YerevaNN @ DeepHack.Babel

Gor Arakelyan[1,2], Tigran Galstyan[1,2], Karen Hambardzumyan[1,2], Hrayr Harutyunyan[1,2], Hrant Khachatrian[1,2], Martin Mirakyan[1,3]

[1] YerevaNN research lab, [2] Yerevan State University, [3] American University of Armenia

Problem Statement

Create a machine translation system that can learn on a parallel corpus of 50K sentences and two monolingual corpora with 500K+ sentences for an unknown language pair.

- The only available information about the language is that standard tokenization rules of Western languages are applicable.
- English-Russian parallel corpus was given for local validation of the algorithms.
- The algorithms were tested on two more language pairs on the evaluation server: Hidden 1 and Hidden 2.
- The system was evaluated using BLEU score.

Transformer Architecture

Almost all our experiments were done using Google's Transformer architecture [4]. It does not use recurrent or convolutional layers, instead it applies multiple layers of multi-head self-attention mechanism on the input vectors. Decoder does the same on the right-shifted output vectors (which are known at the training time). In addition to self-attention, decoder has another attention layer that looks at encoder's outputs. The architecture contains many other tricks like positional encoding, layer normalization, shortcut connections etc.

The system is very fast at training, but for inference it still needs to generate the sentence word-by-word (so usually inference is slower than training a single epoch). Transformer does not use pre-trained word embeddings. It learns its own (shared or not) embeddings during training.

We used *Sockeye* framework, Amazon's implementation of Transformer and few other architectures written in MXNet. We used the default parameters for most of the experiments. By default, model selection was performed using perplexity score. For some experiments we switched to BLEU (marked as *Stop@BLEU* in the Results chart), which was much slower (we had to limit the number of sentences on which BLEU was calculated to 1500) but produced better results.

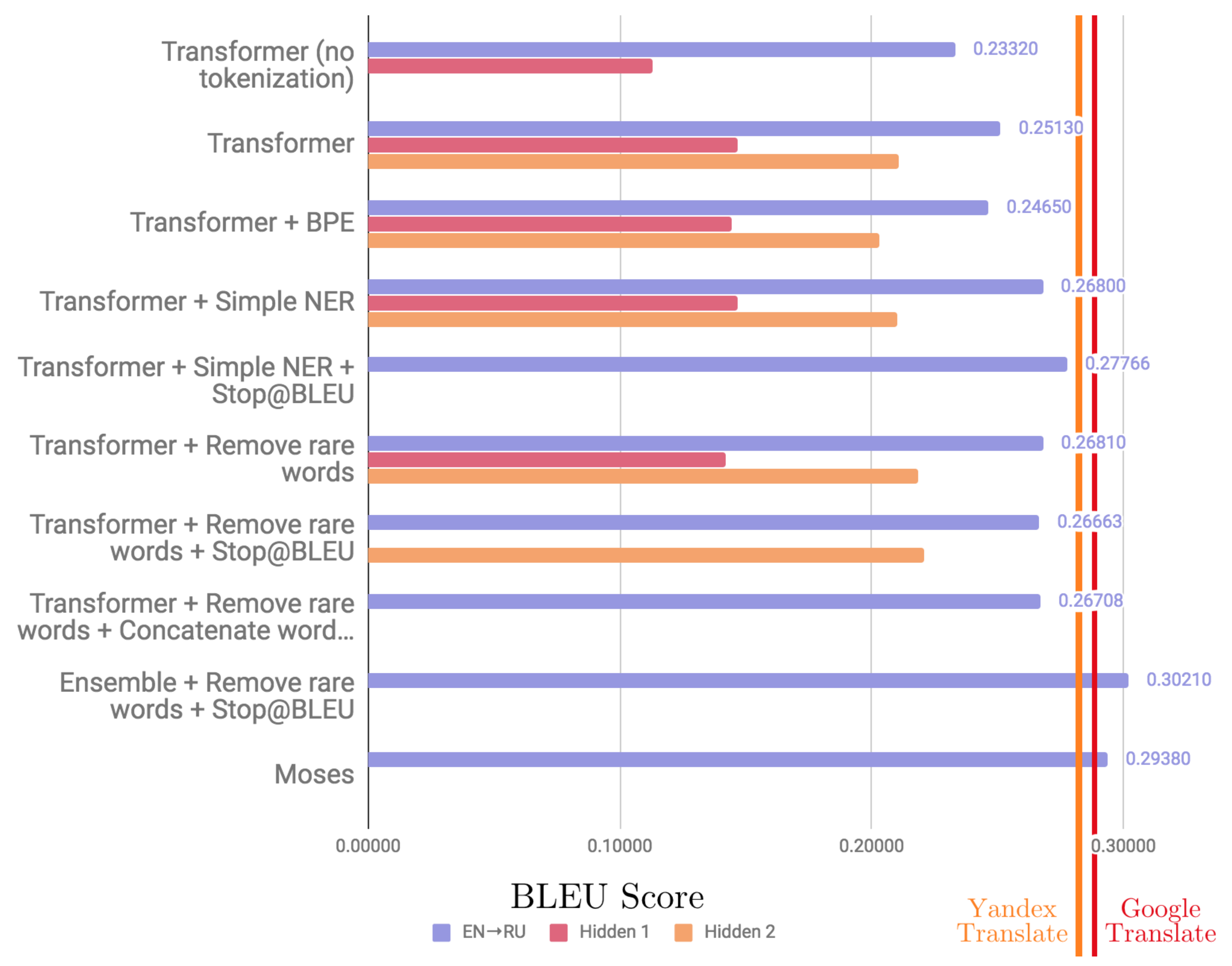
Tokenization

- We used NLTK tokenizer for almost all our experiments.
- We also experimented with Byte-Pair-Encodings of [3] using the implementation from *subword-nmt*. It did not improve the performance on English-Russian pair.

Ensembling

- *Sockeye* also supports ConvS2S and RNN-based architectures.
- We trained all three versions on the same data (NLTK tokenization, removal of rare words, Stop@BLEU) and created an ensemble using Sockeye's built-in implementation.

Our Results



Handling rare words

Transformer architecture does not seem to learn to handle rare words for low-resource parallel corpora. We used three tricks to deal with the problem.

1. *Simple NER*: Find words that start with a capital letter in both source and target sentences and replace them with UNK symbols. Recover them after inference.
2. *Remove rare words*: We replaced the words that appear less than 10 times in the parallel corpus with UNK tokens. We used *fast_align* unsupervised word alignment toolkit [1] to match UNK tokens in the source and target sentences.
3. *Concatenate word-by-word translation*: We use a dictionary based on the word alignment algorithm to do word-by-word translation, and then concatenate it to the source sentence before passing to Transformer (both at training time and during inference). Idea comes from [2]. This didn't really help.

Other experiments

- We quickly produced translations for English-Russian pair using Google Translate and Yandex Translate.
- We used *MOSES* toolkit as an SMT baseline. We used the same NLTK tokenization, learned a language model on both parallel and monolingual corpora, then trained translation model with -alignment grow-diag-final-and -reordering msd-bidirectional-fe flags.
- MOSES could beat all our NMT systems except the one with ensembling.
- We tried to learn POS tagging on Universal Dependencies data using Transformer. We reached 91.06% accuracy for Czech language (which has one of the largest annotated corpora). State-of-the-art is 98.83%. The decoder did not learn to consistently produce an output with the same length as the input.

Backtranslation?

We had unsuccessful attempts to use backtranslation to improve the performance.

- When we used backtranslated sentences from the first epochs, their quality was too low and the system never learned to translate better.
- We tried a "warmup" technique to learn Russian-English direction on the parallel corpus only and then add backtranslated sentences for the English-Russian direction. We couldn't handle the overfitting properly and never got better results than in a pure supervised setting.

References

[1] Chris Dyer, Victor Chahuneau, and Noah A. Smith.
A simple, fast, and effective reparameterization of ibm model 2.
In *HLT-NAACL*, 2013.

[2] Robert Östling, Yves Scherrer, Jörg Tiedemann, Gongbo Tang, and Tommi Nieminen.
The helsinki neural machine translation system.
In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*, pages 338–347, Copenhagen, Denmark, 2017. ACL.

[3] Rico Sennrich, Barry Haddow, and Alexandra Birch.
Neural machine translation of rare words with subword units.
In *Proceedings of the 54th Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 1715–1725. ACL, 2016.

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin.
Attention is all you need.
In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

Acknowledgements

We would like to thank the organizers of DeepHack.Babel for the amazing atmosphere and a great school program.

Contact Information

- http://yerevann.com/
- https://github.com/yerevann/
- Email: info@yerevann.com