Data Augmentation for Languages with small datasets

Grishma Gupta [*1] **Lokit Kumar Paras** [*1] **Maulik Shah** [*1] **Abhash Kumar Singh** [*1]

Abstract

Finding annotated data for every target task in every target language is difficult, more so when the target language has only small datasets available. It has been observed that the performance of techniques like neural machine translation (NMT), phrase-based statistical machine translation (PBSMT), drops starkly in low-resource conditions, requiring large amounts of auxiliary data to achieve competitive results. For such scenarios, techniques like weak-supervision, data augmentation, transfer learning, etc. have shown promising results. In this project, we explore data augmentation techniques to improve Machine Translation for the Hindi language.

1. Introduction

The state-of-the-art deep learning models in many natural language processing (NLP) tasks require a large number of training examples. Deep learning architectures like BERT and its variants like RoBERTa and ALBERT have proven to be robust on several NLP tasks, but the datasets are prohibitively large. However, a lot of domains often lack sufficient data for training such deep models. Such domains are usually neglected or haven't received much attention in the past few years. However, in recent years, techniques like weak-supervision, data augmentation, active learning, etc. have shown promising results even in such scenarios.

One such domain is Machine Translation for languages that lack sufficient parallel corpora. Techniques like neural machine translation (NMT), phrase-based statistical machine translation (PBSMT), have been known to require large parallel datasets to show competitive results. Through this project, we explore the possibility of using data augmentation techniques while training NMT models, to improve the quality of machine translation even for a low resource language - Hindi.

[*]Equal contribution [1]University of Wisconsin - Madison.

1.1. Neural Machine Translation

Neural Machine Translation is the task of converting a sequence of words from a source language, like English, to a sequence of words in a target language like Hindi or Spanish using deep neural networks.

(Lample et al., 2018) show that using pre-trained cross-lingual word embeddings, to initialize the lookup table, can significantly improve the performance of an unsupervised machine translation model.

1.1.1. XLM

In (Lample & Conneau, 2019), the authors introduced a new multilingual language model, dubbed XLM, which leads to significant performance gains for a wide range of cross-lingual transfer tasks. It has been empirically seen that such a cross-lingual language model can be used to obtain:

- a better initialization of sentence encoders for zero-shot cross-lingual classification
- a better initialization of supervised and unsupervised neural machine translation systems
- language models for low-resource languages
- unsupervised cross-lingual word embeddings

Therefore, (Lample & Conneau, 2019) proposed pre-training the entire encoder and decoder with XLM model to bootstrap the iterative process of Unsupervised Neural Machine Translation (UNMT) (Lample et al., 2018).

In our project, we adopt a similar framework over UNMT, for evaluating the performance of the data augmentation techniques in machine translation for the Hindi language.

2. Related Work

Neural Machine Translation (NMT) (Bahdanau et al., 2016) is based on Sequence-to-Sequence encoder-decoder model along with an attention mechanism to enable better handling of longer sentences (Bahdanau et al., 2016). However, it requires immense amounts of parallel data in the order of tens of millions of sentences.

(Lee et al., 2017) and (Johnson et al., 2017) have shown that multilingual NMT is capable of performing machine translation for language pairs with small parallel dataset with the help of other languages. However, these techniques face challenges at lexical-level and sentence-level sharing between low-resource language and other languages. This restricts the languages for which this technique would be effective enough.

This is where the Unsupervised Neural Machine Translation (UNMT) (Lample et al., 2018) proves useful, as the model tries to learn translation even without presence of any parallel data.

(Şahin & Steedman, 2018) have proposed label preserving techniques borrowed from image processing : cropping and rotating, for augmenting low-resource European languages with rich case marking systems. (Wei & Zou, 2019) have proposed four simple but powerful operations: synonym replacement, random insertion, random swap, and random deletion for augmenting data in context of text classification. This approach showed strong results for smaller datasets on convolutional and recurrent neural network architectures. In this project, we apply the same techniques and test on tranformer based architectures.

3. Dataset

Hindi is one of the major languages in the Indian subcontinent. According to the 2001 Census of India, Hindi has 422 million native speakers and more than 500 million total speakers (Wikipedia, 2017). In contrast, English is spoken by just around 125 million people in India, of which a very small fraction are native speakers. Hence, there is a large requirement for digital communication in Hindi to interface with the rest of the world via English and immense potential for English-Hindi machine translation. However, the parallel corpus available in the public domain is quite limited. To try to solve this problem, we have considered the English-Hindi language pair for data augmentation. For this purpose, we took the **IIT Bombay English-Hindi Corpus** (Anoop Kunchukuttan, 2018) as our dataset.

The IIT Bombay English-Hindi corpus contains parallel corpus for English-Hindi as well as monolingual Hindi corpus collected from a variety of existing sources and corpora developed at the Center for Indian Language Technology, IIT Bombay.

The parallel corpus has been compiled from a variety of existing sources primarily OPUS (Tiedemann, 2012), HindEn (Bojar et al., 2014), TED (Abdelali et al., 2014), Judicial Domain Corpus, Mahashabdkosh, Indian Government corpora, Hindi-English linked wordnet, Gyaan-Nidhi corpus, as well as corpora, developed at the Center for Indian Language Technology, IIT Bombay over the years. The training corpus consists of sentences, phrases as well as dictionary entries, spanning many applications and domains. The details of the training corpus are shown in Table 1.

Source	**Segments**
GNOME (Opus)	145,706
KDE4 (Opus)	97,227
Tanzil (Opus)	187,080
Tatoeba (Opus)	4,698
OpenSubs2013 (Opus)	4,222
HindEnCorp	273,885
Hindi-English Wordnet Linkage	175,175
Mahashabdkosh: Dictionary	66,474
Mahashabdkosh: Examples	46,825
Mahashabdkosh: Definitions	46,523
TED talks	42,583
Indic multi-parallel corpus	10,349
Judicial domain corpus - I	5,007
Judicial domain corpus - II	3,727
Indian Government websites	123,360
Wiki Headlines	32,863
Book Translations (Gyaan-Nidhi Corpus)	227,123
Indian Government websites 2	69,013
Indian Government websites 3	47,842
Total	**1,609,682**

Table 1. Details of the IITB English-Hindi Parallel Corpus (training set)

For monolingual corpora, we use Wikipedia dump using WikiExtractor2 for both English and Hindi.

4. Data Augmentation

We employ 4 different text editing techniques for augmentation (Wei & Zou, 2019). For a given sentence in the training set, we randomly choose and perform one of the following operations:

1. **Synonym Replacement (SR)**: Choose α fraction of random words from the sentence that are not stop words. Replace each of these words with one of its synonyms chosen at random.
2. **Random Insertion (RI)**: Find a random synonym of a random word in the sentence that is not a stop word and insert that synonym into a random position in the sentence. Do this for α fraction of the total words.
3. **Random Swap (RS)**: Randomly choose two words in the sentence and swap their positions. Do this for α fraction of the total words.
4. **Random Delete (RD)**: Randomly choose n words in the sentence and delete them. Do this for α fraction of the total words.

For our experiments, we range α from 0 (no augmentation) to 0.5 (half the sentence is augmented).

Examples of the above operations are presented in Table 2.

Operation	Sentence
None	The dogs were playing in the park.
SR	The **puppies** were playing in the park.
RI	The dogs are playing in **garden** the park.
RS	The **playing** were **dogs** in the park.
RD	~~The~~ dogs ~~were~~ playing in the park.

Table 2. Sentences generated using EDA. SR: synonym replacement. RI: random insertion. RS: random swap. RD: random deletion.

4.1. Sentence generation

The IIT Bombay English-Hindi corpus contains 1.6M sentences in the training set. We randomly took subsets of 100k, 500k, and 1M sentences and applied each of the four editing techniques five times per sentence to generate 500k, 2.5M and 5M augmented sentences per technique.

4.2. Libraries

Princeton Wordnet was used for finding synonyms in English. For this purpose, we used the `wordnet` package from Python Natural Language Toolkit (NLTK). For finding synonyms in Hindi, we used the `pyiwn` (Panjwani et al., 2017) package in Python which uses Hindi Wordnet from CFILT, IIT Bombay in the background.

5. Neural Machine Translation

In this section, we briefly explain the flow of our machine translation model as shown in Figure 1 and how we used English-Hindi parallel dataset in it.

5.1. Tokenization

We use BPE tokenization (Radford, 2018) to process both English and Hindi sentences in the dataset. As shown in (Lample & Conneau, 2019), it is seen that using BPE reduces the size of input data and also improves the alignment of embedding.

(Lample & Conneau, 2019) shows that BPE randomly samples the corpora and increases the number of tokens for low resource languages. This also prevents the words to be split at each character level for Hindi. In our project, we processed the monolingual corpora and parallel corpora with BPE tokenization and created a shared vocabulary.

5.2. Pre-training language models

As previously mentioned, using pre-trained XLM models as the encoder and decoder can significantly improve the performance of Unsupervised Neural Machine Translation (UNMT). Therefore, taking the idea from (Lample & Conneau, 2019), we tried training XLM models using different objectives:

- Masked Language Modeling **(MLM)**
- Combined - Masked Language Modeling(**MLM**)) + Translation Language Modeling **(TLM)**

MLM objective tried in (Lample & Conneau, 2019) is the unsupervised model, which doesn't use the parallel corpus for training the model. On the other hand, the TLM objective uses the English-Hindi parallel corpus to train the language model. In this objective, the words are randomly masked in both English and Hindi. Therefore, to predict a masked word, the model looks for the word in both English and Hindi representation. This shows an improvement in embeddings and models tend to align for both English and Hindi representations. The model at each iteration takes a batch of sentences from both English and Hindi for training. In our project, we adapted this approach and trained on MLM objective which uses monolingual corpora, and MLM+TLM objective which uses parallel corpus along with monolingual corpora. The results from our experiment with each objective are summarized in Section 6.

5.3. Machine translation model

For learning the machine translation model, our setting is identical to the one of (Lample et al., 2018), except for the initialization step where we use these cross-lingual language models as encoder and decoder to pre-train the full model as opposed to using only the embeddings for the lookup table.

6. Experiment and Results

In this section, we first describe the experimental setup and preprocessing stage. Then, we describe the results from the trained language models and NMT model to highlight the impact of data augmentation.

6.1. Training details

To use the language models as encoders and decoders in NMT, we first pre-trained them using MLM and TLM objectives. For MLM we trained only on English and Hindi monolingual datasets, and for MLM+TLM we also used parallel en-hi data augmented corpus.

In this pre-training step, we used an embedding dimension of 512, a dropout rate of 0.1, and learned positional embed-

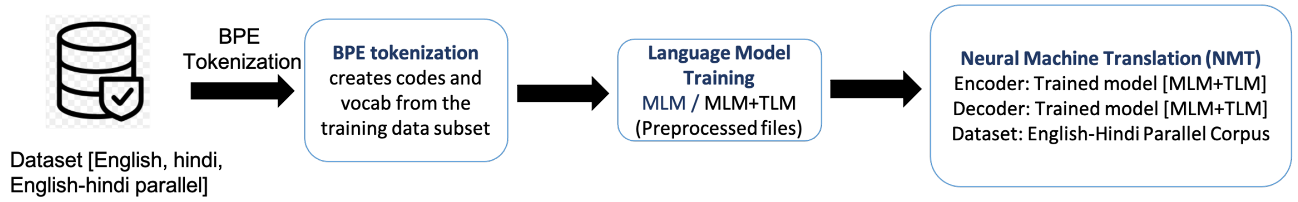
BPE Tokenization
Dataset [English, hindi, English-hindi parallel]
BPE tokenization creates codes and vocab from the training data subset
Language Model Training MLM / MLM+TLM (Preprocessed files)
Neural Machine Translation (NMT)
Encoder: Trained model [MLM+TLM]
Decoder: Trained model [MLM+TLM]
Dataset: English-Hindi Parallel Corpus

Figure 1. Neural Machine Translation framework

dings. We trained our model with Adam optimizer (Kingma & Ba, 2017), with a learning rate of 0.0001 and batch size 32. The metric average perplexity was used as the stopping criteria while training.

Once we had the language models trained, we now use them in the NMT model training process. Most of the training parameters remain the same as before: dropout rate of 0.1, embedding dimension of 512, adam optimizer with a learning rate of 0.0001, and batch size 32. Only this time, we use the BLEU score as the stopping criteria.

We conducted our experiments on 1M sentences for each language and randomly sampled data of 100K, 500k and 1M sentences from augmented parallel dataset.

All the models and code are implemented in PyTorch and trained on an NVIDIA GeForce GTX 1080 Ti GPU.

6.2. Data Preprocessing

We used monolingual data from Wikipedia dump using WikiExtractor[1] for monolingual corpora of English and Hindi. We use parallel data from the IITB corpus (Anoop Kunchukuttan (2018)) for English-Hindi data. We used Moses (Koehn et al. (2007)) tokenizer to tokenize both the languages and used fastBPE[2] to learn BPE codes and vocab. The words are split into subword units. The BPE codes are learned on the concatenation of sentences sampled from the combined training set.

6.3. Results and Analysis

In this section, we explain in detail how we compare the performance of XLM models in machine translation trained with MLM and TLM objectives on original and augmented data.

6.3.1. Pre-training models

We trained these model for encoder and decoder and evaluated it in two settings: a model that is trained on monolingual corpora only and one trained along with parallel corpus. Our monolingual corpus had 1M sentences from each language, and we used the augmented parallel dataset with upto 1M sentences randomly sampled.

We observed that the approach of using MLM+TLM performs better than MLM in terms of average perplexity and next word prediction accuracy. Therefore, for future evaluations on machine translation task, we focus more on the models trained with MLM + TLM objective.

6.3.2. NMT

To evaluate performance improvements due to augmentation, we use unsupervised neural machine translation model. For both the encoder and the decoder, we consider different possible initializations: MLM pretraining, MLM + TLM pretraining, MLM + TLM pretraining with augmentation. The results of BLEU score evaluation using these different pretrained models is given in Table 3

Table 3. Results for unsupervised NMT with a dataset of 100k sentences

Encoder-Decoder	En-hi	Hi-En
MLM	5.6	5.2
MLM+TLM	7.4	7.1
MLM+TLM (Delete)	7.8	7.3
MLM+TLM (Synonym)	7.5	7.2
MLM+TLM (Insert)	7.5	7.3
MLM+TLM (Swap)	7.7	7.4

As we see from Table 3, augmented data helps further improve the MLM + TLM model, as it can learn the embeddings in a better way. We especially found the Random Deletion and Random Swap to be more helpful than Synonym Replacement and Random Insert. This leveraging of parallel data for a low resource language like Hindi can be helpful and get us a significant boost in performance.

We can see in Figure 2 that the performance gain from using

[1]https://github.com/attardi/wikiextractor
[2]https://github.com/glample/fastBPE

data augmentation techniques diminishes as the dataset size increases. Moreover, as we increase the fraction of the sentence augmented (i.e. α), the performance gain decreases. We suspect that as more and more of the sentence gets augmented, it becomes less and less semantically similar to the original sentence, resulting in a decrease in quality of the gains.

7. Challenges

7.1. Small synset size for Hindi

We used the `pyiwn` (Panjwani et al., 2017) Python package for finding synonyms in the Hindi language which has just 40371 synsets for Hindi. Due to the smaller size of the Hindi synset, we observed that the augmentation operations like Random Insert generated the same sentence as input in absence of synonyms.

7.2. Reduced model capacity

For pre-training language models with higher accuracy, the models need to have a high capacity. In (Lample & Conneau, 2019), the authors suggested using an embedding dimension of 2048.

However, as part of our experiments, we overstepped GPU capacity even with an embedding dimension of 1024, as the number of model parameters becomes too large. Therefore, we switched to using an embedding dimension of 512.

7.3. Restrictions due to BPE Tokenization

The XLM + UNMT framework expects input data to be preprocessed using BPE tokenization. This prevented us from using language models pre-trained on much larger datasets for longer durations.

8. Conclusion

In this project, we tried to examine the impact of data augmentation techniques in improving the performance of machine translation for low-resource languages. The UNMT model when trained with cross-lingual models, by itself shows improved results for such low-resource domains.

However, from the experiments, we can see that with the help of augmentation technique of random swap or random delete, there is an improvement of $\sim 4.5\%$ in the average BLEU score. On the other hand, we only see an improvement of $\sim 1.3\%$ in case of Synonym replacement and Random insert operations. Overall, the performance gain is expected to diminish as we increase training set size, however, we still would see a similar behavior among different augmentation operations. Also, the BLEU scores show that addition of TLM to MLM led to a significant gain.

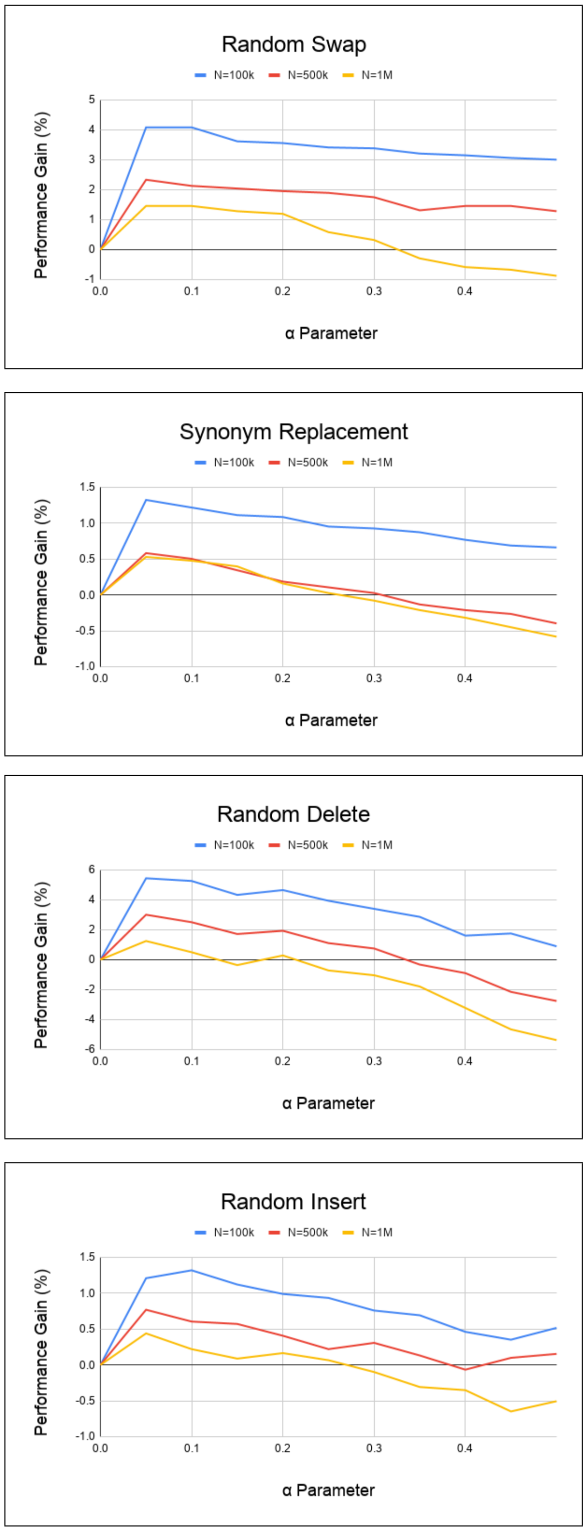

Figure 2. Performance gain(%) vs α for different augmentations.

We believe that if this experiment is carried out with more GPU memory, allowing a model with higher capacity, and trained for a longer time, we can get better language models and hence improve the average BLEU scores.

References

Abdelali, A., Guzman, F., Sajjad, H., and Vogel, S. The AMARA corpus: Building parallel language resources for the educational domain. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pp. 1856–1862, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/877_Paper.pdf.

Anoop Kunchukuttan, Pratik Mehta, P. B. The iit bombay english-hindi corpus. In Language Resources and Evaluation Conference, 2018.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate, 2016.

Bojar, O., Diatka, V., Rychlý, P., Straňák, P., Suchomel, V., Tamchyna, A., and Zeman, D. HindEnCorp - Hindi-English and Hindi-only corpus for machine translation. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pp. 3550–3555, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/835_Paper.pdf.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. Google's multilingual neural machine translation system: Enabling zero-shot translation, 2017.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. Moses: Open source toolkit for statistical machine translation. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions, pp. 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P07-2045.

Lample, G. and Conneau, A. Cross-lingual language model pretraining, 2019.

Lample, G., Ott, M., Conneau, A., Denoyer, L., and Ranzato, M. Phrase-based neural unsupervised machine translation, 2018.

Lee, J., Cho, K., and Hofmann, T. Fully character-level neural machine translation without explicit segmentation, 2017.

Panjwani, R., Kanojia, D., and Bhattacharyya, P. pyiwn : A python-based api to access indian language wordnets. 2017.

Radford, A. Improving language understanding by generative pre-training. 2018.

Şahin, G. G. and Steedman, M. Data augmentation via dependency tree morphing for low-resource languages. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 5004–5009, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1545. URL https://www.aclweb.org/anthology/D18-1545.

Tiedemann, J. Parallel data, tools and interfaces in opus. In In Language Resources and Evaluation Conference, 2012.

Wei, J. W. and Zou, K. EDA: easy data augmentation techniques for boosting performance on text classification tasks. CoRR, abs/1901.11196, 2019. URL http://arxiv.org/abs/1901.11196.