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

Kevin Chavez, Christopher Liu
{kechavez, cwtliu}@stanford.edu

Motivation

- 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 availability of Yup’ik Eskimo and English parallel text, and a member with fluency of the language in our team, we developed a pipeline for reliable translation of this language pair.
- Yup’ik is **polysynthetic** and a **low-resource** language, posing unique challenges and trade-offs for machine translation

pissur-	**@~+yug-**	**-llru-**	**-nrite-**	**+’(g/t)uk**
(to hunt)	(to want)	(past)	(negation)	(2 subjects)

pissuryullrunrituk = The two did not want to go hunting.

Approach

- We built parsing and dictionary lookup tools to retrieve additional information from existing Yup’ik-English dictionaries to augment our datasets upstream of the RNN.

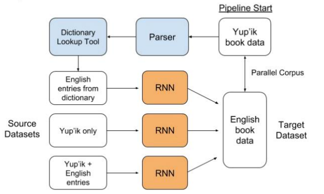

Figure 1: Pipeline of the project

- We evaluated accuracy on various augmented source datasets containing Yup’ik words and English lookup definitions.
- Blue boxes are toolkits we built. White boxes are datasets. Orange boxes are separately trained models.

Data Preparation

Conversational parallel text Yup’ik/English from 10 books (including the Bible), totaling **~100,000 sentences**.

- manually scanned with object character recognition.
- data cleaning: aligning parallel texts, removing empty entries, non-ASCII characters, book header artifacts, etc.
- 93/3.5/3.5 train/dev/test datasets.

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.
- For Yup’ik Eskimo, a polysynthetic language consisting of morphemes (roots, postbases, endings), the following tokenization methods were applied to the dataset:
  - Rule-Based Parsing (RBP) using existing grammar roles
  - Byte Pair Encoding (BPE) as an unsupervised parsing method

Bidirectional RNN

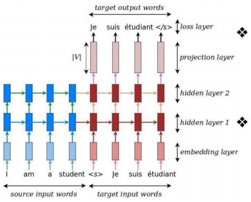

Figure 2: Representation of a Bidirectional RNN
Source: https://github.com/tensorflow/nmt

- **Recurrent neural networks** are state-of-the-art for machine translation tasks. Our method applied a **bidirectional LSTM model with attention**.
- As part of parameter tuning, we explored performance trade-offs ending with learning rate (0.5), number of layers (2), batch_size (128), and exponential learning rate decay.

Experiments

1. Ypk only (NLTK word tokenizer) → En
2. Ypk only (RBP) → En
3. En only (DL)→ En
4. Ypk (RBP) + En (DL) → En
   a. Sentence-Level Start/End Tokens, punc. removed
   b. Sentence-Level Start/End Tokens, punc. and stop words removed
5. Ypk only (BPE 15K) → En
6. Ypk (BPE 15k) + En (DL) → En
   a. Sentence-Level Start/End Tokens, punc. removed
   b. Sentence-Level Start/End Tokens, punc. removed, 2 hidden layers

* Ypk is Yup’ik. En is English. RBP is rule-based parser. BPE is byte pair encoding. DL is dictionary look-up. English was tokenized using the NLTK word tokenizer function.

Exp	Dev BLEU	Test BLEU
1	9.58	9.02
2	8.51	8.33
3	6.91	6.64
4a	5.71	5.79
4b	5.88	5.79
5	13.52	12.71
6a	12.38	11.39
6b	12.45	11.88

Figure 3: Dev and Test BLEU at step with highest BLEU (out of 100), from Yup’ik to English

Results: BLEU Graphs

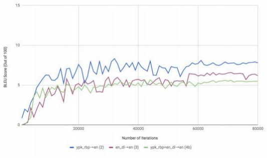

Figure 4: Yup’ik rule-based and dictionary lookup runs

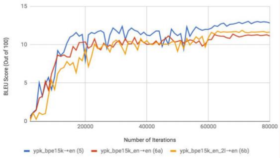

Figure 5: Yup’ik BPE 15k and dictionary lookup runs

Analysis

- Conclusions
  - Tokenization upstream of the RNN improves accuracy.
  - Augmenting the dataset with the English dictionary definitions did not outperform Yup’ik only inputs using our methods.
    - Increased ambiguity when including definitions
    - Model may not be complex enough
- Challenges
  - Out of memory issues when increasing input size
    - Trade-offs when reducing input size (punctuation and stop words)
- Future work
  - Gather more training data.
  - Increase computing capabilities.
  - Experiment with alternative network architectures when combining Yup’ik and English dictionary lookup.

Complementary Project (CS224n)

- Our project was focused on building a rule-based parser and trying various tokenization schemes upstream of the RNN.
- With a set vocabulary size (30k), Morfessor 2.0 tokenizer had highest accuracy.
- When comparing 10k, 15k, and 30k BPE merges, BPE 15k did best.

Acknowledgments & References

Professors Kian Katanforoosh and Andrew Ng, TA: Surag Nair

GitHub Link: https://github.com/cwtliu/yupik-mt

[1] Yup’ik Eskimo Dictionary, 2nd edition. Alaska Native Language Center, 2012.
[2] Bahdanau et al. Neural machine translation by jointly learning to align and translate. ICLR, 2015.
[3] Lample et al. Unsupervised machine translation using monolingual corpora only. CoRR, abs/1711.00043, 2017.
[4] Vandeghinste et al. Metis-ii: Machine translation for low resource languages. Machine Translation, 22, 2007.
[5] Ann Irvine and Chris Callison-Burch. Combining bilingual and comparable corpora for low resource machine translation. Association for Computational Linguistics, 8, 2013.
[6] Minh-Thang Luong, Eugene Brevdo, and Rui Zhao. Neural machine translation (seq2seq) tutorial. https://github.com/tensorflow/nmt, 2017.