Exploring the Limits of the Wake-Sleep Algorithm on a Low-Resource Language

Stanford CS224N Custom Project

Abisola Olawale
Department of Computer Science
Stanford University
aolawale@stanford.edu

Abstract

Machine Translation (MT) remains an open sub-problem in Natural Language Processing (NLP). Recently developed systems have achieved impressive, near-human performances on certain languages, but these systems are heavily dependent on large parallel corpora in the source and target languages. Thus, this approach is impractical for languages with sparse bodies of text; the real challenge lies in creating machine translation systems that are able to provide high quality, syntactically-correct translations with small amounts of parallel corpora. This work aims to utilize the wake-sleep back-translation algorithm introduced by Cotterell and Kreutzer [1] in order to generate synthetic training data that will then be used on OpenNMT's pre-built models in order to achieve high-quality Yoruba-English translations. We also explore the effectiveness of the wake-sleep algorithm when used in-conjunction with in-domain data (in domain meaning data of a similar–or very closely related–subject as the original training data) versus out-of-domain-data (data not necessarily related in topic to the training data). We examine the performance differences between a smaller, in-domain dataset and a larger, less topical out-of-domain dataset and compare those results to our baseline. We find that performing the wake sleep algorithm on a small, in-domain dataset leads to a decrease in BLEU score of about 5 points when compared to the baseline model (14.47). Training on an out-of-domain dataset leads to a 6 point decrease in the BLEU when compared to the baseline.

1 Introduction

Finding methods to tackle the data scarcity problem in the context of machine translation for low resource languages is an ongoing effort in the NLP community. Excellent results have been achieved with systems that use large amounts of parallel training data, where a system can learn word-mappings between the source and target languages. Unfortunately, large parallel corpora are relatively rare–even for high resource language pairs. A promising approach to this problem has been back-translation, where data asymmetry between the source and target language is exploited so that a large monolingual corpus is used to generate an additional, synthetic parallel corpus that is then fed back to the source-to-target model to be used as training data.

The wake-sleep algorithm from Cotterell and Kreutzer [1] is an extension of the backtranslation algorithm (one iteration of wake-sleep is backtranslation). At a high level, the wake-sleep algorithm works by training two NMT systems: one translates from the source language to the target language (in the forward direction, i.e. the forward model) while the other translates from the target language to the source language (i.e. the backward model). The backward model is then used to "hallucinate" translations for an additional monolingual corpus, additional parallel data for the forward model, leading to quality source-to-target translations. Then, the forward model "dreams" translations for the hallucinated translations, which is then fed back to the backward model. In this manner, both the backward and forward model are improved. This process is repeated until both models have converged. We implement the algorithm using a West-African, low-resource language (Yoruba).

We implement this algorithm by fine-tuning two language models for Yoruba (source) and English (target) trained on the King James Bible with two additional datasets. The first dataset, consisting of religious text (the Quran and the Book of Mormon) are considered our in-domain text. The second dataset (consisting of one million words of American English) is very dissimilar in syntax and structure to the training data, with a wide variety of topics. We find decreased BLEU scores for both the in-domain and out-domain data (9.69 and 8.44, respectively) when compared to the baseline model BLEU results (14.47).

2 Related Work

Duy et al. [2] introduce an iterative back-translation algorithm that is very similar to the wake-sleep algorithm. They also use forward and backward models to translate source language to target language and vice versa, and they use the backward model to generate synthetic data. They do not retrain the backward model as Cotterell and Kreutzer do. They find that they achieve tangible improvements in the BLEU scores English-German, English-French, and English-Farsi systems when the systems are allowed to converge (meaning their iterative back-translation is run for approximately 100K iterations). They also found that their results were heavily dependent on high-quaity monolingual and back-translated synthetic data.

Abdulmumin et al. [3] also approach this problem using a back-translation scheme. Instead of using the output of the backward model to train the forward model, they feed the backward model's out put back to itself in a self-training loop. The motivation behind this approach is to first maximize the quality of the translations of the backward model to ensure the synthetic data that will later be fed to the forward model is as high-quality as possible. They find that this approach leads to an improvement in the BLEUs of English-Veitnamese and English-German models by 1.5 and 11.06, respectively.

Similarly Zhang et al. [4] propose two approaches to NMT in low resource cases: 1) they experiment with the self-learning approach by building a baseline machine translation systemwith a parallel data and then optimize the backward model and 2)they attempt to predict target and reordered source sentences at the same time. With this approach, they train two systems, where one aims to predict a source to target sentence while the other system aims to select the correct ordering of the source sentence.

3 Approach

3.1 Wake Sleep Algorithm

The wake-sleep algorithm is implemented by training two Transformer models: one in the forward direction (Yoruba to English) we'll denote as F and one in the backward direction (English to Yoruba) which we'll denote as B. Each model was trained on the King James Version of the Bible–the forward model was trained on the Yoruba version while the backward model was trained on the English version. The backward model "hallucinates" source data by translating a monolingual corpus M in the target language (in this case, the monolingual corpus was in English) to produce B_{back}. B_{back} is then passed to the forward model, which "dreams" a translation of the synthetic data (we'll call this B_{dreamt}). This dreamt data is then passed to the backward model as training data for the backward model. The process is repeated for N iterations (Cotterell and Kretuzer report minimal improvements after three iterations, so we train for a maximum of three iterations). Note that when $N = 1$, we simply train F on B_back, which is the algorithm for back-translation. To save time with training, the forward and backward models were also fine-tuned on the in-domain and out-domain datasets instead of training on the Bible concatenated with the new data.

```
for i in 1... N:
  B_back = B.infer(M)
  F <- F.train(B_back)
  B_dreamt = F.infer(B_back)
  B <- B.train(B_dreamt)
```

4 Experiments

4.1 Data

The underlying baseline models are trained on the King James version of the Bible (approximately 30K sentences). A text file of the King James version of the bible in Yoruba was downloaded from the internet and was then converted into a CSV file. In the preprocessing step, PyGconverter [1] was used to remove verse line numbers and names from the start of each line to ensure the data consisted of only the actual Bible verses. The English version of the King James Bible was downloaded from Kaggle's Bible Corpus [2], with minimal preprocessing needed.

Kaggle's Religious Text [3] was used as the in-domain corpus when implementing the wake-sleep algorithm. This datasets consists of the Quran, the Book of Mormon, a book of meditations, The Gospel of Buddha, and the King James Bible. The Quran and the Book of Mormon were determined to be most similar to the Bible in terms of syntax and vocabulary content, so they were selected as the in-domain data. Together, they totalled approximately 58,000 sentences. The English corpus from Kaggle [4] was chosen as the out-of-domain corpus. This corpus consists of about 158,000 sentences.

4.2 Evaluation method

The model performance is evaluated by calculating the BLEU score in addition to human evaluation. While using the BLEU score is good because it provides an unbiased means of scoring the translation, there are numerous cases where the baseline provided surprisingly close English translations that oftentimes captured the spirit of the original text, so it is also important to discuss how well the model performed in less quantitative terms. For example, examine the following phrase (from the OpenNMT Transformer model):

Source (Yoruba): *O si lọ siwaju diẹ, o si dojubolẹ, o si gbadura pe, bi o ba le ṣe, ki wakati na le kọja kuro lori rẹ̀.*
Target (English): *And he went forward a little, and fell on the ground, and prayed that, if it were possible, the hour might pass from him.*
Model prediction: *And she went before a little space before her face to the earth, and said, if it be come to pass, that the time may pass over him.*
Human translation: *And he/she moved forward a little, and he/she put his/her face to the ground and prayed that, if it were possible, the time/hour may pass over him/her.*

While this sentence achieves a BLEU score of only 5.05, we see that the model prediction is not unreasonably far off from a human's translation. The model seems to do a pretty good job of doing a word-by-word translation of the input sentence, which aligns pretty well with a human's translation.

4.3 Experimental details

Two baseline models were created with OpenNMT. The first was an out-of-box default bidirectional Encoder-Decoder RNN model with a global attention mechanism OpenNMT's default hyperparameters. This was was trained for 5000 steps, with Adam used as the optimizer and a learning rate of 1. Initial tests with this model showed consistently poor results (it achieved a maximum BLEU of 2.43), so the second baseline was used for the rest of the experiments in order to save time and resources during training.

The second model architecture was a Transformer with six encoder and decoder layers and mulit-headed attention as described in the work Vaswani et al. [5]. The baseline backward and forward models used for translation were both models with this architecture. The forward model (Yoruba to English) was trained for 5000 steps (at this point we saw very good results in our translations) and the backward model (English to Yoruba) was trained for 10,000 steps. The backward model was trained

[1]https://github.com/gray-adeyi/pygconverter
[2]https://www.kaggle.com/oswinrh/bible/version/1
[3]https://www.kaggle.com/tentotheminus9/religious-and-philosophical-texts
[4]https://www.kaggle.com/espn56/english-corpus

for a longer period of time because of two reasons: 1) We did not see quality translations when sentences were randomly sampled and translated with the backward model trained at 5000 steps and 2) we wanted to ensure that we would obtain the highest-quality synthetic data when implementing the wake-sleep algorithm. Both models were trained using the Adam optimizer, with a learning rate of 1.

Using the in-domain and out-domain datasets, we implement the wake-sleep algorithm. We run a "baseline' version of the wake sleep algorithm for one iteration, which is essentially just vanilla back translation and then run it iteratively, training the backward model for a total of three iterations. As reported by Cotterell and Kreutzer [1], we see a decreasing positive effect when they experiment with the number of iterations. Since we finetune the preexisting forward and backward models instead of training from scratch, we ran each model for a maximum of 2000 steps on each iteration. This was done to save training time.

, so we anticipate this also generalizing to the number of sentences for Yoruba-English. While Edunov et al. [6] use millions of lines of text for the training data and show how performance changes as the synthetic data scales up, this project faces the constraint of finding enough English data in the same religious text domain. This constraint was imposed in order to generate as-noiseless-as-possible synthetic Yoruba data. We hypothesize that using smaller amounts of synthetic data might actually be beneficial to the goals of this project because i) we want to find the inflection point at which back-translation becomes detrimental, so using smaller increments of data helps us narrow this down with more detail and ii) we do not anticipate high-quality Yoruba synthetic data, so we expect to see a decrease in the positive effects of the synthetic data in the Yoruba-English model fairly quickly. Translating from English to Yoruba is a difficult task for even human speakers because of the existence of words that exist in English that do not exist in Yoruba (native speakers often just use the English word). The use of idioms, proverbs, and metaphors when speaking Yoruba also makes translating from English into Yoruba a tricky endeavor; a model could generate a perfect word-by-word translation from English to Yoruba, but it could be syntactically unintelligible in Yoruba. Coupled with the scarcity of Yoruba corpora, creating a high-quality English-Yoruba translation is difficult. Because of these reasons, we do not expect relatively small monolingual corpus size to pose a huge problem.

4.4 Results

	BLEU
Baseline 1: LSTM	0.2.43
Baseline 2: Transformer	14.44

	BLEU
Iteration 0: In domain	8.67
Iteration 1: In-domain	9.43
Iteration 2: In-domain	9.69

	BLEU
Iteration 0: Out domain	7.81
Iteration 1: Out-domain	8.02
Iteration 2: Out-domain	8.44

5 Future work

Next steps include running the experiments for back-translation–once second baseline has finished training. I also plan on making tweaks to baseline models by starting on pre-trained embeddings.

Facebook's fastText offers pretrained models for word embeddings in Yoruba and English, so feeding these in to the OpenNMT baselines might yield some improvements in performance.

References

[1] Ryan Cotterell and Julia Kreutzer. Explaining and Generalizing Back-Translation through Wake-Sleep.

[2] Cong Duy, Vu Hoang, and Trevor Cohn. Iterative Back-Translation for Neural Machine Translation. (2016):18–24, 2018.

[3] Idris Abdulmumin, Bashir Shehu Galadanci, and Abubakar Isa. Enhanced Back-Translation for Low Resource Neural Machine Translation Using Self-training. pages 355–371, 2021.

[4] Jiajun Zhang and Chengqing Zong. Exploiting source-side monolingual data in neural machine translation. *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 1535–1545, 2016.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017-December(Nips):5999–6009, 2017.

[6] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 489–500, 2020.