CS 229: Project – Final Report
Machine Translation from Inuktitut to English: Parsing Strategy

Christopher (Egalaaq) Liu (CS 221 & CS 229),
Brian Wang (CS 221 & CS 229),
Yao Yang (CS 229),
Paul Magon de La Villehuchet (CS 221)

Motivation

Automatic translation tools do not yet exist for endangered Inuit languages such as Inuktitut, one of the 8 major Inuit languages of Canada and spoken by around 35,000 people in North Canada.

In addition to the challenge posed by the limited training data available, another major challenge in Inuktitut-English translation is the fundamental difference in the two languages. Inuktitut is a polysynthetic language; its words are made of multiple elementary units (morphemes), added as postbases to a root. Thus, a word in Inuktitut can be equivalent to a whole sentence in English.

Our project will specifically look at how applying morpheme parsing and data augmentation to the Inuktitut input dataset improves Inuktitut to English translation. For CS 229, our experiment will specifically explore different morpheme parsing strategies. As a higher objective, since this is a novel application, we hope that an attention-based seq2seq NMT model will provide a reliable translator from Inuktitut to English.

Related Work

Our experiment builds off of preceding work using the sequence-to-sequence model with attention which has been shown to be state-of-the-art in machine translation tasks. However, our experiment attempts to apply neural machine translation methods to a dataset of limited size (300k lines), in comparison to relatively much larger WMT datasets (2.2m lines). We also experimented with application of the BPE/subword-unit parsing strategy on our Hansard dataset.

Sutskever et al. used deep neural networks for the machine translation on English to French with the WMT-14 (Workshop on Machine Translation) dataset. The paper used a multi-layered LSTM model for both the encoder and decoder. It also found that reversing the order of the source sentences improved performance significantly. (Sequence to Sequence Learning with Neural Networks, 2014)

Cho et al. also used deep neural networks for machine translation from English to French with the WMT-14 dataset, using the computed probabilities from an RNN Encoder-Decoder as an additional feature in a statistical machine translation model. (In Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation, 2014)

Kalchbrenner et al. obtained good results while using recurrent neural networks for machine translation. (Recurrent Continuous Translation Model)

Bahdanau et al. proposed the attention mechanism which we employ in our experiment. The attention mechanism allows a model to focus on different parts of the source sentence when predicting a target token instead of encoding source and target into fixed-length vectors. This experiment was also conducted on the WMT-14 dataset. (Neural Machine Translation by Jointly Learning to Align and Translate, 2015)

Sennrich et al. proposed the sub-word unit (BPE) parsing strategy in order to address the open vocabulary problem. The experiment was conducted on the WMT-15 (English-German) dataset. (Neural Machine Translation of Rare Words with Subword Units, 2016)

Project Pipeline

Dataset

We mainly used the dataset published by the legislative body of Nunavut, Canada, with the help of Benoit Farley and the National Research Council of Canada (NRC). The dataset contains recordings from parliamentary proceedings from April 1, 1999 to November 8, 2007 and includes around 400,000 lines of parallel Inuktitut and English.

Cleaning

To prepare the data for machine translation, we wrote scripts to clean and preprocess the data. The scripts accomplished tasks such as removing empty entries, non-ASCII characters, and unwanted entries caused by webscraping when the dataset was originally created (i.e. web page artifacts). In addition, the dataset was paired down to a corpus with the 50,000 shortest Inuktitut entries in order to speed up the process of hyperparameter tuning.

Data Tokenization

Data tokenization is an important part of preprocessing for machine translation. Indeed, neural networks can only learn

Department of Computer Science,
Stanford University, 2017

a finite number of words and they will show poor performance if the size of the vocabulary is too large. Tokenization allows us to break down similar words into component morphemes and word stems.

Rule-Based Parsing We tokenized our dataset with the help of a rule-based tokenizer, developed by the National Research Council Canada, 2009 (Benoit Farley). For a given sentence, the rule-based tokenizer outputs a list of candidate parsings. We wrote a program in Java that applied the rule-based tokenizer to input files so that vocabulary was broken down into space-delimited components. Because it is not obvious which level of tokenization is optimal, we created two sets of tokenized input files at varying levels of granularity - one with more tokenization (maximum parsing) and one with less tokenization (minimum parsing).

Byte-Pair Encoding (BPE) Parsing Byte-pair encoding (BPE is an unsupervised tokenization method to learn a vocabulary from a dataset. In BPE, the vocabulary is first initialized with all individual characters and then extended repeatedly by merging the most frequent pair of existing vocabulary tokens found in the dataset (Sennrich et al., 2016).

BPE can be applied to source and target datasets individually or jointly. Jointly-applied BPE learns the encoding of the union of the vocabulary of both languages. This improves consistency between the source and target segmentation, in case jointly used tokens are segmented differently between both languages.

We implemented joint BPE encoding to our datasets. We also experimented with various vocabulary sizes in order to understand the level of tokenization that produces the highest accuracy.

Neural Machine Translation

Recurrent Neural Networks Historically, Statistical Machine Translation (SMT) has been the most widely-used and studied machine translation method. However, Recurrent Neural Networks (RNN) have recently been shown to match performance of SMT systems. RNN maintains an internal state, such that the information from previous states can be used to inform future performance.

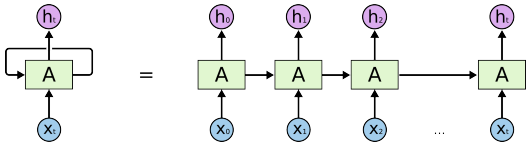

Figure 1: Basic Structure of a RNN

In our case, we used Bidirectional Neural Nets. BRNN differ from RNN by introducing a second hidden layer that flows in the reverse direction. However, RNN have one major shortcoming: RNN cannot learn information over long distances. For this reason, we used a Long Short-Term Memory (LSTM) RNN, which is capable of learning long-term dependencies. They have the same structure as RNN with one major difference: the existence of a cell state which stores information along the way.

Attention Mechanism The attention mechanism is a way to focus on different parts of the input sentence based on what the model has produced so far and is effective in sequential encoding and decoding tasks such as machine translation. This mechanism allows us to address the challenges that arise from Inuktitut-English word alignment. It does not rely on hard-coded rules and is completely automated.

Evaluation Metric: BLEU Score

Bi-lingual evaluation understudy (BLEU) score is a metric for a predicted translation with a reference translation. The simple unigram BLEU score is computed by taking the maximum total count of each predicted word in any of the reference translations, summing the counts over all all unique words in the predicted translation, and dividing the sum by the total number of words in the predicted translation. For evaluation of our experiments, we use a modified BLEU score analysis as implemented in Moses. The multi-BLEU implementation in Moses takes the weighted geometric average of n-gram BLEU-scores $(n = 1, 2, 3, 4)$, and multiplies that result by an exponential brevity penalty factor.

$$BLEU = BP \cdot \left(\sum_{n=1}^{N} \frac{\log(P_n)}{N} \right)$$

Here P_n denotes the precision of n-grams in the hypothesis translation, and the N we use here is 4.

$$P_n = \frac{\sum_{n-gram \in hyp} \text{count}_{clip}(n-gram)}{\sum_{n-gram \in hyp} \text{count}(n-gram)}$$

$$BP = \begin{cases} \exp\left(1 - \frac{|\text{ref}|}{|\text{hyp}|}\right) & \text{if } |\text{ref}| > |\text{hyp}| \\ 1 & \text{otherwise} \end{cases}$$

Experiments

We completed the following three Inuktitut-English dataset experiments on Corn and local machines using the parameters described in the preceding section.

(1) Minimum parsed Inuktitut & Unmodified English

(2) Maximum parsed Inuktitut & Unmodified English

(3) BPE-parsed Inuktitut & BPE-parsed English (joint)

The difference types of parsing were completed using the methods described in the data tokenization section. Additionally, as a supplementary experiment, we evaluated performance after applying different merge counts for BPE parsing. As a proof-of-concept, we also tested performance when adding an attention mechanism, beam search, and increases to batch size.

Results and Analysis

Parameter Tuning

Results To explore parameter tuning, we used the minimum parsing dataset containing the 50,000 shortest lines of

Inuktitut randomly split into training, dev (validation), and test sets, with 45,000/2,500/2,500 lines respectively, with the corresponding English target lines.

Based on TA's recommendations, we conducted experiments varying dropout input keep probability, decoder number of layers, and decoder number of neural units. The model parameters we used for different experiments is described in Table 1.

source_ reverse	dropout input keep probability	decoder # of layers	decoder # of units	BLEU score	eval step
false	0.8	1	128	11.84	50,000 (default)
true	0.6	1	128	12.84	149,000
true	0.4	1	128	11.16	128,000
true	0.8	2	128	11.97	65,000
true	0.6	2	128	12.51	72,000
true	0.6	2	256	11.75	53,000

Table 1 : Summary of all the experiments for parameter tuning

Analysis It was difficult to assess differences between the parameters because BLEU scores tended to fluctuate so that the results had significant overlap in range. We hypothesize this might have been due to the size of the dev and test sets, and that increasing the dev and test set size would have helped to stabilize the BLEU score evaluations. In fact, for our full results, we used dev/test sizes of 5,000 and did not see as much fluctuating in BLEU score evaluations of predicted outputs.

Initially, we also explored tuning of the learning rate parameter, but we found that this was not necessary due our use of the Adam optimizer. Adam adaptively updates learning rates for individual parameters by using moving averages of the mean of variance of past gradients, incorporating an idea similar to stochastic gradient descent with momentum. This optimization strategy enables the algorithm to converge quickly, even if starting from a larger learning rate (Kingma et al., 2014).

We chose to proceed with the default set of parameters for our full experiments, with one change to include source reversing, a decision we made based on previous work that has shown source reversing to improve machine translation performance (Sustskever et al., 2014).

Rule-Based vs. BPE

Results The following plot shows that BPE-parsed data with 10k merges outperformed both rule-based parsing experiments in translation accuracy. The final test evaluations for min-parse, max-parse, and BPE-10k were 2.23/5.90/10.05 respectively (see Table 2.)

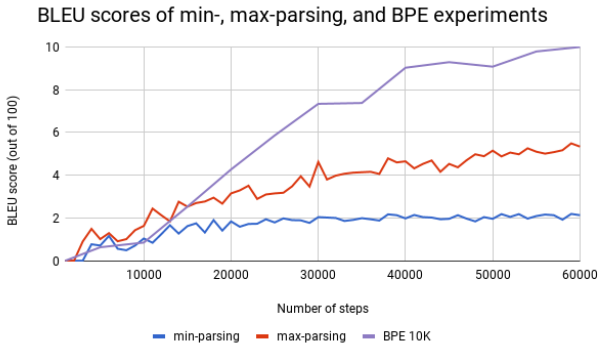

Figure 2: Min/max rule-based and BPE 10k parser results

Dataset	Dev BLEU	@Step	Test BLEU
min-parsing	2.19	59,000	2.23
max-parsing	5.49	59,000	5.90
BPE 10k	10.00	60,000	10.05
BPE 20k	4.03	40,000	4.05
BPE 30k	4.26	48,000	4.41
Without Beam	10.00	60,000	10.05
With Beam	10.01	60,000	10.17
Batch 32	5.49	59,000	5.90
Batch 128	9.93	29,000	10.16

Table 2 : Dev and test BLEU scores for rule-based vs. BPE experiments

Analysis For our rule-based experiments, our models performed poorly likely due to open vocabulary issues. In particular, an additional hyper-parameter that is chosen for neural machine translation models is the vocabulary size. In theory, a larger vocabulary allows for better prediction results since a word must be "in-vocabulary" (as well as commonly occurring enough in the dataset) for it to be learned and predicted.

However, there is also a trade-off between vocabulary size and computational efficiency. The NMT model computes a soft-max over all tokens in the vocabulary at each step, so the larger the vocabulary, the longer the model takes to train and make predictions for each step.

Due to limited computational resources, we chose to constrain the vocabulary size for the rule-based experiments to 30,000 tokens. However, we hypothesize that constraining the vocabulary to 30,000 tokens caused the models to perform more poorly, as there were likely many words that were not able to be predicted due to being "out-of-vocabulary." In contrast, BPE, which doesn't suffer from "out-of-vocabulary" problems was able to exhibit much stronger performance, in line with the work described in Sennrich et al., 2015.

BPE using different merge counts

Results The following plot shows that BPE-parsed data with 10k merges significantly outperformed BPE-parsed

data with 20k or 30k merges. In fact, BPE with 20k or 30k merges also performed more poorly than rule-based max-parsing. Final test evaluations for BPE-10k, 20k, and 30k were 10.05/4.05/4.41 respectively (see Table 2.)

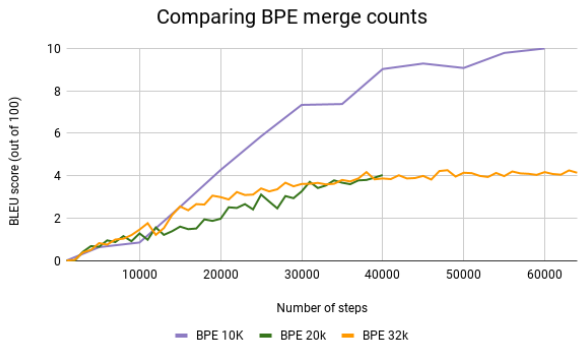

Figure 3: BPE results with and without attention

The following phrases are examples of output with each of these models:

original
Could the member please repeat his question?
maximum parse
I think that was a question that I was asking for a question
BPE 10k
I wonder if he could clarify his question
BPE 20k
Just a response to the minister if you could respond to that
BPE 32k
Maybe if you could ask me if they could explain it

Analysis According to previous work, choice of BPE vocabulary size is somewhat arbitrary (Sennrich et al., 2016). The optimal vocabulary size depends on the language pair and the size of the training data. In our case, choosing 10,000 merges returned the highest translation accuracy.

We hypothesize that for a limited-size dataset with a large number of unique words like Inuktitut, learning too large of a vocabulary size can hurt performance if too many unique words are learned. This hurts the ability of the model to generalize meanings to subword units, which is the original motivation behind parsing.

Supplementary Results and Analysis

Attention Mechanism Proof-of-Concept

Results The following figure demonstrates that using an attention mechanism improved our model. For this comparison, default parameters were used with English as the source and Inuktitut as the target language.

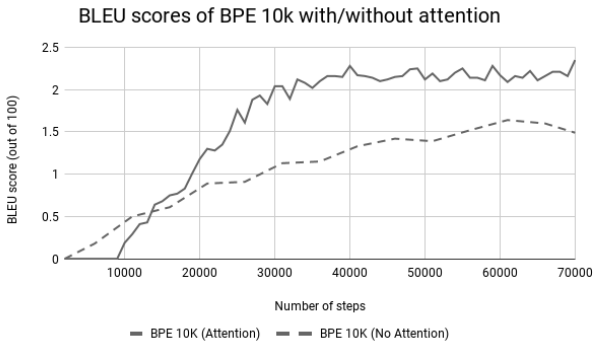

Figure 4: BPE results with and without attention

Analysis Using an RNN with an attention mechanism extends the model by allowing it to automatically (soft)search for parts of a source sentence that are relevant to predicting a target word. Our results are consistent with previous research suggesting that using an attention mechanism will increase translation performance (Bahdanau et al., 2015).

Beam-Search

Results One way to improve prediction is through usage of beam search, which considers 5 "beams" of candidate predictions at each step instead of a greedy approach. Our test BLEU scores on BPE-10k with and without beam search were 10.17/10.05, respectively (see Table 2).

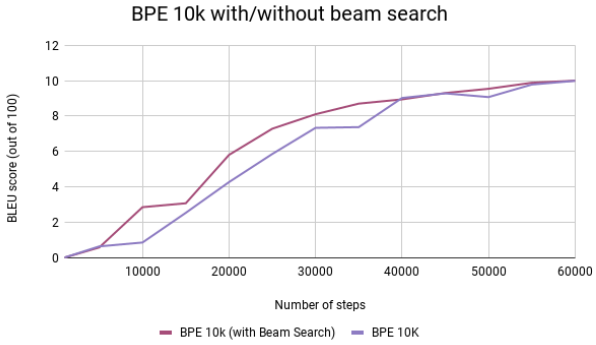

Figure 5: Internal Structure of a RNN

Analysis In our experiment, beam search did not make significantly better predictions; however, it took a significantly longer time to generate a prediction due to the additional searching time. Without additional computational resources, we would not choose to use beam search due to the additional increase is computational cost.

Batch Number

Results An increased batch size allows for better parameter updates due to more accurate gradient approximation, as

the sum of the batch gradients are averaged over the size of the batch. With an increasingly large number of parameters to update in deep neural networks, we hypothesize that batch parameter updates may be increasingly important to help the parameters update in the right direction. We ran two experiments of the max-parsed data using default parameters with different batch sizes of 32 and 128 to explore the differences in performance of varying batch size.

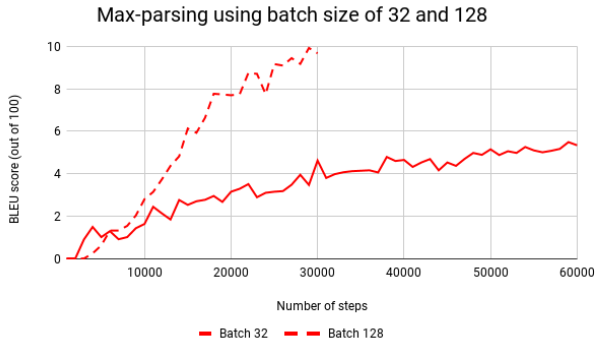

Figure 6: Batch size comparison

Analysis Upon inspection of our plot, we can see that step 40,000 of the batch-32 experiment is only slightly better than step 10,000 of the batch-128 experiment, suggesting that at least for our experiments, there is no evidence that batch-128 updates are significantly better than batch-32 updates or vice versa.

Conclusions and Future Work

In conclusion, using byte-pair encoding to learn a vocabulary outperformed rule-based parsing methods for our dataset, due to being able to handle the open vocabulary challenge that is particularly difficult for Inuktitut, and also due to being able to identify common subword units. Subword units aid in learning meanings and aligning words across languages. However, with limited-size datasets, choosing an appropriate vocabulary size is critical to obtaining optimal performance; more is not always better.

Given more computational resources, we would like to more thoroughly understand the optimal performance achievable through this dataset, as the number of steps we could train our model for was limited.

Also, we believe the rule-based method could potentially be further improved. For example, introducing a method that maps different variations of a single morpheme (e.g. viarr, viar, viat, vian) to a single standard representation (viar), could help with consistent tokenization across the dataset.

As additional future work, we hope to apply the lessons learned from both CS221 and CS229 experiments on similar agglutinative Eskimo/Inuit languages. Our next goal is to apply this to Central Yup'ik Eskimo, a language primarily spoken in Southwest Alaska!

Complementary CS 221 Project

In addition to the CS 229 project, we conducted a complementary project in CS 221 that examines how data augmentation affects machine translation from English to Inuktitut (reversed direction). We experimented with augmenting the Hansard dataset with conversational lines from the Bible, as well as the full Bible.

Including conversational lines did not significantly change performance, while including the whole Bible caused our performance to suffer significantly. We hypothesize that this is because BPE learned common Bible-specific tokens at the expense of less-common Hansard tokens, which hurt prediction on Hansard lines. However, increasing the BPE merge count improved performance on the Bible-augmented dataset. Overall, data augmentation did not result in significant performance improvements from our experiment results.

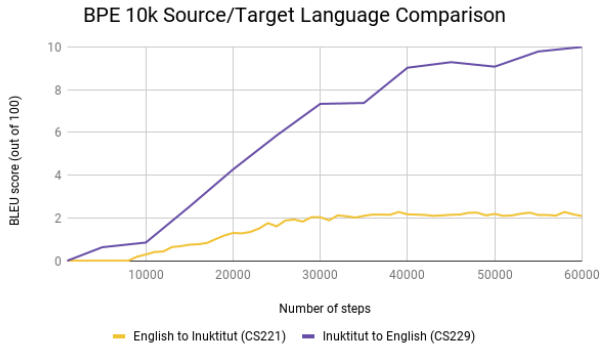

Figure 7: BLEU score v. number of steps comparing language direction

Target/Source Language Direction Our results were notably different after reversing language direction for BPE with 10k. For both datasets, word segmentation occurred more in the Inuktitut dataset than in the English dataset. Thus, we hypothesize that the difference in results is due to the increased difficulty of returning an accurate prediction for a more segmented language. For example, it might understandably be more difficult to predict 30 tokens and place them in the right order than it would to be predict 10 tokens.

The task of translating from English to Inuktitut requires not only the prediction of more total tokens but also the prediction of more tokens in the correct order, to be rejoined after post-processing. In addition, output order of predictions is also critical to correct Inuktitut translation, as increased segmentation also increases the possibility of incorrect rejoining of segmented predictions, which would not occur in a less segmented language, which is more likely to output full tokens that do not need to be rejoined. We hypothesize that the significantly increased amount of segmentation in Inuktitut is the main factor contributing to the difference in performance for prediction between the two directions.

Contributions

Brian

Modified Java program for rule-based parsing of data, data parsing, seq2seq/tensorflow setup, parameter tuning, model training, implementation of BPE, research planning

Christopher

data parsing, data cleaning, parameter tuning, seq2seq/tensorflow setup, splitting data for model training, model training, Bible data preparing, implementation of BPE, research planning

Yao

data parsing, data cleaning, parameter tuning, model training

Paul (CS 221)

Data parsing, Creation of 50k dataset, Parameter tuning, Model training, implementation of BPE

References

(1) Bahdanau, D.; Cho, K.; Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Proc. International Conference on Learning Representations

(2) Cho, K.; Van Merrinboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation

(3) Briggs, J.; Johns, A.; Cook, C. (2015) Utkuhiksalingmiut Uqauhiitigut: Dictionary of Utkuhiksalingmiut Inuktitut Postbase Suffixes. The postbase dictionary we can use to build our own limited parser

(4) Kalchbrenner, N.,Blunsom, P. (2013). Recurrent Continuous Translation Models. In EMNLP (Vol. 3, No. 39, p. 413)

(5) Kingma, D., Ba, J. (2014). Adam: A method for stochastic optimization

(6) Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation. ACL-2002: 40th Annual meeting of the Association for Computational Linguistics. (pp. 311318)

(7) Sennrich, R.; Haddow, B.; Birch A. (2016). Neural Machine Translation of Rare Words with Subword Units. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (pp 17151725)

(8) Seq2Seq Framework
https://google.github.io/seq2seq/

(9) Sutskever, I.; Vinyals, O.; Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112)

(10) The UQAILAUT Project. Accessed Oct 20, 2017. http://www.inuktitutcomputing.ca/Uqailaut/info.php

(11) Yonghui W. et al. (2016) Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation