic, have a freely free word order and use a syllabary writing system, like Cherokee, we believed that this would give us positive results. We had 1,287,230 pairs of lines in the dataset, which was much more sizable than our Cherokee dataset, but still makes Inuktitut a low-resource language.

In addition, we trained on English. We wanted to prime our model for different English texts. Because our Cherokee to English dataset and was mostly based on the King James Bible and classic books, the English that it picked up from Dataset 1, we wanted to prime it for English that would be closer to the test set, which was comprised from text from the King James Bible and classic books.

Dataset 3: For our second dataset, we used the King James version of the Bible that we found, published in 1611.

Dataset 4: We also found classics published in the 19th and 20th centuries in Project Gutenberg (see Appendix for list and data).

5.2 Evaluation method

In order to keep our method of comparison consistent to the Cherokee, we used bilingual evaluation understudy (BLEU). BLEU is a convenient metric to compare how our machine's translation compares to the sample translation without a qualified human to compare the quality of translations. With only one possible expected translation, and multiple possible ways to translate any given sentence, the BLEU score for a given translation can be low even if the translation is accurate. However, we can still compare the expected English translation to our machine translation and make some qualitative observations.

5.3 Experimental details

We ran each of our languages through a Neural Machine Translation Model. For the model, we used by default a learning rate of 0.0005, 2500 iterations per epoch, a batch size of 32 and a dropout rate of 0.3, unless otherwise specified below.

Since Cherokee and Inuktitut were both polysynthetic languages, and this meant that we would have a large number of unique words, we tokenized the dataset both of these languages, as well as English, splitting each sentence into subwords using byte-pair encoding, or by splitting into characters. This would allow us to input more frequently occurring subunits into the model.

We needed to tokenize each of Cherokee, Inuktitut and English separately using the source training files for Cherokee and Inuktitut and the test training files for Cherokee and Inuktitut. We did not want to combine the source files for Cherokee and Inuktitut because our Inuktitut dataset was much larger than our Cherokee dataset, which would have caused byte-pair encoding to devote a greater portion of subwords to Inuktitut than would have been optimal. As a result, we split Inuktitut and Cherokee separately, then joined the vocabularies together in the same vocabulary file that mapped tokens to numbers so that Inuktitut and Cherokee subword numbers would not overlap. However, we combined all the English datasets together, as we wanted the same English word to be tokenized the same way each time. We could not guarantee this behavior if we tokenized each file separately even with different vocab sizes for each file, so combining all the files made tokenization simpler and guaranteed the same number would be mapped to each token each time in the pretraining, training and testing sets.

5.3.1 Hyperparameter Tuning for Tokenization

When it came to characters, we did not have to make any decisions about how large our vocabulary size should be since we were constrained by the number of unique characters - there were 238 unique characters in the Cherokee dataset, 264 unique characters in the Inuktitut dataset and 77 unique characters in the English datasets combined.

It was necessary to decide the best vocabulary size when tokenizing by subwords. We first tokenized and created a .model file for our source training set for Inuktitut-English dataset separately from the Cherokee-English and English-English dataset in order to determine the best vocabulary size pair for the source and target that would maximize BLEU score for Inuktitut. We trained on the training set, calculating perplexity on the dev set and then evaluating the BLEU score on the test set.

Once we had determined the best vocabulary size for Inuktitut, we separately tuned the parameters for the best vocabulary size pair for the Cherokee source and target set (since the combined English dataset is now larger), now testing for the optimal BLEU score on the dev set to avoid overfitting by choosing our hyperparameters based on the test set.

5.3.2 Transfer Learning

We pretrained on our Inuktitut dataset, then trained on our Cherokee dataset, tokenizing by characters, and then with our optimal subword sizes from our Inuktitut-English dataset and using a source word size of 9000 for Cherokee, we tested out how different epoch sizes during training would affect our results.

5.3.3 Transfer Learning with Copied Monolingual Data

Using the technique cited in 3.3, we set up six experiments that picked up from our model pretrained on the most optimal Inuktitut subword settings.

In Experiment 1, we combined our Dataset 3, the King James Bible with the Cherokee dataset, adding the same lines from the King James Bible in both the Cherokee source file and the English target file.

In Experiment 2, we combined our Dataset 4, the selected classics with the Cherokee dataset, adding the same lines from the selected classics in both the Cherokee source file and the English target file.

In Experiment 3, we combined our Dataset 3 and Dataset 4, the Bible and selected classics with the Cherokee dataset, adding the same lines from the Bible and selected classics in both the Cherokee source file and the English target file.

Experiments 4-6 were similar to Experiments 1-3, respectively, using the same datasets except that we pretrained on the English-English datasets after pretraining on the Inuktitut dataset and prior to training on the separate Cherokee-English dataset.

Report how you ran your experiments (e.g. model configurations, learning rate, training time, etc.) First, we wanted to ensure that we were considering an optimal token size for splitting our pretraining Inuktitut training set into.

5.4 Results

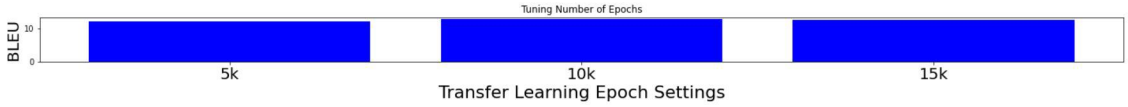
5.4.1 Hyperparameter Tuning for Tokenization

Below, we graph the results for our Inuktitut Tokenization experiments. We label each of our bars with the source and target vocab sizes. We find that the best settings come from tokenizing with 18,000 subwords for the source data and 12,000 subwords for the target data. We obtain a BLEU score of 36.75 for Inuktutit, which is interestingly higher than the maximum BLEU score (29.9) we could find for Inuktitut-English translation for experiments not involving tagging or backtranslation. [8]

Below, we graph the results for our Cherokee Tokenization experiments. We pretrain on Inuktitut using our best source vocab size for Inuktitut and then train on Cherokee. We use the same target vocab size for both English datasets for the reasons described earlier in 5.3.1. We label each of our bars with the source and target vocab sizes, or if we used chars for tokenizing both. We find that the best settings come from tokenizing with 19,000 subwords for the source data and 10,000 subwords for the target data. This gave us a BLEU score of 13.02.

5.4.2 Transfer Learning

From our various number of epochs, we saw that a medium number of epochs gave us the best result, with 10,000 epochs gave us a BLEU score of 12.68.



5.4.3 Transfer Learning with Copied Monolingual Data

First, we added different combinations of English datasets to our training data as described above.

Then, we used our different combinations of English datasets for pretraining instead, as described above.

We also tried different subwords for Experiment 1 and 4. However, our results were underwhelming, as our highest score from these experiments came from Experiment 6 with 10k subwords being used in pretraining giving us a BLEU score of 2.28.

6 Analysis

To determine the strength of our best model, we look at how it performs across sentences of different lengths. We first sort the test set by length of English sentences. Then, we split the test set into eight equal parts, and compute the BLEU score on each part.

Sentence Length Range	BLEU
10-38	12.62
39-57	14.26
58-76	11.94
77-91	12.99
92-109	13.41
110-130	11.18
131-162	12.72
163-465	14.63

Figure 2: A table of sentence length ranges and corresponding BLEU scores. Each of the eight sections have an equal number of sentences.

From the table, there's no clear pattern about model performance on sentence length. One possible reason for this is that there may not be a sufficient amount of data for there to be enough discrepancy between the different categories of sentence lengths.

7 Conclusion

It is quite difficult to improve performance on a low-resource language. Given our time and computing constraints, there are many methods that we would have loved to try but did not have the chance to implement.

We would have been interested in refining our pretraining or training with copied monolingual data. We dove deeper into the code for Google SentencePiece to understand how tokenization worked better, and realized that tokenization may have worked better if we had tokenized on the entire Cherokee and English source file combined. However, as stated previously, this would result in a disproportionately higher amount of subwords being in English due to the disproportionately higher amount of English sentences compared to Cherokee text. To solve this issue, we could have limited our English sentences for the tokenization process or augmented our Cherokee data.

Independently of anything else, augmentation may have improved our results as well, as we would have had much more data.

Though the gains were limited, using a NMT with reordered embeddings, which essentially entailed stacking a separate model on top of our original NMT to reorder the sentence order were shown to provide improved results. [9] Given that we anecdotally saw many of our translation errors come from incorrect sentence order, we believe this could improve our BLEU score.

We also believe that backtranslation and meta-learning would have resulted in improved results, as using many high-resource languages to improve the learning for a low-resource language [10] was shown to have empirically provide decent results on a low dataset for Romanian-English translation, although Romanian-English still had more sentences than our Cherokee-English dataset and is not as complex, as it is not polysynthetic.

References

[1] Shiyue Zhang, Benjamin Frey, and Mohit Bansal. Chren: Cherokee-english machine translation for endangered language revitalization. *arXiv preprint arXiv:2010.04791*, 2020.

[2] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas, November 2016. Association for Computational Linguistics.

[3] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, August 2017. Association for Computational Linguistics.

[4] Rico Sennrich and Biao Zhang. Revisiting low-resource neural machine translation: A case study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy, July 2019. Association for Computational Linguistics.

[5] Anna Currey, Antonio Valerio Miceli Barone, and Kenneth Heafield. Copied monolingual data improves low-resource neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, pages 148–156, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[6] Elissa Li and Chris Manning. Cs224n assignment.

[7] 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 *European Language Resources Association*, 2020.

[8] Rebecca Knowles, Darlene Stewart, Samuel Larkin, and Patrick Littell. NRC systems for the 2020 Inuktitut-English news translation task. In *Proceedings of the Fifth Conference on Machine Translation*, pages 156–170, Online, November 2020. Association for Computational Linguistics.

[9] Kehai Chen, Rui Wang, Masao Utiyama, and Eiichiro Sumita. Neural machine translation with reordering embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1787–1799, Florence, Italy, July 2019. Association for Computational Linguistics.

[10] Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

A Appendix (optional)

Books used for Dataset 3

Title	Year Published
Frankenstein; Or, The Modern Prometheus by Mary Wollstonecraft Shelley	1818
Pride and Prejudice by Jane Austen	1813
Alice's Adventures in Wonderland by Lewis Carroll	1865
A Modest Proposal by Jonathan Swift	1729
The Great Gatsby by F. Scott Fitzgerald	1925
A Tale of Two Cities by Charles Dickens	1859
Et dukkehjem. English by Henrik Ibsen	1879
Metamorphosis by Franz Kafka	1915
The Importance of Being Earnest: A Trivial Comedy for Serious People by Oscar Wilde	1895
The Yellow Wallpaper by Charlotte Perkins Gilman	1892