How Low Can You Go? A Case Study in Extremely Low-Resource NMT

Stanford CS224N Custom Project
Mentor: Prof. Christopher Manning

Anfal Siddiqui
Department of Computer Science
Stanford University
anfal@stanford.edu

Daniel Zhang
Department of Computer Science
Stanford University
dzhang3@stanford.edu

Abstract

Neural Machine Translation (NMT) has improved dramatically in the past decade, with many NMT systems for high-resource languages approaching human-quality translations. However, many of the world's languages are low-resource, with very little digitized parallel data available to train NMT models for them. Although there have been many advancements in developing techniques for low-resource NMT, many languages still have orders of magnitude less data than those used in the associated studies. One such extremely low-resource language is Cherokee, which has less than 15,000 parallel Cherokee-English sentences available. We present a case-study that evaluates the efficacy of common low-resource NMT techniques on Cherokee-English (ChrEn) translation. We analyze the performance of data augmentation, noisy self-training, back-translation, aggressive word dropout, pre-trained word embeddings, pre-trained decoders, mT5, and additional LSTM layers on improving a ChrEn NMT system. We find that pre-training the decoder with 100,000 monolingual English Sentences and back-translation using 5,000 English sentences offer a 0.9 and 0.8 BLEU score improvement over the baseline, respectively, while noisy self-training and aggressive word dropout provide inconsistent benefits in this extremely low-resource setting.

1 Introduction

Recent advancements in neural machine translation model design have produced considerable improvements in both translation accuracy and quality. With these recently released models, machine translations between languages with a large amount of bilingual data (like Spanish to English) have started approaching human translations in both accuracy and quality. However, machine translations for many of the world's endangered languages have not been able to see similar improvements because these languages lack the large corpuses of bilingual data that made models built for more frequently used languages more accurate. While there has been research in the field on overcoming the hurdle of data scarcity on NMT, most of it has focused primarily on languages that, while having less data than the dominant world languages, still have orders of magnitude more data than extremely low-resource languages like UNESCO's endangered languages.

Our work focuses on applying and evaluating many of these techniques for low-resource NMT towards improving a baseline model for translating an extremely low-resource language into English. We chose Cherokee for our case study: with less than 2000 fluent speakers left, the Cherokee language is highly at risk of dying out, and with less than 10MB of digitized bilingual and monolingual data combined, its dataset is extremely small even by low-resource NMT standards. In this study, we look to investigate which of the common techniques used for overcoming data scarcity in low-resource NMT continue to perform well in the extremely low-resource scenario, and what sorts of architectural changes, augmentation, or pre-training techniques most improve the model's

performance. Our motivation for undertaking this task is to help generate tools to assist with the prevention of cultural loss through language loss. Though we know NMT alone will not be the answer to preserving a language, we hope that automated translations can aid in the learning of the next generation of speakers and assist in translating existing works.

2 Related Work

Much of our work is built off of the existing work of Zhang, et. al., whose paper contributed the Cherokee-English dataset our work uses for training [1]. Though this paper helped form the foundation of our approach to tackling the generation of a better extremely low-resource NMT model, one limitation we noticed with it was that the authors did not try many different methods in their research to optimize their Cherokee-English model. Thus, the focus of our work became more directed towards applying the extensive variety of methods developed by recent research for low-resource NMT to tackle the issue of data scarcity.

One of the more promising recently developed methods is self-training, in which more parallel translation data is generated from monolingual sentences by placing each sentence through the forward or backward neural translation model. These generated translations are then fed back into the network as training data for future iterations of the NMT model. The work of Chen, et. al. at Facebook Research was able to increase the BLEU scores of their low-resource models for Burmese to English by anywhere from 3 to 8 points using this method [2]. Transfer learning, where the translation model for the low-resource language is trained on top of a model built for translating other languages, also has repeatedly proven to be highly effective, producing improvements in BLEU scores for low-resource NMT models by up to 7 points in some cases [3]. Other methods tried by low-resource NMT researchers had more mixed results: Qi et. al. found that using pre-trained word embeddings trained using fastText on Wikipedia either greatly improved low-resource NMT models (+11 BLEU score) or had negligible effect on them, with no in-between [4]. One specific minor optimization that we found especially interesting was the idea of aggressive word dropout, where up to half of all the words in a sentence were dropped at random during training. As described in the work of Sennrich et. al., adding this minor technique managed to increase their BLEU score by more than 3 points over their previous model [5].

A major caveat of many of these related papers, though, is that their research was based off of datasets many orders of magnitude larger than the dataset we are working with. Facebook's low-resource NMT paper, for example, involved training a model off of a dataset of 28,000,000 sentences of monolingual source data, while our dataset is composed of $\approx$ 5,000 monolingual sentences. We thus realized that not all of the methods tried in the above papers may remain fully effective on our extremely low-resource NMT setting and would need to be evaluated.

3 Approach

3.1 Baseline

Our baseline model uses a standard 1-layer LSTM architecture [6], with a bidirectional encoder and a decoder that uses multiplicative attention. It operates at the subword-level, with inputs tokenized using a SentencePiece tokenizer [7] that was trained from our parallel dataset. This LSTM architecture was maintained throughout the majority of our experiments, except where specified.

3.2 mT5

We also tried a separate baseline model where we trained the Cherokee-English corpus on top of the weights generated by Google's transformer-based mT5 model [8]. This model was trained with the SimpleTransformers library, and was trained for two epochs with a learning rate of 0.00004.

3.3 Noisy Self-Training

For noisy self-training, we generated more data by putting each source-side monolingual sentence x through forward neural machine translation to produce a translation $\bar{y}$. Noise was added to x by randomly shuffling words, dropping whole words, and inserting blank tokens to randomly replace words, as inspired by [9] - logic we implemented ourselves. This step of adding noise is theorized

to provide a strong regularization effect that is not seen in regular self-training. For three iterations, we produced this noisy dataset using the current model (starting with our baseline model and then using the model produced on the previous iteration for subsequent loops), trained a new model from scratch with it, and then further finetuned it on the original parallel corpus.

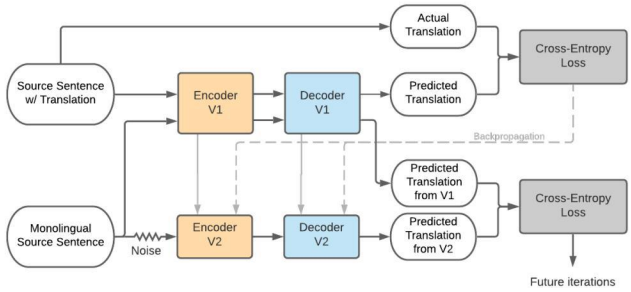

Figure 1: A diagram of our noisy self-training model. Note that the source sentences with translations are still used in second iteration of the translator, as well as all future iterations.

3.4 Back-Translation

Our implementation of back-translation works analogously to self-training, except using target-side monolingual data. We translated each target sentence y to the source language using a target-to-source NMT model to produce $\bar{x}$, which together generates the pseudo-parallel dataset ($\bar{x}$, y). We first implemented and trained an English-to-Cherokee (EnChr) NMT model on our parallel dataset and then used it to produce this pseudo-parallel dataset. We then trained a new Cherokee-to-English model using a combination of the parallel and pseudo-parallel datasets [10].

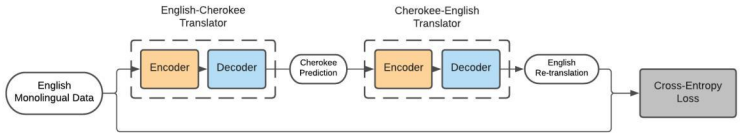

Figure 2: A diagram of our back-translation model

3.5 Iterative Back-translation and Self-Training

We also tried combining back-translation and self-training in an iterative fashion by generating pseudo-parallel datasets from both source and target monolingual data on each iteration. We then used these pseudo-parallel datasets in conjuction with the parallel data to train new forward and backward models. In our implementation, we produced the pseudo-parallel datasets using the current models (starting with our baseline ChrEn model and the EnChr model previously implemented for back-translation) and trained new forward and backward models from scratch using the pseudo-parallel and parallel datasets. We experimented with training the models using a combination of all the data and separately training first on the pseudo-parallel dataset and then finetuning on the true parallel dataset.

3.6 Aggressive Word Dropout

We also experimented with aggressive word dropout. As mentioned above, prior work had shown this method can increase BLEU scores of low-resource models by 3+ points [5]. As Cherokee is a polysynthetic language, we believed there was a chance that dropping entire words in the source sentence would result in too much context loss to produce a good translation, so we chose to implement both whole-word dropout and subword dropout in our models.

3.7 Pre-trained Word Embeddings

The work of Qi et al. showed that using pre-trained embeddings can bring modest to large BLEU score improvements. The authors theorized that pre-training made the embedding spaces more consistent, which brings words with similar meanings closer together and creates similar "semantic neighborhoods" between the the two languages. Additionally, the pre-trained embeddings also were shown to have better representations of rare concepts in low-resource languages, resulting in better translations to English [4]. We made use of BPEmb subword embeddings for both Cherokee and English (which were pre-trained on Cherokee/English Wikipedia) and experimented both using and not using the subword tokenizer provided by the BPEmb library to process inputs to our model [11].

3.8 Pre-train Decoder

While there is a dearth of Cherokee monolingual data, we had access to an abundance of English monolingual data that could be used to pre-train the decoder to make it a stronger conditional language model and produce more fluent translations. Previous exploration of similar techniques saw BLEU score improvements of 1.3 points [12]. In our implementation of this technique, we initialized our ChrEn model as usual, but only trained/exercised the decoder portion. We provided the decoder a zero-vector as its initial hidden state, as well as a single zero-vector as its encoder hidden states to remove the effects of attention on the next-word prediction and for ease of implementation. After pre-training, we finetuned the model on our parallel dataset.

3.9 Two Layer LSTM

We next tried two two-layer LSTM models, one with a single layer decoder, and one with a two-layer decoder. Both of these LSTM models were implemented with a two-layer bidirectional encoder and a decoder that utilized multiplicative attention, analogous to our one-layer LSTM baseline model.

3.10 Data Augmentation

We also tried standard data augmentation techniques we implemented ourselves such as randomly permuting text and dropping words on the source side, both "on-the-fly" during training and beforehand (producing up to 50% more data). As Cherokee is a language that does not have a set sentence structure, we expected such text permutation to be a viable route to producing semi-accurate synthetic data.

4 Experiments

4.1 Data

For our NMT task, we used the ChrEn dataset produced by Zhang et. al [1]. This dataset is comprised mostly of text from the New Testament, as well as assorted news articles and children's books like Charlotte's Web. It is made up of a training set of 14,855 parallel Cherokee-English sentences, a dev set of 1,000 sentences, and a test set of 1,000 sentences. We tokenized sentences from this dataset into subwords using SentencePiece before inputting them into our model. We also made use of the 5,210 monolingual Cherokee sentences provided in the ChrEn dataset. We additionally made use of up to 100,000 News Crawl 2017 [1] sentences as compiled by the ChrEn paper authors as English monolingual data, as well as up to 3 million sentences from News Crawl 2007 that we retrieved ourselves.

Lastly, we compiled a separate set of approximately 400,000 English monolingual sentences from Project Gutenberg. We wrote a custom Python script to retrieve the top 100 English books directly from Project Gutenberg's website and tokenized the raw text into sentences using NLTK [13].

4.2 Evaluation Metric and Experimental Details

We used BLEU scores (computed using SacreBLEU [14]) as our means of evaluating the performance of our ChrEn models.

[1] http://data.statmt.org/news-crawl/en/

Our experiments were ran on Microsoft Azure machines, and had an initial learning rate of .0005 (with a learning rate decay factor of 0.5) and a dropout rate of 0.3. We used a batch size of 32 for the majority of our experiments, but used a batch size of 16 for experiments using decoder pre-training due to memory limitations on our Azure machines. We trained for a maximum of 30 epochs and evaluated perplexity on the dev set every 200 iterations. We used early-stopping if perplexity on the dev set did not improve for 5 trials.

We used the standard NMT mechanism of cross-entropy loss between the log probability distribution of the predicted subword and the target subword, summed over the timesteps of the target sentence.

4.3 Results and Technique Analysis

As seen in 1, our baseline achieved a BLEU Score of 12.5 on the dev set. We evaluate the performance of each of our techniques below.

Name	Architecture	Technique	It	M_s	M_t	M_t Src	WD	S	WB	BLEU	Delta
Baseline	1-L LSTM	-	-	-	-	-	-	-	-	12.5	-
A1	1-L LSTM	On the fly data aug.	-	-	-	-	0.1	3	-	11.7	-0.8
A2	1-L LSTM	Data aug. before	-	-	-	-	0.1	3	-	11.5	-1.0
D1	1-L LSTM	Aggr. WD	-	-	-	-	0.5	-	-	13.2	+0.7
D2	1-L LSTM	Aggr. SWD	-	-	-	-	0.5	-	-	9.8	-2.7
S1	1-L LSTM	Noisy ST	3	5250	-	-	0.1	3	0.2	13.4	+0.9
B1	1-L LSTM	BT	-	-	5K	News Crawl	-	-	-	13.3	+0.8
B2	1-L LSTM	BT	-	-	10K	News Crawl	-	-	-	12.9	+0.4
B3	1-L LSTM	BT	-	-	50K	News Crawl	-	-	-	11.6	-0.9
I1	1-L LSTM	BT + ST (Comb)	1	5250	10K	News Crawl	-	-	-	1.5	-11.0
I2	1-L LSTM	BT + ST (Sep)	1	5250	5K	News Crawl	-	-	-	13.0	+0.5
I3	1-L LSTM	BT + ST (Sep)	3	5250	5K	News Crawl	-	-	-	13.2	+0.7
I4	1-L LSTM	BT + ST (Sep)	6	5250	5K	News Crawl	-	-	-	12.5	+0.0
I5	1-L LSTM	BT + ST (Sep)	1	5250	10K	News Crawl	-	-	-	12.3	-0.2
E1	1-L LSTM	PT WE (F)	-	-	-	-	-	-	-	1.3	-11.2
E2	1-L LSTM	PT WE (U)	-	-	-	-	-	-	-	0.1	-12.4
E3	1-L LSTM	PT WE (U) + BPE	-	-	-	-	-	-	-	1.3	-11.2
M1	Transformer	mT5	-	-	-	-	-	-	-	1.1	-11.4
L1	2-L LSTM	2-L Enc/1-L Dec	-	-	-	-	-	-	-	11.3	-1.2
L2	2-L LSTM	2-L Enc/Dec	-	-	-	-	-	-	-	7.7	-4.8
P1	1-L LSTM	PT Dec	-	-	50K	News Crawl	-	-	-	12.7	+0.2
P2	1-L LSTM	PT Dec	-	-	100K	News Crawl	-	-	-	14.0	+1.5
P3	1-L LSTM	PT Dec	-	-	1M	News Crawl	-	-	-	13.0	+0.5
P4	1-L LSTM	PT Dec	-	-	3M	News Crawl	-	-	-	14.0	+1.5
P5	1-L LSTM	PT Dec	-	-	400K	Gutenberg	-	-	-	9.8	-2.7
C1	1-L LSTM	Noisy ST + Aggr. WD	3	5250	-	-	0.5	3	0.2	13	+0.5
C2	1-L LSTM	Noisy ST + PT Dec	3	5250	-	-	0.1	3	0.2	13.4	+0.9
C3	1-L LSTM	Agg. WD + PT Dec	-	-	-	-	0.5	-	-	9.1	-3.4

Table 1: Results from all of our experiments. It=Number of iterations for techniques where the training loop needs to be repeated (ex: back-translation); M_s=Number of monolingual source sentences; M_t=Number of monolingual target sentences; M_t Src=Source of monolingual target sentences; WD/SWD=Word/Subword Drop Rate; S=Number of word shuffles; WB=Word Blank Rate; BLEU=BLEU Score on Dev Set; Delta=Difference in BLEU Score between this model and the baseline; ST=self-training; BT=back-translation; Comb=Combined training of pseudo-parallel and parallel datasets; Sep=Separate training of pseudo/parallel; PT=Pre-training; WE=Word Embeddings; F=Frozen; U=Unfrozen; BPE=Uses BPE Tokenizer; Enc=Encoder; Dec=Decoder

mT5-Based Model: We received extremely disappointing results when we ran our mT5-based model on the dev set– after two epochs of training, the model was only able to achieve a BLEU score of 1.1. We hypothesize this is because the mT5 model was heavily trained off of multilingual data from mostly European and Asian languages, and that our extremely small corpus of Cherokee data was unable to produce much of an effect on the preexisting mT5 weights. Thus, all of our following models were built on top of the LSTM architecture we used in our baseline.

Noisy Self-Training: Given that our small amount of monolingual Cherokee data was of mixed quality, with many sentences a combination of English and Cherokee, and the baseline model we used to kick of the iterations was not particularly fluent, we expected self-training to perform poorly. Yet, it was among one of our best performing techniques (+0.9 on baseline). What this appears to demonstrate is that the strong regularization effect of noisy self-training that helps NMT models better generalize is not entirely dependent on the amount or quality of the monolingual data it uses.

Back-translation: As the English-Cherokee NMT model we used to drive back-translation had a BLEU Score of $\approx$9, we did not expect this technique to perform well. Yet, back-translation performed

nearly as well as self-training. However, back-translation performed progressively worse as more target monolingual data was added, with the model trained with an additional 50,000 sentences performing worse than the baseline. The poor performance of this model is unsurprising as the pseudo-parallel dataset of likely very poor quality has a nearly 5:1 ratio with the parallel data when using 50,000 target sentences. This may suggest that in these extremely low-resource settings, the benefits of back-translation are best seen when the amount of target monolingual data is less than the parallel data.

Iterative Back-translation and Self-Training: Iterative back-translation and self-training did not fare as well as either technique alone. Experimenting with combining the pseudo-parallel and parallel dataset and then training showed a dramatic drop in performance - likely because the high number of poor quality synthetic samples diluted the impact of the clean ones. Separating training on the two datasets was more effective, but again was dependent on the quantity of target monolingual sentences used, with the clear winner being using 5,000 of those sentences. From our experimentation, 3 iterations of back-translation and self-training produced the highest BLEU score improvement over the baseline (+0.7). As this is worse than just doing noisy self-training or back-translation on their own, it may be that the combined pseudo-parallel dataset - which comes close to matching the size of the real parallel dataset - just has too many poor quality samples in it, which diminishes the benefits of both techniques.

Aggressive Word Dropout: Aggressive word dropout performed very well on the dev set, only slightly under the performance of noisy self-training or back-translation - which is remarkable given how simpler and less computationally expensive this technique is. We expected subword dropout to perform better than dropping whole words, but it performed even worse than the baseline model. It seems that despite Cherokee being a polysynthetic language, dropping subwords does not help with regularization but just serves to be an impediment to training. However, it may also be that dropping subwords at a probability of 0.5 is too high. Further tuning experiments would be needed to make better assessments. Despite this, it seems that aggressive word dropout on its own can be a useful technique even in extremely low-resource settings.

Pre-trained Word Embeddings: Using pre-trained word embeddings performed exceptionally poorly. Freezing or not freezing the pre-trained word embeddings or using the BPEmb tokenizer to better align our model's vocabulary with the embeddings did little to improve this technique. However, its poor performance is not unexpected. Cherokee Wikipedia is known to be of poor quality [1] and this was evident even in the vocabulary of the subword embeddings, which was a $\frac{1}{4}$ the size of our SentencePiece-learned vocabulary and had many English subwords. While other Cherokee word-level pre-trained embeddings exist from FastText [15], they were also trained on Cherokee Wikipedia and did not seem worth trying. Clearly this technique is highly dependent on having good pre-trained word-embeddings. Given other extremely low-resource languages also would lack adequate monolingual text to pre-train quality word-embeddings, this technique does not seem promising in this domain.

Pre-trained Decoder: Pre-training the decoder was by far the most successful approach we tried, with the best model (P2) providing a +1.5 BLEU score improvement over the baseline. Interestingly, the improvements do not scale linearly with the amount of target monolingual data the decoder is pre-trained on, with 1 million sentences performing worse than 100,000 sentences, but 3 million sentences performing roughly equal to 100,000. This may be an artifact of randomness of training and would need further exploration. Given that our target monolingual data was composed of 2007/2017 news data, we believed using text from Gutenberg that was more aligned with the parallel dataset would produce larger gains. However, it caused a drop in performance compared to even the baseline. It is possible that this was due to the quality of the Gutenberg dataset we produced, which was done using a fairly basic web scrape and NLTK tokenization. Despite this, the general success from pre-training the decoder underscores its utility in extremely low-resource settings.

Two-Layer LSTM: Both of our two layer LSTM models performed worse than the baseline, with the model with a 1-layer decoder performing closer to the baseline, and the model with a 2-layer decoder performing roughly 5 BLEU points worse. We surmise this is because we did not re-weight the dropout values properly for the latter model and were not able to perform extensive enough hyperparameter searching to find the best parameters to train both these models with.

Data Augmentation: Both of our basic data augmentation strategies scored worse than the baseline. This was not surprising: as none of the authors of this paper are Cherokee speakers, we resorted

to using naive strategies for producing more parallel data that perturbed the source data without doing anything to the target data. This likely just filled our small dataset with many subpar examples. Clearly, more care must be taken when trying to augment an extremely small parallel dataset.

Combining Techniques: We tried some quick experiments combining a few of our best performing techniques together (C1 to C3 in 1). No combination experiment provided any improvement over using the techniques alone, with combining aggressive word dropout with pre-training the decoder actually doing worse than the baseline. While its possible that these techniques may clash with each other (noisy self-training + aggressive word dropout in particular could be causing too intense of a regularization effect), we believe its more likely that the hyperparameters that worked well for each individually were ill-suited for using them in combination. Deeper experimentation would be needed to make a determination one way or the other.

4.3.1 Evaluation of Best Models on Training Set

Name	Technique	BLEU (Dev Set)	BLEU (Test Set)	Delta
Baseline	-	12.5	12.4	-
P2	PT Dec 100K	14	13.3	+0.9
P4	PT Dec 3M	14	13.1	+0.7
S1	Noisy ST	13.4	12.4	+0
B1	BT	13.3	13.2	+0.8
D1	Aggr. WD	13.2	12.4	+0

Table 2: Delta=Difference between model and baseline on test set

Overall, our best performance on the dev set came from P2 (Pre-training decoder on 100K sentences), which also achieves the best BLEU Score of 13.3 on our test set (+0.9 compared to the baseline). Aggressive word dropout, noisy self-training, and back-translation were also found to be helpful techniques on the dev set.

Table 2 shows a sampling of the best models from our best performing techniques evaluated on the test set. While P2 was still the highest scoring model on the test set, its performance gain over the baseline was somewhat reduced compared to on the dev set. Noisy self-training and aggressive word dropout provided no gains over the baseline on the test set, which seems to indicate that in this extreme low-resource setting, these techniques may not provide consistent improvements for all inputs. However, back-translation remained almost perfectly consistent between the test set and the dev set. The consistency of back-translation but not self-training was suprising given how closely related the techniques are. We suspect the resiliency of back-translation may have to do with the EnChr model we built and the complexities of mapping the two languages: it is possible that the EnChr model captured certain relationship details between the languages that most of our ChrEn models did not. The data that was produced using back-translation, while of low quality, may have also better reflected (if not exaggerated) these other relationships, allowing the ChrEn model that was trained on this data to also capture these relationships, as well as those seen in the original data. As a result, the back-translation model may be better equipped to generalize to various different inputs.

5 Qualitative Analysis

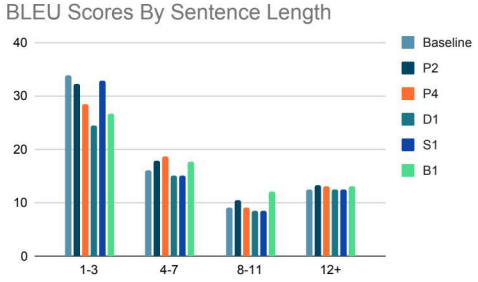

Figure 3: Grouped by length in words of golden English translation

5.1 Best Models By Sentence Length

3 shows the test set BLEU Scores of some of our best models grouped by the length of the sentences. Unsurprisingly, all of the models do progressively worse as the length of the sentence increases. Some particularly noteworthy results are those for P2, D1, and B1. P2 and B1 both do better than other models on longer sentences. The monolingual English data that both were trained with tended to skew towards longer sentences, which may have primed both models to be able to try to translate more parts of longer source sentences, driving up the BLEU score by increasing the number of n-gram overlaps. D1 does noticeably worse than the other models on short (1-3 words) sentences, which we suspect is because dropping words in short sentences during training leaves little for the model to learn from and interferes with it's ability to learn how to properly translate such inputs.

5.2 Translation Examples

- Gold: When I presented the letter and brought up my business—launching quickly into the details of citizenship and ownership of land—the look in Jackson's eyes suggested he might have a killing or two left in him.
- Baseline: I started in the book and I heard what I could not see what I would have very small and their own ones.
- S1: I could see how we could see how we could see how we could see how we could have two eyes on their eyes.
- D1: I told how what I didn't know what what I could do, I told how it was that I could not have some of them with their own legs.
- B1: I had done the books and I'm going to get the books and then what had been all of their eyes—the eyes of their eyes were right.
- P2: When I had received the letter from the book and I told you and whatever it was like Jackson's eyes, Jackson's eyes were able to bear two of them
- P4: When I had finished the letter and said, I was just very far at this <unk>ueens from the <unk>ueens of Jackson's eyes that had been able to stand with their eyes.

When examining their translation outputs for a long sentence, it is clear none of our models produce translations an English speaker would find fluent. However, the Pre-trained decoder models, as we suspected, do demonstrate a better ability to translate more parts of a long sentence: in this example, both P2 and P4 pick up on major words from the beginning (letter) and end (Jackson's eyes) of the sentence - something none of the other models are able to do. All the models, however, struggle to capture any details from the middle of the sentence. In many other translation examples, the models all frequently produce translations with repeated phrases, as demonstrated by this output from P4: "Thou knowest the law, that thou shalt not kill, Thou shalt not kill, Thou shalt not steal, Thou shalt not steal, thy father and thy mother".

6 Conclusion

In this work, we evaluated a wide variety of techniques for addressing data scarcity in extremely low-resource neural machine translation on a Cherokee-English dataset. We trained multiple models using techniques such as self-training, back-translation, pre-trained decoders, two-layer LSTMs, and aggressive word dropout. We found that several of these models were able to produce results with higher BLEU scores than our baseline model, with pre-training the decoder achieving the highest BLEU score on our dataset, and noisy self-training and back-translation-based models coming in close behind. We additionally found that back-translation and pre-training decoders most consistently provide performance gains in this extremely low-resource setting.

Steps for future work include more extensive hyperparameter tuning to see if the performance of the different techniques can be further improved, as well as studying more combinations of the methods we tried. Additionally, further assessing the impact of different monolingual English datasets on decoder pre-training and back-translation would be useful. Lastly, our findings could be replicated on small datasets for other endangered languages/language isolates to see how repeatable our results are in other extremely low-resource settings.

References

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

[2] Peng-Jen Chen, Jiajun Shen, Matt Le, Vishrav Chaudhary, Ahmed El-Kishky, Guillaume Wenzek, Myle Ott, and Marc'Aurelio Ranzato. Facebook ai's wat19 myanmar-english translation task submission, 2019.

[3] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation, 2016.

[4] Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[5] 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.

[6] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[7] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[8] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. pages 1–13, October 2020.

[9] Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. Revisiting self-training for neural sequence generation, 2020.

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

[11] Benjamin Heinzerling and Michael Strube. BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages. In Nicoletta Calzolari (Conference chair), Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 7-12, 2018 2018. European Language Resources Association (ELRA).

[12] Prajit Ramachandran, Peter Liu, and Quoc Le. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 383–391, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[13] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*, 2002.

[14] Matt Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics.

[15] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.