**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 polysyntactic and a low-resource language, posing unique challenges and trade-offs for machine translation.
- Passage: **+yup** -**iins** -**erbus** -**+yf** -**tuk** (to hunt) (to want) (past) (preposition) (2 objects) pleysy/putuk = The two did not want to go hunting.

**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 polysyntactic language consisting of morphemes (roots, postbases, endings), the following tokenization methods were applied to the dataset:
  - Rule-Based Parsing (RBP) using existing grammar rules
  - Byte Pair Encoding (BPE) as an unsupervised parsing method

**Bidirectional RNN**

- 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 (25), batch size (128), and exponential learning rate decay.

**Experiments**

1. Yup only (NLTK word tokenizer) → En
2. Yup only (RBP) → En
3. En only (DL) → En
4. Yup (RBP) + En (DL) → En
   a. Sentence-Level Start/End Tokens, punct. removed
   b. Sentence-Level Start/End Tokens, punct. and stop words removed
5. Yup only (BPE 19k) → En
6. Yup (BPE 19k) + En (DL) → En
   a. Sentence-Level Start/End Tokens, punct. removed
   b. Sentence-Level Start/End Tokens, punct. removed, 2 hidden layers

**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:** Rik Kalantoor and Andrew Ng, TA: Suraj Nar

**GitHub Link:** [https://github.com/kechavez/yupik](https://github.com/kechavez/yupik)

**References:**