Offline Corpus Augmentation for English-Amharic Machine Translation

Yohannes Biadgligne, Kamel Smaïli

Offline Corpus Augmentation for English-Amharic Machine Translation*

Yohannes Biadgligne[1] and Kamel Smaïli[2]

[1] Bahir Dar Institute of Technology, Bahir Dar, Ethiopia
yohannesb2001@gmail.com
[2] Loria - University of Lorraine, Nancy, France
kamel.smaili@loria.fr

Abstract. The purpose of this study was to investigate the effect of corpus augmentation on the quality of English-Amharic Machine Translation (MT). In fact, tri-gram and four-gram Statistical Machine Translation (SMT) language models, as well as Neural Machine Translation (NMT) models based on Gated Recurrent Units (GRU) were used. They were trained independently using both the original and augmented corpus to see how the augmentation of the corpus affects the translation quality of these models. These two corpora (original and augmented) contain 225,304 and 463,796 English-Amharic parallel sentences respectively. To complete the corpus augmentation challenge, an offline token level tokenization technique was used. This technique (corpus augmentation) was used before any other MT processes were started. Among several token-level tokenization mechanisms, random insertion, replacement, deletion, and swapping approaches were chosen and implemented. After both models had been trained, the Bilingual Evaluation Understudy (BLEU) ratings were collected and analyzed. Our results demonstrate that the models trained with the augmented corpus outperform their corresponding models (models trained with the original corpus) in terms of BLEU scores. As a result, we can conclude that corpus augmentation did indeed help in the improvement of the performance of both SMT and NMT translation systems.

Keywords: Amharic language, Machine Translation, SMT, NMT, Corpus Augmentation, GRU, Token level augmentation.

1 Introduction

Machine Translation (MT) is one of the most data-intensive Machine Learning (ML) applications available. To train more robust models, it requires a large amount of data. The introduction of Deep Neural Networks (DNNs) has aggravated this problem [1]. Because these architectures may have millions or billions of parameters and necessitate a vast parallel training corpus, which is frequently unavailable for languages like Amharic [2] [3] [4]. Preparing such a large volume

* Supported by Bahir Dar Institute of Technology.

of material is time consuming and exhausting. One method for dealing with this problem is to use data augmentation techniques [5]. Data augmentation is the practice of synthesizing new data from the data at hand [6]. Despite the fact that data augmentation techniques have been used extensively in fields such as computer vision, there has recently been a surge in interest in Natural Language Processing (NLP) as a result of more work in low-resource domains, new tasks, and the popularity of large-scale neural networks that require large amounts of training data [7]. It is the most cost-effective technique to train deep learning architectures and higher N-gram SMT models, especially for languages with limited resources, such as Amharic [8].

1.1 Motivation

In our previous work [9], we were able to collect a relatively big English-Amharic parallel corpus. There are 225,304 English-Amharic parallel sentences in this corpus. SMT and NMT models were trained using this dataset. In the SMT experiment, we utilized KenLM to train a 3-gram language model, while in the NMT experiment, we used a Recurrent Neural Network (RNN) architecture with an attention mechanism. The SMT and NMT models had BLEU scores of 26.47 and 32.44, respectively. To see if other SMT and NMT models performed better, we trained a 4-gram SMT model and a Gated Recurrent Unit (GRU) based NMT architecture.

However, the performance was disappointing and did not meet our expectations. In fact, the BLUE score of the 4-gram SMT model plummets dramatically (from 26.47 to 14.26). That's a BLEU score drop of more than 85%. When compared to the RNN model, the GRU architecture achieves 29.30 BLEU, a 10.7% decrease in performance. When we looked into why these models performed poorly, we discovered that our corpus was insufficient to train a 4-gram SMT model or GRU-based architectures. As a result, we should expand our corpus. We had two choices for increasing the size of our original corpus:

- To collect new data from other sources and add it to the corpus that has already been collected.
- To make use of corpus augmentation methods [10] [11].

We chose the corpus augmentation method over the other option because of its feasibility and timeliness. As a result, the following are the motivations for this research work:

- To see how corpus augmentation influences the performance of English-Amharic MT.
- To improve the performance of our MT models and contribute to the long process of putting Amharic on the world's NLP map.

1.2 Data augmentation approaches for MT

Data augmentation is the process of creating artificial data by tweaking some parameters of the existing genuine data [12]. This technique has been widely

used in digital image processing and computer vision. It is used specifically in computer vision research to increase the size of data sets and, as a result, reduce over-fitting in training Deep Neural Network (DNN) architectures such as Convolutional Neural Networks (CNN) [5] [13] [14]. Data augmentation methods have recently been explored as a means of improving data efficiency in NLP [15]. It acts as a regularizer and helps reduce over-fitting when training a machine learning model. According to the literature, different corpus augmentation techniques are utilized. The following section will provide a concise and precise description of the most commonly used corpus augmentation techniques.

- **Online/Offline augmentation** these two techniques differ in the order in which the augmentation takes place in the pipeline of the training of a machine learning model. Is it before the training begins? (offline) or on the fly? (online). The offline method is better for small amounts of data, whereas the online method is better for large amounts of data. Furthermore, if we augment data using an offline approach, we can store the augmented data for later use, but this is not the case for online augmentation techniques. This means that once the training is completed, the data will be lost [16].
- **Token-level augmentation** manipulates words and phrases in a sentence to generate augmented text while retaining the semantics and labels of the original text as much as possible [15]. This technique can be used in a variety of ways. The following are the techniques that are most frequently used in the literature:
  - **Designed replacement** this technique augments a given text by changing certain words in a given sentence by their synonyms without changing the semantic and syntax of a given sentence. However, improvements from this technique are usually minimal and in some cases, performances may even degrade [17] [18] [19] [20].
  - **Random insertion, replacement, deletion and swapping** this approach manipulates a given sentence by inserting random tokens, replacing non-important tokens with random tokens, randomly swapping tokens, or randomly deleting certain tokens in a given sentence. A well-planned random local modification like this will preserve the semantics and syntax of the sentence [21] [22].
  - **Compositional augmentation** this method creates augmented sentences by recombining different fragments of different sentences. When compared to random swapping, this technique frequently necessitates more carefully designed rules. [23].
- **Sentence-level augmentation** rather than focusing on the tokens, this method considers the entire sentence at once. [15]. This technique includes:
  - **Paraphrasing** this method replaces a given sentence with its equivalent other sentence without changing its meaning. It attempts to preserve the essential meaning of the material being paraphrased. This task can be accomplished using round trip translation or back translation, which is a process of translating sentences into certain intermediate languages and then translating them back to generate paraphrases [24].

– **Adversarial data augmentation** this approach uses a Generative Adversarial Network (GAN) machine learning algorithm to generate fake data from given genuine data. Given a training corpus, this technique learns to generate new fake corpus that look at least superficially similar to human observers. We can find white box and black box methods under the adversarial technique [25].

1.3 Brief introduction of Amharic language

Amharic (አማርኛ/*ənərɨgnə*) is the official language of the Republic of Ethiopia. It is estimated that there are over 57 million users worldwide. It is written in its own script, which is known as Fidel (ፊደል/*fidəl*). Amharic is a morphologically complex language in terms of structure. Detailed information about this language can be found in [9].

2 Related works

Data augmentation is a common technique for training deep networks for image processing and computer vision in general [5] [13] [14]. But not long ago it is considered a cumbersome practice in training DNNs for NLP tasks such as MT. Nonetheless, some ice-breaker researchers have recently applied this method to the MT domain and obtained promising results. In this section, we will discuss briefly the most notable contribution that employs data augmentation for MT purposes.

The pioneering work of using data augmentation for MT is adopted by Fadaee et al. [19]. They employed token level augmentation technique to create their augmented corpus. Their goal was to provide novel contexts for rare words. To accomplish this, they looked for contexts where a common word could be replaced by a rare word, and then replaced the corresponding word in the other language with the translation of that rare word. The results of their experiments on simulated low-resource settings show that the method they used improves translation quality by up to 2.9 BLEU points over the baseline and up to 3.2 BLEU points over back-translation.

Additionally, Gao et al. [20] uses token level data augmentation technique for MT but they proposed to compute a weighted average over embedding's of possible words predicted by language models as the replaced input since the averaged representations could augment text with richer information. Experimental results on both small scale and large scale machine translation data sets demonstrate the superiority of their method over strong baselines.

By randomly swapping tokens in one sentence, Artetxe et al. [22] and Lample et al. [21] attempted to create an augmented corpus for French-to-English and German-to-English translation, and their translation model shows an improvement in translation quality. In fact, the BLEU score increased from 15.56 to 21.81 for French-to-English translation and from 10.21 to 15.24 for German-to-English translation.

Demi Guo et al. [23] proposed a straightforward data augmentation strategy to encourage compositional behavior in neural models for sequence-to-sequence problems. SeqMix is their method for creating new synthetic examples by softly combining input/output sequences from the training set. Their model SeqMix consistently outperforms strong transformer baselines by approximately 1.0 BLEU on five different translation data sets (German-English, English-German, English-Italian, English-Spanish).

Even though we refer and cite the above research works from the knowledge base; we are unable to locate any work on English-Amharic MT that employs corpus augmentation. So far, this is the first research contribution that employs the corpus augmentation technique for English-Amharic MT.

3 Experimental Set-up

3.1 Our corpus

Our original English-Amharic parallel corpus was collected using a combination of automatic and semi-automatic methods. There are 225,304 parallel sentences in this corpus. We recently added 6594 newly collected parallel sentences to our original corpus [9] bringing the total size of our corpus to 231,898 sentences.

Table 1: More information on our parallel corpus

Domain	**Number of sentences**
Religion	200,617
Law	14,515
News	16,766
Total	**231,898**

3.2 The techniques used for corpus augmentation

As described in the preceding sections, corpus augmentation is a new performance enhancing technique for MT applications. As a result, we are eager to test this technique on our collected parallel corpus and see how it performs. We chose the token level augmentation technique for this purpose, which is a random insertion, replacement, deletion, and swapping technique.

To perform the corpus augmentation, we created a python script that deletes words with a given probability, replaces words with a given probability, and swaps words up to a certain range. There are four parameters in this script: delete probability, replace probability, swapping range, and filter token. The delete probability and replace probability parameters use probability values ranging from 0 to 1. The swapping range parameter is determined by the length of the

sentence. On the other hand, we can use a constant string (in our case TAB) or an empty string ("") as a filter token. As a result, by changing the values of these parameters, we can generate a variety of augmented sentences with varying degrees of deletion, replacement, and swap. In fact, by varying the values of the aforementioned parameters, we were able to generate seven distinct augmented corpus from our original corpus. To choose a corpus that is sufficiently augmented while still preserving the meaning of the original corpus, the cosine similarity between these seven corpora and the original corpus was calculated (Table 2 shows the cosine similarity measure and the parameter values we used to create the augmented corpus). Finally, we chose an augmented corpus that is 90% (cosine similarity = 0.90138) similar to the original corpus. Because this corpus retains the original corpus' semantics even after some permutation, deletion, and replacement. Cosine similarity [29] is a metric used to compare the similarity of two words, sentences, or corpora. The output values close to one indicate a high degree of similarity. Cosine similarity is mathematically expressed by equation 1.

$$cos(\boldsymbol{o}, \boldsymbol{a}) = \frac{\boldsymbol{o} \cdot \boldsymbol{a}}{||\boldsymbol{o}|| \cdot ||\boldsymbol{a}||} \tag{1}$$

Where **o** and **a** stands for original and augmented corpus respectively.

Table 2: Corpus augmentation parameters and cosine similarity values

Augmented corpus	**Delete prob.**	**Replace prob.**	**Filter token**	**Swapping range**	**Cosine similarity**
1	0	0	""	4	0.9999
2	0	0	TAB	4	0.9889
3	0.1	0.1	TAB	4	0.9013
4	0.1	0.1	TAB	6	0.7693
5	0.2	0.2	TAB	4	0.7278
6	0.2	0.2	TAB	6	0.6595
7	1	1	TAB	Any value	0.0000

The corpus augmentation is performed in a similar manner (using the same parameter values) for both languages (English and Amharic). Once the augmentation was finished we merged the augmented corpus with the original corpus. This results in 463,796 parallel sentences (231,898 original sentences + 231,898 augmented sentences). The text boxes below display sample sentences from our original and augmented corpora for both English and Amharic.

Sample sentence from our original English corpus

Besides, in order to efficiently control and enforce strong punishment resolution upon airplane hijacking, including air brigandage and other different crimes, the presence of the law is mandatory.

Sample sentence from our augmented English corpus

Besides, order efficiently in to control and enforce strong including hijacking, resolution TAB brigandage air and different other crimes, the is presence of mandatory.

Sample sentence from our original Amharic corpus

ከዚህም ሌላ የአውሮፕላን ጠለፋን የአየር ላይ ጨንብድናን ጨምሮ የተለያዩ ወንጀሎችን በተቀላጠፈ ሁኔታ ለመቆጣጠርና ጠንካራ የቅጣት ዉሳኔ ለመስጠት የህጉ መኖር አስፈላጊ መሆኑን አመልክቷል

Sample sentence from our augmented Amharic corpus

ከዚህም ሌላ የአየር የአውሮፕላን ጠለፋን ላይ ጨምሮ ጨንብድናን ወንጀሎችን የተለያዩ በተቀላጠፈ ሁኔታ ለመቆጣጠርና የቅጣት ዉሳኔ ጠንካራ የህጉ መኖር አስፈላጊ መሆኑን

3.3 Preprocessing

Before feeding data (images, text, etc.) to machine learning models, it is recommended that the data be preprocessed to make it more suitable for the model [30]. In general, data preprocessing makes data more convenient to machine learning algorithms, allowing them to learn more effectively and efficiently. Preprocessing is subjective, which means it depends on the purpose of preprocessing, the type of data, the language in consideration, and the technique being used [31]. But it's mandatory to do it no matter what.

In our case, because we are dealing with text data for MT, we must ensure that the data is clean and unambiguous. So, before feeding the data to our MT models, we used the following preprocessing techniques.

– **Character normalization** Amharic has distinct characters with similar sound and meaning. These characters are also known as homophones. This means that in Amharic we can combine different homophone characters to form a single word. For example, the word **Sun** can be written in Amharic like **ጸሀይ**, **ጸሃይ**, **ጸሐይ**, **ፀሀይ**, **ፀሃይ**, **ፀሐይ**. They are all pronounced as **/səhəy/** and and the meaning is the same. As a result, such nature of the language would confuse machine learning algorithms. In this study, we chose the most frequently used characters and substituted them for other characters with a similar sound and meaning to reduce ambiguity (confusion) in our MT models. In addition, we converted Amharic numbers to their Arabic equivalents. This makes our corpus consistent in usage of numbers and alphabetic characters.

- **Tokenization** it is a fundamental task in text processing and NLP. It is the process of breaking down a sentence, phrase, or word into smaller units. In this study, we tokenized sentences, which means we divided the sentences into words. There are several advantages to doing so. To begin with, we can use this tokenized form to easily count the number of words and frequency of words in the corpus. Furthermore, which will be used as a stepping stone for subsequent preprocessing steps. This preprocessing task is done for both languages; English and Amharic.
- **True casing** it is the capitalization of the first character of each sentence's first word. This task has only been done for the English language. Because Amharic has no capital and small letters.
- **Cleaning** this is an essential preprocessing step for MT. Because, at this stage, we will avoid unnecessary and irrelevant parts that are incorporated into the corpus. It involves avoiding punctuation marks, removing extraneous symbols, and rejecting extremely long sentences. Following the removal of punctuation marks and unnecessary symbols, we proceeded to delete extremely long sentences. In this regard, sentences with more than 80 words were discarded. As a result, after cleaning our augmented parallel corpus, the number of parallel sentences was reduced from 463,796 to 460,691.

3.4 SMT experiment

We built SMT systems with Moses using our augmented corpus. In this experiment first we trained a 4-gram KenLM-based language model. This language model was created using Amharic as our target language. We used the entire 463,796 augmented Amharic sentences for this task. Next, the translation model was built using both the source and target languages (English and Amharic, respectively) as input. We divide the augmented and cleaned data (460,691 sentences) into three parts before we begin training our translation model. As a training set, we used 448,190 parallel sentences, while 10,000 and 2501 sentences were used for validation and testing, respectively. The training lasts about 5 hours, after which the BLEU scores are recorded.

3.5 NMT experiment

Various NMT architectures have recently been widely used for training and deploying MT systems. RNN was one of these NMT models that was frequently used in academia and industry. Inherently, RNNs suffer from exploding and vanishing gradients [32]. To resolve this problem Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) neural networks were introduced [33]. These two models are rivals in many machine learning application. However, because GRUs have fewer parameters (and thus require less training time) and produce results comparable to LSTMs, we chose GRU neural networks for this experiment [34].

Our NMT model architecture in comparison to LSTM, GRUs are relatively new architectures that are being used in many machine learning applications [35]. In this section, we will discuss the cell structure of GRU as well as the layered architecture of our GRU-based neural network model.

GRUs have two gates at the cell level: an update gate and a reset gate. Essentially, these are two vectors that determine what information should be sent to the output. They are special in that they can be trained to keep information from a long time ago without losing it due to time or to remove information that is irrelevant to the prediction [36]. Figure 1 depicts the cell structure of GRUs graphically.

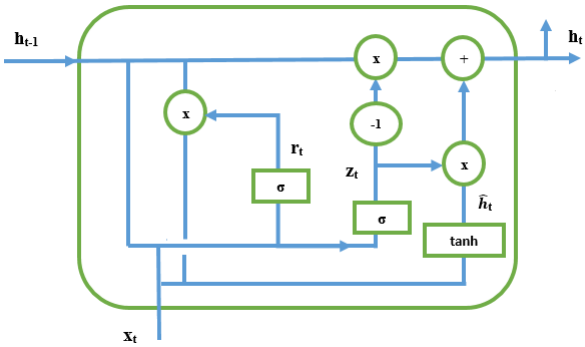

Fig. 1: Structure of GRU cell.

Where **z[t]**is an update gate; **r[t]** is a reset gate; $\hat{\mathbf{h}}$**[t]** is the activation function; **x[t]** is an input vector and **h[t]** is an output vector. Generally, the GRU is defined by the following equations:

The reset gate r[t] is computed as:

$$r[t] = \sigma\Big(\big[W_{\mathrm{r}}x\big]_{\mathrm{t}} + \big[U_{\mathrm{r}}h_{\langle \mathrm{t\text{-}1}\rangle}\big]_{\mathrm{t}} + b_{\mathrm{r}}\Big) \tag{2}$$

where σ is the logistic sigmoid function, and $\big[\ .\ \big]_{\mathrm{t}}$ denotes the t^{th} element of a vector. x and $\mathrm{h_{t\text{-}1}}$ are the input and the previous hidden state, respectively. $\mathrm{W_r}$ and $\mathrm{U_r}$ are weight matrices which are learned. Additionally, $\mathrm{b_r}$ is the bias added to the reset gate.

Consequently, the update gate uses the following mathematical equation to update it's weights.

$$z[t] = \sigma\Big(\big[W_{\mathrm{z}}x\big]_{\mathrm{t}} + \big[U_{\mathrm{z}}h_{\langle \mathrm{t\text{-}1}\rangle}\big]_{\mathrm{t}} + b_{\mathrm{z}}\Big) \tag{3}$$

The output vector h[t] is computed by equation 3.

$$h[t] = \Big(1 - z_{\mathrm{t}}\Big) \odot h_{\mathrm{t\text{-}1}} + z_{\mathrm{t}} \odot \hat{h}_{\mathrm{t}} \tag{4}$$

where

$$\hat{h}_{\mathrm{t}} = \phi_{\mathrm{h}}\Big(W_{\mathrm{h}} x_{\mathrm{t}} + U_{\mathrm{h}}\Big(r_{\mathrm{t}} \odot h_{\mathrm{t\text{-}1}}\Big) + b_{\mathrm{h}}\Big) \tag{5}$$

In order to conduct our GRU-based NMT experiment, we created three distinct GRU layers. They are an encoder layer, an attention layer, and a decoder layer. Each of the encoder and decoder layers has three GRU layers, with a hidden state size of 512. Before we began training the GRU-based NMT model, we divided our augmented corpus into three parts: training, validation, and testing sets, just as we did for our SMT experiment. These sets are preprocessed before being fed to the tokenization and embedding layers. The tokenization layers convert each word to a unique integer value, which is then converted to word embeddings by the embedding layer. The embedding layer has a dimension of 128. The architecture of our GRU based neural network model is depicted by Figure 2.

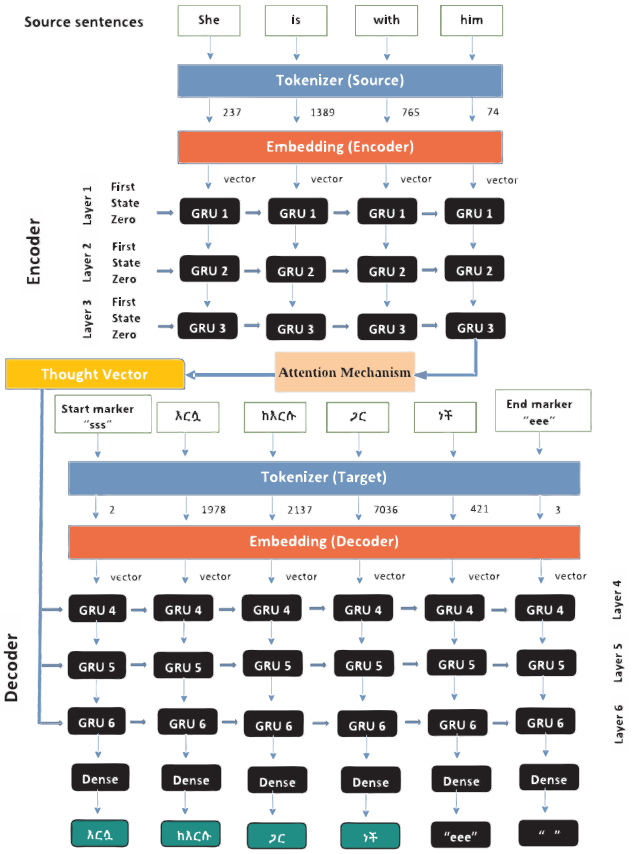

Fig. 2: GRU architecture.

The detailed training parameters are summarized in Table 2.

Table 3: Parameters and values of GRU model

Parameters	**Values**
Training Set	448,190
Validation Set	10,000
Test Set	2,501
Encoder Units	512
Attention mechanism	Bahdanau attention
Decoder Units	512
Embedding size	128
Loss function	cross entropy
Optimizer	RMSprop
Batch Size	512
Epochs	11
Evaluation Metric	BLEU

4 Experimental results

In this experiment a 4-gram SMT language model and a GRU-based NMT models were created and trained on our augmented English-Amharic parallel corpus. As stated in the motivation section of this paper, we tasted various SMT and NMT models before conducting this experiment. The results of those experiments are shown in Table 4.

Even though they were trained with the same corpus (original corpus), the BLEU score results showed a performance drop for both the 4 gram SMT and GRU based NMT experiments compared to RNN with attention and 3-gram SMT. To improve the performance of the two models (4 gram SMT and GRU-based NMT), a bigger corpus was needed. So, we used token level corpus augmentation technique to maximize the number of sentences in our original corpus. After the task of corpus augmentation was finished, we trained the 4 gram SMT and GRU based NMT models with the augmented corpus and their performances were increased by approximately 71% for the 4 gram SMT and 28.9% for the GRU based models compared to the 4 gram SMT and GRU based NMT models that were trained with the original corpus. Similarly, the GRU-based NMT model trained with the augmented corpus outperforms the RNN NMT model trained with the original corpus by 16.49%. These all results are summarized by Figure 3 and Table 4 respectively.

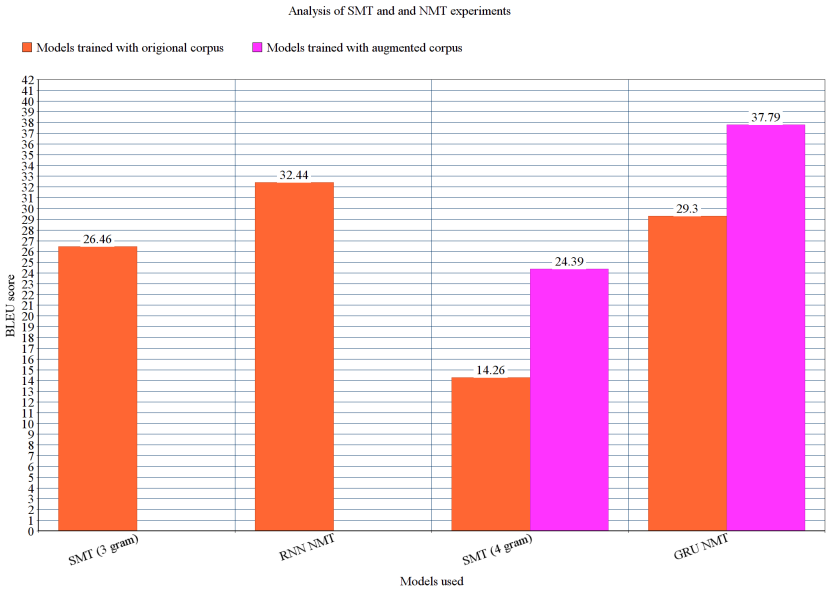

Fig. 3: Comparison of SMT and NMT models.

Table 4: Experimental results

corpus type	Model used	BLEU score
Original corpus	3-gram SMT	26.47
	RNN NMT	32.44
Original corpus	4-gram SMT	14.26
	GRU NMT	29.30
Augmented corpus	4-gram SMT	24.39
	GRU NMT	37.79

5 Conclusion

It is a fact that MT models requires a large parallel corpus. In particular, the introduction of DNN architectures to this domain has exacerbated the need for a huge corpus. This is also required when training higher gram SMT models. Unfortunately, low-resource languages like Amharic lack the resources needed to train robust NMT and SMT models. According to various studies, the majority of MT studies conducted for the Amharic language make use of a very small (small) corpus size resulting in poor translation quality.

To address this issue, we compiled a sizable English-Amharic corpus. This corpus (our original corpus) contains 225,304 parallel English-Amharic sentences. As far as we know, this was the largest corpus found for the language pairs. We

trained 3-gram SMT and RNN-based NMT models on this corpus and obtained 26.47 and 32.44 BLEU, respectively. Despite the fact that the BLEU score for the Amharic language was good in comparison to other similar works, it still needs to be improved in comparison to other highly resourced languages.

The purpose of this study was to look into the effect of corpus augmentation on English-Amharic MT, with the goal of improving the translation quality of SMT and NMT models for the language pairs. Using our collected corpus (original corpus), we trained a 4-gram SMT model and a GRU-based neural network model to achieve this goal. However, this experiment did not show any improvement; in fact, we observed a loss in performance on both models in comparison to our previous work. To improve the performance we trained the 4-gram SMT and GRU-based NMT using our augmented corpus. The augmented corpus contains a total of 463,796 English-Amharic parallel sentences. The corpus was passed through various preprocessing stages before the training of these models began. Following preprocessing, the size of our augmented corpus was reduced to 460,691 clean sentences. These cleaned parallel corpus was then divided into three parts. We used 448,190 parallel sentences for training, 10,000 parallel sentences for validation and 2501 parallel sentences for testing. After our Model had been trained and tested, the BLEU score was calculated. It should be noted that for both SMT and NMT models, we used training, validation, and testing sets of comparable size.

Finally, the BLEU scores of our MT models (both SMT and NMT) were improved. The registered BLEU scores for the 4-gram SMT and GRU-based NMT models were 24.39 and 37.79, respectively. From this We can conclude that corpus augmentation is the most cost effective and time saving approach to improve the performance of MT models especially for under-resourced languages like Amharic. At last, we would like to point out that, to the best of our knowledge, we have the best results ever for English to Amharic translation experiments using both 4-gram SMT and GUR-based NMT.

References

1. Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT. SCI. 2, 160 (2021). https://doi.org/10.1007/s42979-021-00592-x
2. Roh, Yuji, Geon Heo, and Steven Euijong Whang. "A survey on data collection for machine learning: a big data-ai integration perspective." IEEE Transactions on Knowledge and Data Engineering (2019).
3. Schmidt, J., Marques, M.R.G., Botti, S. et al. Recent advances and applications of machine learning in solid-state materials science. npj Comput Mater 5, 83 (2019). https://doi.org/10.1038/s41524-019-0221-0
4. Zador, A.M. A critique of pure learning and what artificial neural networks can learn from animal brains. Nat Commun 10, 3770 (2019). https://doi.org/10.1038/s41467-019-11786-6
5. Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. J Big Data 6, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0

6. Zhao, Amy, et al. "Data augmentation using learned transformations for one-shot medical image segmentation." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.
7. Feng, Steven Y., Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. "A survey of data augmentation approaches for nlp." arXiv preprint arXiv:2105.03075 (2021).
8. Gandhe, Ankur, Florian Metze, and Ian Lane. "Neural network language models for low resource languages." Fifteenth Annual Conference of the International Speech Communication Association. 2014.
9. Biadgline, Yohanens, and Kamel Smaïli. "Parallel Corpora Preparation for English-Amharic Machine Translation." International Work on Artificial Neural Networks, Conference Springer LNCS proceedings. 2021.
10. Praveen Kumar Badimala Giridhara and Chinmaya Mishra and Reddy Kumar Modam Venkataramana and Syed Saqib Bukhari and A. Dengel. "A Study of Various Text Augmentation Techniques for Relation Classification in Free Text." ICPRAM. 2019.
11. `https://amitness.com/2020/05/data-augmentation-for-nlp/`. last accessed August 11 2021.
12. Zemblys, Raimondas, Diederick C. Niehorster, and Kenneth Holmqvist. "gazeNet: End-to-end eye-movement event detection with deep neural networks." Behavior research methods 51.2 (2019): 840-864.
13. Cubuk, Ekin D., et al. "Autoaugment: Learning augmentation strategies from data." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019
14. Shorten, Connor, and Taghi M. Khoshgoftaar. "A survey on image data augmentation for deep learning." Journal of Big Data 6.1 (2019): 1-48.
15. Chen, Jiaao, et al. "An Empirical Survey of Data Augmentation for Limited Data Learning in NLP." arXiv preprint arXiv:2106.07499 (2021).
16. `https://www.analyticsvidhya.com/blog/2021/06/offline-data-augmentation-for-multiple-images/`. last accessed August 11 2021.
17. Kolomiyets, Oleksandr, and Marie-Francine Moens. "A survey on question answering technology from an information retrieval perspective." Information Sciences 181.24 (2011): 5412-5434.
18. Wei, Jason, and Kai Zou. "Eda: Easy data augmentation techniques for boosting performance on text classification tasks." arXiv preprint arXiv:1901.11196 (2019).
19. Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 567– 573, Vancouver, Canada. Association for Computational Linguistics.
20. Fei Gao, Jinhua Zhu, Lijun Wu, Yingce Xia, Tao Qin, Xueqi Cheng, Wengang Zhou, and Tie-Yan Liu. 2019. Soft contextual data augmentation for neural machine translation. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 5539–5544.
21. Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In International Conference on Learning Representations.
22. Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In International Conference on Learning Representations.

23. Demi Guo, Yoon Kim, and Alexander Rush. 2020. Sequence-level mixed sample data augmentation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5547–5552, Online. Association for Computational Linguistics.
24. Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. Advances in Neural Information Processing Systems, 33 .
25. Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. Proceedings of the 57th AnnualMeeting of the Association for Computational Linguistics
26. Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst, Moses: Open Source Toolkit for Statistical Machine Translation, Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic, June 2007.
27. Klein, Guillaume and Kim, Yoon and Deng, Yuntian and Senellart, Jean and Rush, Alexander. OpenNMT: Open-Source Toolkit for Neural Machine Translation". "Proceedings of ACL 2017, System Demonstrations". Jul, 2017. Vancouver, Canada. Association for Computational Linguistics. 67–72
28. Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." arXiv preprint arXiv:1409.0473 (2014).
29. Rahutomo, Faisal, Teruaki Kitasuka, and Masayoshi Aritsugi. "Semantic cosine similarity." The 7th International Student Conference on Advanced Science and Technology ICAST. Vol. 4. No. 1. 2012.
30. Somogyi, Zoltán, and Zoltán Somogyi. "Machine Learning Data." The Application of Artificial Intelligence: Step-by-Step Guide from Beginner to Expert (2021): 113-141.
31. Li, Canchen. "Preprocessing methods and pipelines of data mining: An overview." arXiv preprint arXiv:1906.08510 (2019).
32. Chang, Bo, et al. "AntisymmetricRNN: A dynamical system view on recurrent neural networks." arXiv preprint arXiv:1902.09689 (2019).
33. Dey, Rahul, and Fathi M. Salem. "Gate-variants of gated recurrent unit (GRU) neural networks." 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS). IEEE, 2017.
34. Baghaee, Tina. Automatic Neural Question Generation using Community-based Question Answering Systems. University of Lethbridge (Canada), 2018.
35. Cho, Kyunghyun; van Merrienboer, Bart; Gulcehre, Caglar; Bahdanau, Dzmitry; Bougares, Fethi; Schwenk, Holger; Bengio, Yoshua (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". arXiv:1406.1078.
36. https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be.