Yup’ik Eskimo to English: Machine Translation Using Augmented Datasets

Kevin Chavez **Christopher Liu**
Department of Electrical Engineering
Stanford University
{kechavez, cwtliu}@stanford.edu

Abstract

Machine translation tools do not yet exist for the Yup’ik Eskimo language. It is spoken by around 8,000 people who primarily live in Southwest Alaska. With the increasing availability of Yup’ik Eskimo and English parallel text, and a member with fluency of the language in our team, we developed a pipeline to morphologically parse Yup’ik and retrieve corresponding English definitions from an existing Yup’ik-Eskimo Dictionary. An attention-based seq2seq neural machine translation (NMT) model was trained to predict English sentences from Yup’ik input. Our project explored how augmenting the source Yup’ik input, by including retrieved English dictionary entries as input, affects translation performance. Our results found that tokenization alone increases translation accuracy upstream of the NMT model. However, appending retrieved English definitions as input did not improve translation accuracy using our models.

1 Introduction

The Yup’ik language is polysynthetic; its words are made up of multiple subword units, added as postbases to a root. Thus, a word in Yup’ik can be equivalent to a whole sentence in English. As an example, the Yup’ik word *pissuryullrunrituk* translates to the English sentence *"The two did not want to go hunting"*.

Only a small amount of parallel text exists for this historically oral language and no language processing tools have yet been created for Yup’ik. A comprehensive Yup’ik-English grammar book exists and outlines grammatical rules for Yup’ik morpheme construction [8]. These rules were used to develop our rule-based parser, now available here [4].

Yup’ik is also a low-resource language which poses unique challenges and trade-offs for reliable machine translation. Our main research question for CS 230 specifically explored accuracy after augmenting the datasets with additional dictionary lookup information. The models were then trained to output only English predictions. This analysis required building a pipeline to retrieve dictionary entries from parsed Yup’ik entries. Our evaluation was centered primarily on achieving highest performance in accuracy.

In contrast, the complementary project in CS 224n was primarily focused on tokenization as a research question. The bulk of this project was spent developing the rule-based parser. For analysis, parameters were carefully considered in order to make more conclusive arguments about optimal tokenization strategy. Both project groups contributed significantly to parallel corpus dataset preparation.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

There is little to no machine translation research completed on the Eskimo/Inuit language family. The languages are agglutinative, guiding some decision to focus on morphological segmentation as an optimal tokenization strategy for machine translation according to Vandeghinste et al [6]. More recently, unsupervised tokenization methods have been shown to compete with morphologically pre-processed performance for machine translation [12].

Irvine and Callison-Burch explored the use of mixed-domain datasets by including similar languages as single input, similar to how we have combined various Yup'ik dialects. We remained considerate of potential issues caused by combining different input distributions [7].

We chose to use a bidirectional RNN with an attention mechanism according to the built-in seq2seq encoder-decoder framework provided by Tensorflow [9]. Our initial parameters were selected based on model complexity examples from the seq2seq tutorial. Finally, an attention mechanism [5] was used as a way to incorporate information from the input sentence in the prediction layer. It is effective in addressing the word order differences between English and Yup'ik translations.

3 Dataset and Features

Our dataset includes conversational parallel text in Yup'ik and English from approximately 10 books, totaling to roughly 100,000 lines (averaging 18 English words per line), that were manually scanned with object character recognition. The books were used with permission and are edited by Ann Fienup-Riorden, among others. These books contain written transcriptions of interviews with Yup'ik people from various regions across Southwest Alaska. The books primarily represent Coastal and Lower Kuskokwim dialects. In addition, the Bible contains mainly narrative, as opposed to conversational, texts.

Cleaning: To prepare the data for our pipeline, parallel text was manually aligned and scripts were written to cleanse the data. The scripts accomplished tasks such as removing empty entries, non-ASCII characters, book header artifacts, etc. In addition, the dataset was further divided into train/dev/test datasets, using 3,500 randomly selected sentences for each of the development and test sets.

3.1 Data Tokenization

Neural networks can only learn a finite number of words in vocabulary and will show poorer performance if the size of the vocabulary is too large. In particular, polysynthetic languages suffer from this problem.

Rule-Based Parser: For CS224N, we developed a rule-based parser using existing grammar rules [1]. The parser was primarily used to morphologically parse the Yup'ik dataset upstream of an RNN machine translator and to access corresponding English translations [6].

Parsing the word "pissuryullrunrituk": *pissur- @~+yug- -llru- -nrite- +'(g/t)uk*

BPE Parsing: The Yup'ik corpus was parsed using an unsupervised tokenization method called byte-pair encoding. BPE intends to learn a vocabulary set from the data itself by iteratively merging the most frequent token pairs, beginning with individual characters as tokens.

3.2 Dictionary Lookup

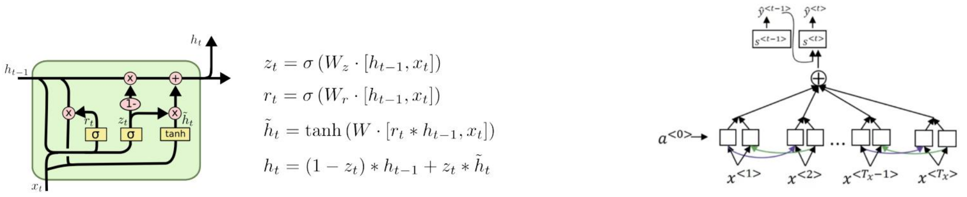
We developed a pipeline that parses Yup'ik words into its roots, postbases and endings, and retrieves their corresponding definitions using Yup'ik to English dictionaries.

4 Methods

Recurrent Neural Networks (RNN) have been widely applied to the application of machine translation. The specific RNN architecture used in this tasks is many-to-many, meaning that it takes as input a variable length sentence and also outputs a variable translation. Due to the low resource nature of this project, the architecture used was of shallow depth. A deeper, two-layer network was used for the best performing run.

Bidirectional RNN: Vanilla RNNs take in as input a sequence to evaluate one word at a time, but avoid important contributions from later units. To address this, bidirectional capabilities are typically added. The architectural change includes additional backward recurrent units to take past and future words into account when making predictions.

LSTM Units: LSTM units are used as part of the network architecture and help ours models "memorize" important information located farther away in longer sentences. LSTMs use separate update and forget gates to decide whether values are to be updated based on inputs. The equations that govern the behavior of the LSTM are show in Figure 1(a).


h_{t-1}
h_t
x_t
r_t
z_t
$\tilde{h}_t$
1-
σ
σ
tanh
$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$
$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$
$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$
$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$
$\hat{y}^{<t-1>}$
$\hat{y}^{<t>}$
$s^{<t-1>}$
$s^{<t>}$
$a^{<0>}$
$x^{<1>}$
$x^{<2>}$
$x^{<T_x-1>}$
$x^{<T_x>}$

(a) LSTM Unit with tanh activation [11]

(b) Attention model with bidirectional units. [10]

Figure 1: LSTM unit and attention mechanism.

The activation values of LSTMs are more easily passed along when connecting many of the units in a recurrent series as was used in our architecture.

Attention Mechanism: An attention model was also used in each experiment as it has generally received much success in recent years [2]. They commonly work with LSTMs to translate longer sentences in a more "human-like" model. A set of attention weights are computed which allow the model to choose a context of words to pay "attention" to during prediction.

BLEU: The bilingual evaluation understudy (BLEU) score was reported as the accuracy metric for machine translation. The BLEU implementation used returns the geometric average of n-gram BLEU-scores ($n = 1, 2, 3, 4$), and multiplies that result by an exponential brevity penalty factor. Here P_n denotes the precision of n-grams in the hypothesis translation, and the N we use here is 4.

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

5 Experiments/Results/Discussion

5.1 Experiments

The project pipeline is displayed in Figure 2. All of our experiments (in Table 1) are grouped into the following three subsets on the pipeline:

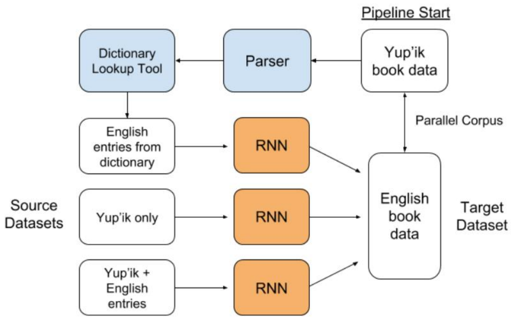


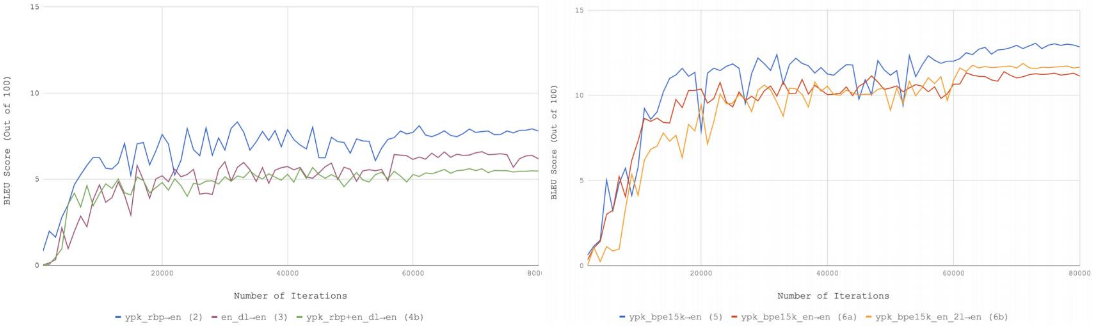
Figure 2: Pipeline for CS 230

Yup'ik Only: Experiments 1, 3, and 6 used the non-augmented datasets as input. Experiment 1 used unparsed Yup'ik as input and was word tokenized using the NLTK toolkit to delimit punctuation. Experiment 3 used rule-based parsed input, generated by our parser. Experiment 6 used BPE with the subword-nmt toolkit [12] using 30k merge operations to generate input.
English Entries from Dictionary: Experiment 3 explored input after parsing Yup'ik words into morphemes and retrieving their corresponding English definitions from existing dictionaries [1]. Only the English definitions were included as input.
Yup'ik + English Entries: Experiments 4, 5, 7, and 8 combined both Yup'ik inputs with English dictionary entries. Experiment 4 appended the rule-based Yup'ik input with the English lookup definitions, delimited by a token placeholder and punctuation removed. Experiment 5 was similar to 4, except that stop words were also removed using the NLTK toolkit [3]. Experiment 7 appended BPE 30k Yup'ik input with English definitions, delimited by a token placeholder and only punctuation removed. Experiment 8 used the same inputs, but with two additional hidden layers.

5.2 Results and Discussion

Experiment	Source	Dev BLEU	Test BLEU
1	Ypk only (NLTK word tokenizer)	9.58	9.02
2	Ypk only (RBP)	8.51	8.33
3	En only (DL)	6.91	6.64
4	Ypk (RBP) + En (DL), punc. removed	5.71	5.79
5	Ypk (RBP) + En (DL), punc. and stop words removed	5.88	5.79
6	Ypk only (BPE 15K)	13.52	12.71
7	Ypk (BPE 15k) + En (DL), punc. removed	12.38	11.39
8	Ypk (BPE 15k) + En (DL), punc. removed, 4 layers	12.45	11.88

Table 1: Experiments and BLEU scores using tuned hyperparameters: learning rate (0.5), exponential learning rate decay, number of layers (2, experiment 8 has 4 layers), number of steps (80,000), maximum sequence length (50), number of units (128), and batch size (128).



(a) Yup'ik rule-based and dictionary lookup runs.

(b) Yup'ik BPE 15k and dictionary lookup runs.

Figure 3: Test BLEU scores for different methods of data augmentation.

Tokenization Without Augmentation: When comparing unparsed, rule-based parsing, and BPE parsing, BPE parsing returned the highest accuracy. Contrary to our hypothesis, using a rule-based parsed input performed worse than the baseline unparsed input. This could be due to unresolved parser matching in the case of some Yup'ik words or due to out-of-vocabulary issues.

Augmenting the Rule-Based Input: As seen in Figure 3(a), Experiment 3 (English dictionary entries) performed slightly worse when compared to Yup'ik only input. This may be due to information loss when removing Yup'ik tokens. Experiment 5 did significantly worse, likely due to an increase in input complexity. Since this experiment included higher average input length and a dual language combination, increased network depth was explored in the later BPE 15k experiments.

Augmenting the BPE 15k Input: We chose to evaluate augmentation on the tokenization method that performed best, BPE 15k, based on our preliminary CS 224n results. Similar to the rule-based set, we saw a drop in performance after appending the English entries. After adding two additional hidden layers, we saw a small amount of accuracy improvement. According to these results, model complexity should be evaluated as an adjustable parameter for higher accuracy.

5.3 Qualitative Analysis of Model Predictions

The models often returned words that were different, yet similar in meaning, compared to reference translation. Although our experimental BLEU scores ranged from 7 to 13 percent, many translations correctly conveyed reference meaning.

For the example below, Experiment 3 was the only prediction that did not reference the "coming of fall", reflecting information loss from Yup'ik replacement. Many examples predicted similar objects in place of hunting "caribou". The augmented rule-based method and BPE methods returned "reindeer" and "fish", suggesting over-influence from dictionary entries. The augmented datasets also had subject and object misdirection, "the reindeer would go and hunt caribou" in Experiment 7.

Exp	Prediction Example
Ref	in the fall , they hunted caribou inland.
1	then when fall came , he 'd bring some caribou back from up there .
2	and when fall came , they 'd hunt caribou from inland .
3	and they 'd start to hunt caribou up there .
4	and when fall came , the reindeer herders became a number of burbot .
5	and when fall came , they 'd be there , and there were a little moose .
6	and when fall came , they would fish up there .
7	and when fall came , reindeer would go and hunt caribou .
8	and when fall came , they 'd go to the ocean from inland .

Table 2: Example predictions

6 Conclusion/Future Work

We explored several methods of augmentation to improve the accuracy of Yup'ik Eskimo translation. Overall, BPE 15k tokenization performed best in terms of BLEU accuracy. Appending with English dictionary lookups decreased accuracy using our models. According to our qualitative review, synonyms were introduced and some prepositional information was lost. These unintended effects will need to be addressed further for data augmentation of English dictionaries to be useful.

As next steps, we will consider reversing the source and target language. We will also consider including word vectorization or hidden layer depth as an adjustable parameter for more complex input. We could also improve the translation predictions using methods such as beam search. Another direction to be explored is an architecture combining predictions from our highest performing models.

Lastly, we are developing a mobile application to integrate tools from both this and the supplementary project. Morphological analysis, dictionary lookups, and reversed translation from English to Yup'ik will be made available to the public. Our hope is that these translation tools can hold back the endangerment of the Alaskan Yup'ik Eskimo language by making access to translations easier.

7 Contributions

Christopher: Morpheme Sampler, Parser Rule Set, Data Cleanse, Align Parallel Text, Book scan/OCR, RNN Runs, Ending Extract, Bible webscrape, BPE/Morfessor app, Ruleset Labeling
Kevin: AWS Setup, Morpheme Sampler, Dictionary Lookup Tool, Data Cleanse, RNN Runs, Dictionary Regex Extraction, Ruleset of Labeling
Laura: Parser (major contributor), Concatenator, RNN Runs, Dictionary Regex Extraction, Project Formulation, Azure Setup, App Developer (supplementary)

References

[1] *Yup'ik Eskimo Dictionary, 2nd edition*. Alaska Native Language Center, 2012.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[4] temigo cwtliu, kechavez. Yupik-mt. `https://github.com/cwtliu/yupik-mt`, 2018.

[5] Bahdanau et al. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[6] Vandeghinste et al. Metis-ii: Machine translation for low resource languages. *Machine Translation*, 22, 2007.

[7] Ann Irvine and Chris Callison-Burch. Combining bilingual and comparable corpora for low resource machine translation. *Association for Computational Linguistics*, 8, 2013.

[8] Steven A Jacobson. *A practical grammar of the Central Alaskan Yup'ik Eskimo language*. 1995.

[9] Minh-Thang Luong, Eugene Brevdo, and Rui Zhao. Neural machine translation (seq2seq) tutorial. *https://github.com/tensorflow/nmt*, 2017.

[10] Andrew Ng. Sequence to sequence - attention model. Stanford University Lecture, 2018.

[11] Christopher Olah. Neural networks, types, and functional programming, September 2015. [Online; posted 3-September-2015].

[12] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Code Repository:
https://github.com/cwtliu/yupik-mt

COMPLEMENTARY PROJECT IN CS224N ATTACHED

Yup’ik Eskimo and Machine Translation of Low-Resource Polysynthetic Languages

Christopher W. Liu **Laura Domine**
Stanford University
{cwtliu, ldomine}@stanford.edu

Abstract

Machine translation tools do not yet exist for the Yup’ik Eskimo language, spoken by around 8,000 people who live primarily in Southwest Alaska. With the increasing availability of Yup’ik Eskimo and English parallel text, as well as a member with fluency of the language in our team, we developed a morphological parser for Yup’ik based on an existing Yup’ik Eskimo dictionary. We trained a seq2seq neural machine translation model with attention to translate Yup’ik input into English. We compared the influence of different tokenization methods, namely rule-based, unsupervised (Byte Pair Encoding) and unsupervised morphological (Morfessor) parsing on the Yup’ik input on BLEU scores for translation. We find that using a tokenized input increases the translation accuracy. Although overall Morfessor did best with a vocabulary size set at 30k, our first trials suggest that BPE performed best using a reduced vocabulary size.

1 Introduction

The Yup’ik language is **polysynthetic**: its words are made up of many morphemes, yielding a high morphemes-to-word ratio. Thus, a word in Yup’ik can be equivalent to a whole sentence in English. As an example, the Yup’ik word *pissuryullrunrituk* translates to the English sentence *The two did not want to go hunting*. Only a small amount of parallel texts exists for this historically oral language and no language processing tools have yet been created for Yup’ik. A comprehensive Yup’ik-English grammar book was published in 1995 by Steven Jacobson, a trained mathematician, and his Yup’ik wife, Anna Jacobson. It outlines grammatical rules for Yup’ik morphology with high mathematical structure. These rules were used to develop a rule-based parser now openly available on GitHub.

Yup’ik is also a **low-resource language** which poses unique challenges and trade-offs for reliable machine translation. Our primary goal was to train an NMT model to reliably translate words from Yup’ik to English. To this end we used state-of-the-art recurrent neural network (RNN) architectures, specifically bidirectional models with an attention mechanism.

Tokenization is usually the first preprocessing step of a machine translation pipeline. Neural networks can only learn a finite number of words in vocabulary and will show poorer performance if the size of the vocabulary is too large. Polysynthetic languages in particular suffer from this issue. Using the rule-based parser that we developed, we compare different tokenization schemes upstream of the NMT model, namely rule-based, Byte Pair Encoding (BPE, unsupervised) and unsupervised Morfessor tokenization using default parameters. Hyperparameters are carefully considered in order to make conclusive arguments about optimal tokenization strategy.

In contrast, the complementary project in CS230 was primarily focused on data augmentation and accuracy improvement. Both project groups contributed significantly to parallel corpus dataset preparation.

2 Related work

The Eskimo/Inuit language family has been the subject of little to no machine translation research. These languages are agglutinative, which justifies a focus on morphological segmentation as an optimal tokenization strategy for machine translation according to Vandeghinste et al [3]. More recently, unsupervised tokenization schemas have been shown to compete with morphologically pre-processed parsing schemas [8].

We chose to use a bidirectional RNN with an attention mechanism according to the built-in seq2seq encoder-decoder framework provided by Tensorflow [5]. The seq2seq tutorial included many informative use cases, model size and complexity chosen for particular corpora sizes. We selected our initial parameters according to these metrics and the agglutinative language type. Finally we added an attention mechanism as a way to incorporate information from the input sentence in the prediction layer. It is effective in addressing the word order differences between English and Yup'ik translations [2].

3 Approach

3.1 Dataset

Our dataset is made of conversational parallel text in Yup'ik and English from 10 books, totaling to roughly 100k lines (averaging 18 English words per line) that were manually scanned with object character recognition. The books were used with permission and are edited by Ann Fienup-Riorden, Alice Rearden, Marie Meade, Eliza Orr, among others. They contain written transcriptions of interviews with Yup'ik people from various regions across Southwest Alaska, which represents primarily Coastal and Lower Kuskokwim dialects. In addition, the Bible contains mainly narrative, as opposed to conversational texts.

To prepare the data for our pipeline, parallel text was manually aligned. Data cleaning was performed with Python scripts to remove empty entries, non-ASCII characters, book header artifacts, etc. In addition, the dataset was further divided into train/dev/test (93/3.5/3.5) datasets using 3,500 randomly selected sentences for each of the development and test sets.

3.2 Tokenization schemes

We coded a rule-based parser using existing grammar rules outlined in [1] [4]. The parser is primarily used to perform morphological parsing of the Yup'ik dataset upstream of an RNN machine translator. This parser is also an important part of the data augmentation pipeline for the complementary project in CS230.

Another tokenization scheme that we included in our comparison is an unsupervised method called Byte Pair Encoding. BPE learns a vocabulary set from the data itself by iteratively merging the most frequent token pairs, beginning with individual characters as tokens.

Morfessor 2.0 toolkit was developed by Sami Virpioja, Peter Smit, Stig-Arne Grnroos, and Mikko Kurimo at Aalto University [9]. It includes an unsupervised morphological segmentation tool that we used in our comparison of tokenization schemes.

We also use the word tokenizer functionality of NLTK which delimits words based on punctuation as our unparsed tokenization scheme.

3.3 Network Architecture

Due to their recent successes, **Recurrent Neural Networks (RNN)** have become widely used for machine translation applications. For this task a many-to-many architecture is usually selected, meaning the input and output are of variable lengths. Due to the low-resource nature of Yup'ik machine translation the architecture has a shallow depth.

Bidirectional RNNs address a typical issue of RNNs: although they take as input a sequence to evaluate one word at a time, they avoid important contributions from earlier units. Additional backward recurrent units take past and future words into account when making predictions.

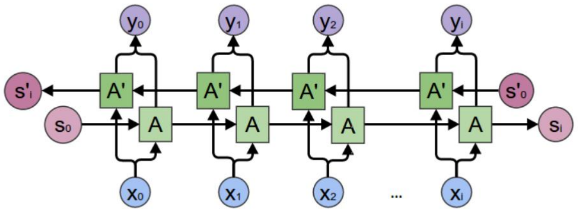
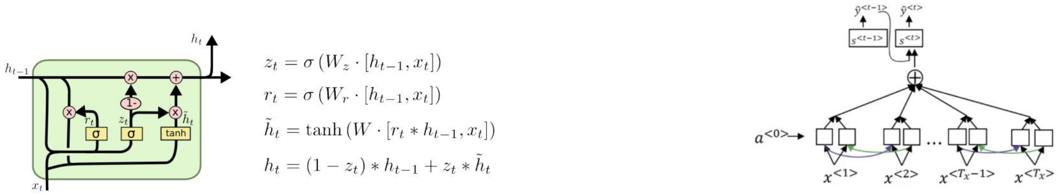


Figure 1: Bidirectional model showing propagation of past and future unit contributions. [7]

Long short-term memory **LSTM** units are used as part of the network architecture. They help our model to memorize important information located farther away in longer sentences. LSTMs use separate update and forget gates to decide whether values are to be updated based on inputs. The equations that govern the behavior of the LSTM are:



h_t
h_{t-1}
r_t
z_t
1-
$\tilde{h}_t$
σ
σ
tanh
x_t
$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$
$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$
$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$
$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$
$\hat{y}^{<t-1>}$
$\hat{y}^{<t>}$
$s^{<t-1>}$
$s^{<t>}$
$a^{<0>}$
...
$x^{<1>}$
$x^{<2>}$
$x^{<T_x-1>}$
$x^{<T_x>}$

(a) LSTM Unit with tanh activation [7]

(b) Attention model with bidirectional units. [6]

Figure 2: LSTM unit and attention mechanism.

Finally an **attention mechanism** was also used in each experiment as it has generally received much success in recent years [2]. They commonly work with LSTMs to translate longer sentences in a more ”human-like” model. A set of attention weights are computed which allow the model to choose a context of words to pay ”attention” to during prediction.

3.4 Evaluation method

The bilingual evaluation study (**BLEU**) is a standard accuracy metric for machine translation. The quality of the machine translation is captured nicely by BLEU since it assumes human-level performance to be the optimal comparison. The BLEU implementation used returns the geometric average of n-gram BLEU scores ($n = 1, 2, 3, 4$), and multiplies that result by an exponential brevity penalty factor.

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

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

4 Experiments

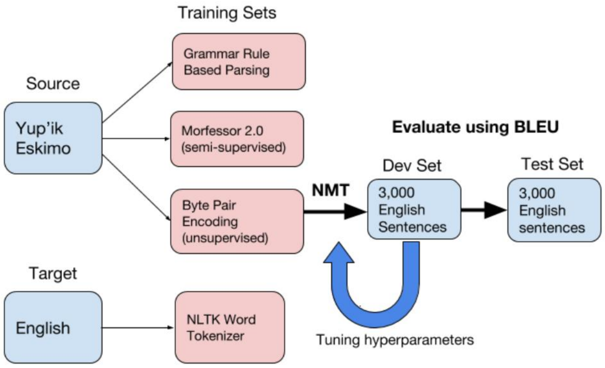


Figure 3: Overview of our pipeline

Our first set of experiments compares tokenization strategies using datasets set to 30k vocabulary size. Experiment 1 used unparsed Yup'ik as input and was word tokenized using the NLTK toolkit to delimit punctuation. Experiment 2 used rule-based parsed input, including unresolved words, generated by our implemented parsing method. Experiment 3 used the Morfessor 2.0 toolkit [9], with default batch training, to tokenize Yup'ik input. Finally, Experiment 4 used byte pair encoding with the subword-nmt toolkit [8] using 30,000 merge operations to generate input.

The second set of experiments were designed to evaluate translation performance using only unsupervised BPE tokenization. We compared results using merge operation counts of 10,000, 15,000, and 30,000. By comparing these datasets using the BPE parsing scheme, and varying vocabulary sizes, we found that 15,000 merges returned highest performance for this dataset.

After hyperparameter search, each model was run with the following parameters: learning rate (0.5), exponential learning rate decay (Luong234 schema), number of layers (2), number of steps (80,000), maximum sequence length (50), number of units (128), and batch size (128).

4.1 Results

Table 1: BLEU at step with highest score (out of 100)

Exp	Source	Dev BLEU	Test BLEU
1	Yupik (Unparsed NLTK)	9.58	9.02
2	Yupik (Rule-based)	8.51	8.33
3	Yupik (Morfessor)	13.33	12.59
4	Yupik (BPE 30k merges)	12.39	11.77
5	Yupik (BPE 15k merges)	13.52	12.71
6	Yupik (BPE 10k merges)	13.19	12.66

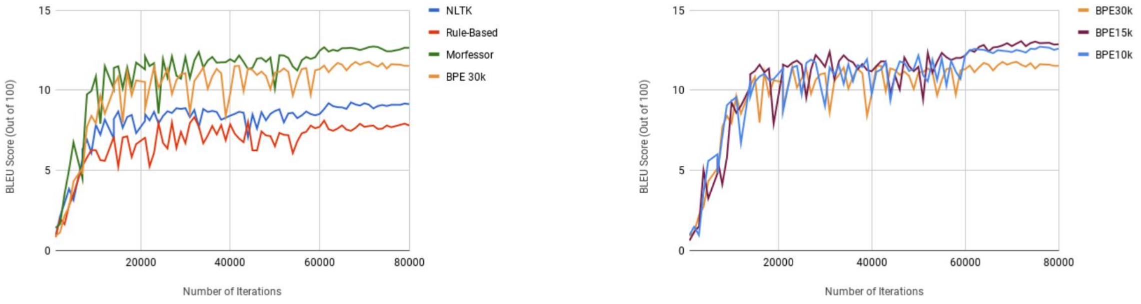
4.1.1 Tokenization Strategy (Experiments 1-4)

Both Morfessor and BPE 30k (unsupervised) datasets outperformed the baseline unparsed NLTK dataset. These results suggest that, after controlling for vocabulary size, parsed methods are preferred over unparsed methods for increased prediction accuracy.

Contrary to our initial hypothesis, the rule-based parsed dataset performed worse compared to the baseline unparsed method. We hypothesize that this could be due either to the remaining unresolved parts of Yupik words that did not match to a particular morpheme in our parser output, or to out-of-vocabulary issues.

4.1.2 BPE Merge Count (Experiments 4-6)

Experiment 5 parsed with BPE using 15,000 merge operations outperformed the other BPE and Morfessor experiments. Vocabulary size is an adjustable parameter that introduces trade-offs between parsing levels optimal for Yup'ik and consistency needed for experimental comparison. For our Yup'ik dataset, which includes mixing of conversational, narrative, and dialectical differences in domains, unsupervised tokenization returned the highest accuracy for machine translation.



(a) Test BLEU for each tokenization method (30k vocab size) (b) Test BLEU for each BPE method with varying vocabulary size

Figure 4: BLEU scores of the test set

4.2 Error analysis

Tables 2-4 contain selected predicted translations using the trained models from all experiments. The models often returned words that were different, yet similar in meaning, compared to the reference translation. In Prediction Sentence 1, none of the models predicted negation words. Although our experimental BLEU scores range from 7 to 13 percents in accuracy, the quality of the translations are fairly sufficient in conveying original sentence information. In addition, the particular experiment type that most closely matched reference translation generally varied.

Table 2: Some example predictions for each experiment

Tokenization	Prediction Sentence 1	Prediction Sentence 2
Original	***i know you are not a good hunter .***	***the dog went hunting and brought food for her .***
Unparsed	*i believe you know you are always alone .*	*the dog went out to the qaygi to hunt .*
Rule-based	*i know you , you poor things .*	*since the dog had n't eaten , he went hunting and hunted with food .*
Morfessor	*you are in the future .*	*his dog would try to obtain his dog to obtain food .*
BPE 30k	*i know you , i am a successful hunter .*	*that one went to dog food for her .*
BPE 15k	*i know you were a good hunter . "*	*his dog started to hunt for food to hunt .*
BPE 10k	*i know you , i know you are hunting .*	*the dog had to use the dog for a long time .*

Table 3: Some example predictions for each experiment (continued)

Tokenization	Prediction Sentence 3
Original	***in the spring, they hauled seals up from the ocean. "***
Unparsed	*in spring , spotted seals always spotted seals . "*
Rule-based	*in spring they had a seal spotted seal oil , and they were unable to catch them . "*
Morfessor	*in the spring , they brought spotted seals up there , they 'd haul spotted seals up there . "*
BPE 30k	*in the spring they were bringing supplies to the ocean in the ocean . "*
BPE 15k	*in spring they brought the spotted seals up in the spring . "*
BPE 10k	*in the spring , they brought grass up to shore . "*

Table 4: Some example predictions for each experiment (continued)

Tokenization	Prediction Sentence 4
Original	***when people were roughhousing in their homes, ghosts would appear.***
Unparsed	*and when we 'd pick berries , we 'd go and get them to our destination .*
Rule-based	*and when she got to the intent , she saw that it was a ghost .*
Morfessor	*and when they 'd constantly have a situation too much , a ghost would surface it too much .*
BPE 30k	*and when they were talking to us in a community , they would come to the land .*
BPE 15k	*and when they discussed something , ghosts would also pull them up to the ghosts .*
BPE 10k	*and when they were about to get too rambunctious , they 'd bring a ghost up .*

5 Conclusion and Future Work

Contrary to our expectations about morphology the tokenization scheme that scored best was not the rule-based parser but an unsupervised tokenization scheme, Morfessor 2.0 toolkit. Our second run of experiments with BPE parsing only points to a targeted vocabulary size that might be best suited for Yup'ik. This could be useful for any future studies that wish to work with this particular language.

Future work could include applying BPE to the rule-based parsed dataset in order to decrease the vocabulary size yielded by the rule-based strategy while retaining its morphological relevance. Another direction worth being explored to improve the translation predictions is postprocessing methods such as beam search and upgrading to more complex models. Ultimately we plan to reverse the language pair in order to perform bidirectional translation. To rule out some unnatural predictions we could use sentiment prediction and combine it with the NMT model prediction to return the translation with the highest subjective score. Finally, we are working on a phone application that could integrate tools from both projects - dictionary lookup, morphological analysis and of course translation between English and Yup'ik - and make them available to the public. Our hope is that this translation tool could fight back language endangerment by facilitating access to translation for Yup'ik language learners.

6 Supplementary Project for CS 230

Project Group: Christopher Liu and Kevin Chavez

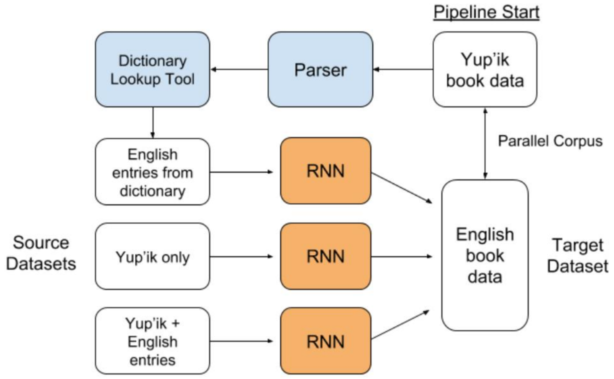


Figure 5: Overview of the CS 230 pipeline

In addition to the CS 224n project, we conducted a complementary project in CS 230 that examines how data augmentation affects machine translation outlined in Figure 5. Specifically, we developed a pipeline that parses Yup'ik words into morphemes and retrieves their corresponding English definitions from Yup'ik English dictionaries [1]. We experimented with appending the Yup'ik inputs with these English dictionary entries, using various start/end token schemes and removing punctuation and stop words.

Overall, data augmentation did not result in significant performance improvements using our methods in CS 230. Including English definitions decreased performance slightly. We hypothesize that this is because adding English definitions made our original Yup'ik input less specific (introducing non-gender specific language and similar, but not exact, English definitions) and also potentially due to insufficient model complexity (since our input increased in size and now contains two separate languages).

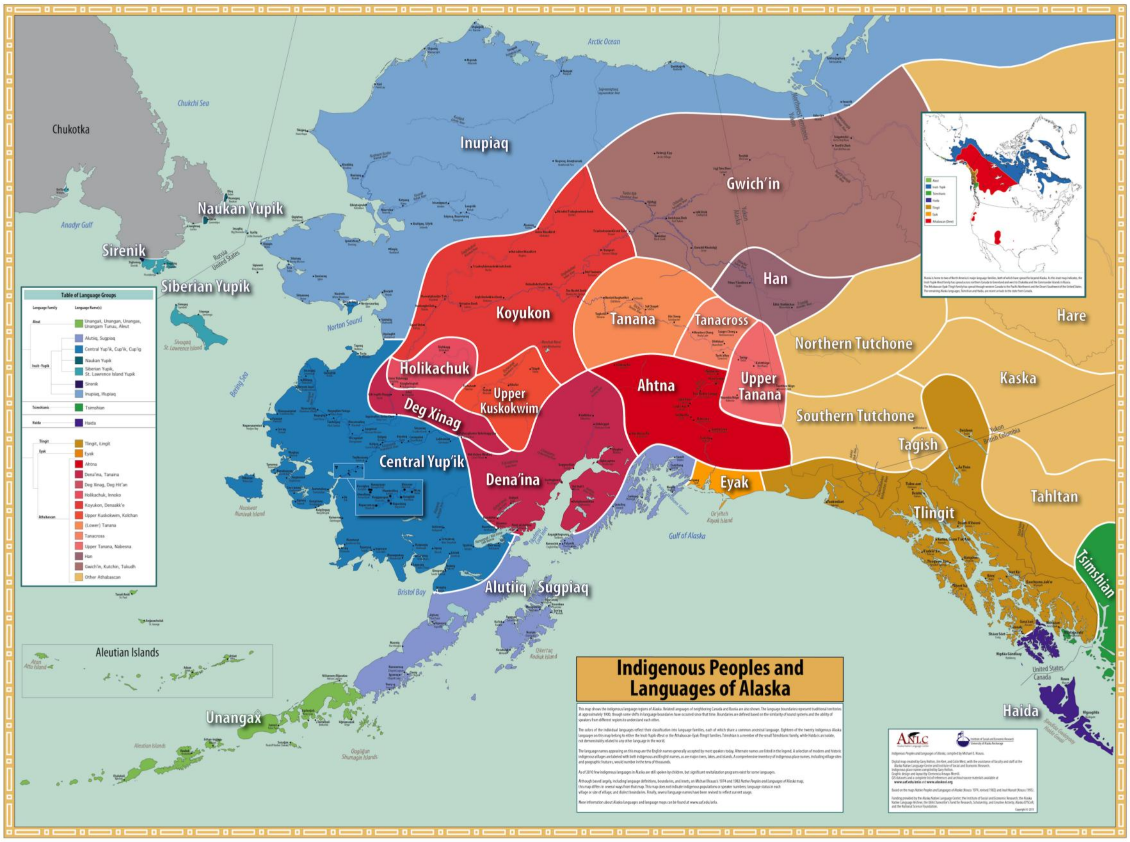


Figure 6: Geographical location of Central Yup'ik language and neighbouring languages influence areas

References

[1] *Yup'ik Eskimo Dictionary, 2nd edition*. Alaska Native Language Center, 2012.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Vandeghinste et al. Metis-ii: Machine translation for low resource languages. *Machine Translation*, 22, 2007.

[4] Steven A Jacobson. *A practical grammar of the Central Alaskan Yup'ik Eskimo language*. 1995.

[5] Minh-Thang Luong, Eugene Brevdo, and Rui Zhao. Neural machine translation (seq2seq) tutorial. *https://github.com/tensorflow/nmt*, 2017.

[6] Andrew Ng. Sequence to sequence - attention model. Stanford University Lecture, 2018.

[7] Christopher Olah. Neural networks, types, and functional programming, September 2015. [Online; posted 3-September-2015].

[8] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

[9] Sami Virpioja, Peter Smit, Stig-Arne Grönroos, Mikko Kurimo, et al. Morfessor 2.0: Python implementation and extensions for morfessor baseline. 2013.