IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 10, No. 2, June 2021, pp. 306~315
ISSN: 2252-8938, DOI: 10.11591/ijai.v10.i2.pp306-315

Effective preprocessing based neural machine translation for English to Telugu cross-language information retrieval

B. N. V. Narasimha Raju[1], M. S. V. S. Bhadri Raju[2], K. V. V. Satyanarayana[3]
[1,3]Department of Computer Science and Engineering, Koneru Lakshamaiah Education Foundation (KLEF), Green fields, Vaddeswaram-522502, Guntur District, AP, India
[2]Department of Computer Science and Engineering, S R K R Engineering College, China amirma, Bhimavaram-534204, West Godavari District, A.P, India

Article Info

Article history:

Received Jul 31, 2020
Revised Feb 20, 2021
Accepted Mar 20, 2021

Keywords:

Cross-language IR
Long short-term memory
Machine translation
Neural machine translation
Preprocessing

ABSTRACT

In cross-language information retrieval (CLIR), the neural machine translation (NMT) plays a vital role. CLIR retrieves the information written in a language which is different from the user's query language. In CLIR, the main concern is to translate the user query from the source language to the target language. NMT is useful for translating the data from one language to another. NMT has better accuracy for different languages like English to German and so-on. In this paper, NMT has applied for translating English to Indian languages, especially for Telugu. Besides NMT, an effort is also made to improve accuracy by applying effective preprocessing mechanism. The role of effective preprocessing in improving accuracy will be less but countable. Machine translation (MT) is a data-driven approach where parallel corpus will act as input in MT. NMT requires a massive amount of parallel corpus for performing the translation. Building an English - Telugu parallel corpus is costly because they are resource-poor languages. Different mechanisms are available for preparing the parallel corpus. The major issue in preparing parallel corpus is data replication that is handled during preprocessing. The other issue in machine translation is the out-of-vocabulary (OOV) problem. Earlier dictionaries are used to handle OOV problems. To overcome this problem the rare words are segmented into sequences of subwords during preprocessing. The parameters like accuracy, perplexity, cross-entropy and BLEU scores shows better translation quality for NMT with effective preprocessing.

Corresponding Author:

B. N. V. Narasimha Raju
Department of Computer Science and Engineering
Koneru Lakshamaiah Education Foundation (KLEF)
Green fields, Vaddeswaram-522502, Guntur District, AP, India
Email: buddaraju.narasimharaju@gmail.com

1. INTRODUCTION

CLIR is a subfield of information retrieval (IR). In IR, we retrieve the relevant data by using the user query. CLIR is used to retrieve the data in another language than that of the user's query language. In this, we need to translate the user query language to another language. Machine translation (MT) is useful for the translation of data from one language to another. The need for MT has increased due to the rise of users in native languages. In 1990s, 80% of the web content was in English. In 2011 it had fallen to 27%. This is because of the rise in the content of other languages like Russian, French, German, and so-on. The KPMG analysis in India during 2017 stated that Indian language internet users are 38% in 2011 which was raised to

57% in 2016 and expected to be 72% by 2021. This is a clear sign of an increase in the importance of native languages. There is a need for appropriate MT mechanism for translating from one language to another, especially in the Indian languages. This helps the users to go through the content that was present in other than the native language.

Machine translation consists of different kinds of translation. Earlier direct translation has to translate the sentence from the source language into a target language. This technique uses a bilingual dictionary for translation. Now corpus-based translation is used, which is categorised into statistical and neural machine translation. Statistical machine translation (SMT) will depend on both bilingual corpus and statistical models. In NMT, the neural networks will perform the translations. Earlier SMT was used, but NMT has shown improvement in the accuracy of the translation. MT is a data-driven approach and depends on the corpus [1]-[3]. Without an adequate amount of corpus, it cannot achieve better translations. NMT uses RNN architecture [4] and mainly depends on the parallel corpus. So, there is a need to collect more parallel corpus for better translations.

NMT consists of three phases, viz. preprocessing, encoding and decoding. Preprocessing is performed on the parallel corpus. Obtaining the parallel corpus for Telugu-English language is difficult because they are resource-poor. Collection of the parallel corpus is either done manually or by using tools. The parallel corpus may contain different noises and inconsistencies. These kinds of problems will be more in morphologically rich languages like Telugu. The parallel corpus may contain data replication like the same source and different translations and vice-versa. This would confuse the machine while performing translations.

NMT shows good results, but the translation faces problems like out-of-vocabulary [5], [6]. If NMT uses a limited-size vocabulary with highest frequency words, then it leads to OOV problems. This result in poor translations [7]-[9]. If the source sentence contains more frequent words, then the translation will be good. If the source sentence contains more unknown words, then the translation will be poor. This kind of problems will be more in both resource-poor and morphologically rich languages. Earlier, OOV problems are using word-level NMT along with back-off dictionaries [10]-[12]. These models are not suitable for unknown words. So, they copy the unknown words into the target text. It would be apt for the named entities but not for all the unknown words.

The preprocessing phase will remove all the noises, data replication and OOV problems in the data. Then the encoding and decoding phases of the NMT will use the data. Preprocessing will solve different kind of problems which improves the quality of translation. So, preprocessing is an essential step in NMT. The next phases in NMT are encoding and decoding. NMT consists of two neural networks [13]-[16], i.e. encoder and decoder. The source sentence is the input for the encoder, and the decoder will generate the translation for the source sentence. The encoder-decoder phase can also handle the problem in processing long sentences.

2. RELATED WORK

Earlier SMT techniques are essential for translating the sentences. B.N.V Narasimha Raju *et al*. proposed [17] SMT for translation which consists of language model, translation model and decoder. In this, phrase-based translation model has achieved adequate quality of translation. Later neural networks along with MT has improved the translation when compared to the SMT. Mai Oudah *et al*. proposed [13] the combination of NMT and SMT for English-Arabic languages. In this, NMT has shown good performance but suffers from the translation of short sentences. This was resolved by using both statistical and neural MT, but it lacks good tokenization schemes. These tokenization problems are handled during preprocessing.

Preprocessing is one of the main steps in NMT. If preprocessing is not applied to the corpus, then the quality of the translation will be less. Preprocessing can perform tokenizations, remove unimportant words, and so-on. Preforming these techniques will depend on the situation. Duygu Ataman *et al*. proposed [18] using a fixed-sized vocabulary. The conventional methods will cause semantic and syntactic losses. These problems are solved by unsupervised morphology learning which reduces vocabulary. Anoop Kunchukuttan *et al*. proposed [19] the text normalization on Hindi-English parallel corpus and perform tokenization by using Moses toolkit.

Kyunghyun Cho *et al*. proposed [20] RNN encoder-decoder along with a gated recursive convolutional neural network. Here encoder is for extracting representations of fixed length from the input of variable length. From the representation, the decoder will generate the translations. This mechanism has shown good results for short sentences if it has less number of unknown words. In this way, NMT is showing a good performance when compared to the other techniques. The performance of NMT will decrease if there are more number of unknown words. So, these unknown words are also a kind of OOV problems.

Preprocessing can handle data replication problems in the English to Telugu Translation. The data replication will confuse the model while making the translations, which degrades the performance. So, remove the replicated data from the parallel corpus. To overcome the problem of OOV, we have used a tokenization scheme called byte-pair encoding (BPE) during preprocessing. BPE was generally used for data compression [21]. These are a kind of subword models which achieve better accuracy for the translation. It is also possible to translate some words that are not present at training time. Mattia A. Di Gangi *et al*. has proposed [22] byte-pair encoding in preprocessing for English-German languages.

In NMT, both the encoder and decoder are one of the phases in MT. In general, the regular RNN will be used for encoder and decoder phases. The regular RNN will have a problem when handling the longer-range dependencies in the source sentence. The proposed model uses long short-term memory (LSTM) instead of regular RNN in encoder and decoder. LSTM will produce better accuracy than regular RNN in case of long-range dependencies. NMT needs to use LSTMs in both encoder and decoder phases. In the preprocessing phase, it handles both OOV and data replication problems.

3. NEURAL MACHINE TRANSLATION

NMT consists of three phases, i.e. preprocessing, encoding and decoding, as shown in Figure 1. NMT is a data-driven approach. So, it depends on the parallel corpus. NMT requires large parallel corpus for generating better translations. The collection of the parallel corpus is also a problem for the resource-poor languages like Telugu-English. The creation of parallel corpus can be either manual or by using tools. Usage of tools will generate noises. Handling these noises are difficult. So, manual preparation of the corpus would be better but still there exist noise and inconsistencies in the data. If the Parallel corpus is inconsistent or noisy, then it will reduce the accuracy in translation. The preprocessing phase will handle both noises and inconsistencies in a parallel corpus. NMT also faces a problem with data replication and OOV. Data replication in the parallel corpus will consist of the same source sentences with different translations and vice-versa. NMT will handle both the data replication and OOV problems while preprocessing.

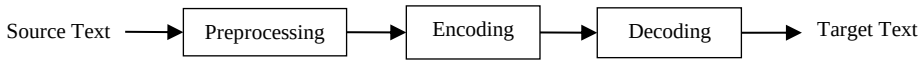

Figure 1. Steps in neural machine translation

In preprocessing, the source text is the input. It removes unwanted characters, symbols and so-on from a parallel corpus. Then remove the replicated data in a parallel corpus. First, we load the parallel corpus and convert the characters to lowercase. Now, remove the unwanted symbols in the parallel corpus. After removing them save the filtered corpus. Now remove the replicated data from the filtered corpus. For this first, we have combined the parallel corpus into a single file by maintaining a delimiter. Identify and remove the replicated sentences from Telugu-English parallel corpus.

In general, the parallel corpus would contain the data of both Telugu and English in two files. Consider a sample parallel corpus as,

This is ravi house	ఇది రవి ఇల్లు
Ravi gone to school	రవి బడికి వెళ్ళాడు
This is ravi house	ఇది రవి గృహము

Parallel corpus consists of data replication. Now convert English corpus to lowercase. Now parallel corpus will be

this is ravi house	ఇది రవి ఇల్లు
ravi gone to school	రవి బడికి వెళ్ళాడు
this is ravi house	ఇది రవి గృహము

remove the unwanted symbols and data replication in the parallel corpus. Parallel corpus consists of data replication in sentences 1 and 3. It has the same source sentence but different translations. To remove the

replication in data, combine parallel corpus into a single file by using a delimiter. The delimiter is useful for separating both English-Telugu parallel corpus. The delimiter used is */*. Now parallel corpus will be,

this is ravi house*/*ఇది రవి ఇల్లు

ravi gone to school*/*రవి బడికి వెళ్ళాడు

this is ravi house*/*ఇది రవి గృహము

Convert the complete corpus into the Unicode format. Identify whether there is any data replication. The corpus present before the delimiter is English and after the delimiter is Telugu. Identify the replicated sentences in the English or Telugu language corpus. Now, remove the complete sentence i.e. both English and corresponding Telugu language sentence. Line 1 and 3 consists of replicated data. After removing the data replication, the corpus will be,

this is ravi house*/*ఇది రవి ఇల్లు

ravi gone to school*/*రవి బడికి వెళ్ళాడు

Remove the data replication in a parallel corpus. Now, pass the corpus for further preprocessing. A parallel corpus of 20000 lines was passed for testing data replication. It has reduced the corpus to 17589. Parallel corpus has 2411 replicated sentences. Removal of data replication would raise the performance of NMT. NMT requires Telugu-English parallel corpus for training. The English-Telugu parallel corpus is a resource-poor language. Due to low resource, the vocabulary of the corpus may contain high-frequency words which cause the OOV problem [23]-[25]. If the input of NMT consists of unknown words, then it will reduce performance. To remove this OOV problem, word segmentation techniques are used. Divide the unknown word into subword units by using BPE, then try to translate by using subwords.

Byte pair encoding [12], [21] is a data compression mechanism, that is used for merging frequent pairs of bytes. This technique is also useful for word segmentation. Merge the characters or character sequences. In BPE, initialize the symbol vocabulary with character vocabulary. Represent each word as a character sequence with a special delimiter at the end. The delimiter is useful after translation to restore the original token. Count all the symbol pairs and replace the most frequent symbol pair with a new symbol which represents an n-gram character. Merge the frequent n-gram characters to form a single symbol. In BPE, the initial vocabulary size and the final symbol vocabulary sizes are equal. The BPE Algorithm in Figure 2 is useful for this kind of word segmentation. Apply BPE for both the source and target vocabulary. It is compact in text or vocabulary size. That means having a guarantee that the subword unit is present in the respective language training text.

```
Algorithm: Byte-pair encoding
Input: It contains a set of strings S and the target vocab size is k
procedure BPE (S, k)
        X is the all unique characters in S
        while |X| < k do
                tm, tn is the Most frequent bigram in S
                tl is tm + tn
                X is X + [tl]
                Replace each occurrence of tm, tn in S with tl
        end while
        return X
end procedure
```

Figure 2. Algorithm for byte pair encoding

In corpus, BPE will count the frequency of each word. Split word into characters and place a special token called </w> at the end of the word. For example, consider the word “high” and the tokens for the word are [“h”, “i”, “g”, “h”, “</w>”]. Count the frequency of all words in the corpus. It will specify the vocabulary for tokenized word along with its corresponding counts. For example, consider

{'h i g h </w>': 4, 'h i g h e r </w>': 2, 'n e w e s t </w>': 6, 'w i d e s t </w>': 3}

In every iteration, find the most frequent consecutive byte pair and merge the two byte-pair tokens into one. In the first iteration, the byte pair “e” and “s” occurred 6+3=9 times. Merge them into new token “es”. Now the vocabulary is

{'h i g h </w>': 4, 'h i g h e r</w>': 2, 'n e w es t </w>': 6, 'w i d es t </w>': 3}

In the next iteration, byte pair “es” and “t” occurred 6+3=9 times. Merge them into new token “est”. Now the vocabulary is

{'h i g h </w>': 4, 'h i g h e r</w>': 2, 'n e w est </w>': 6, 'w i d est </w>': 3}

In the next iteration, byte pair “est” and “</w> are more frequent. Merge the byte pairs into a new token “est</w>”. Repeat this until the defined subword vocabulary size or the next highest frequency pair is 1. Suppose if a word “highest” need to encode then the BPE would split it into two subwords viz. “high” and “est</w>”. By using the subwords, NMT would try to translate them. In the same way, any unknown words are converted to subword units and try to translate them. Apply the same process to the parallel corpus. Translating the unknown words will increase the performance of NMT.

After preprocessing, input will be passed to the encoder of the NMT. In general, the end-to-end approach is used for sequence learning. The sequence to sequence model will generate a fixed-length output from fixed-length input. Here the length of input and output may vary. The encoder consists of a multilayer LSTM, and it will map the input sequence to a fixed dimensionality vector. The decoder is also an LSTM that will generate the target sequence from the dimensionality vector [26], [27]. This architecture is shown in Figure 3. This mechanism will read the input sentences completely, and then it will start generating the output. The model will stop generating the output when it encounters a <eos> token.

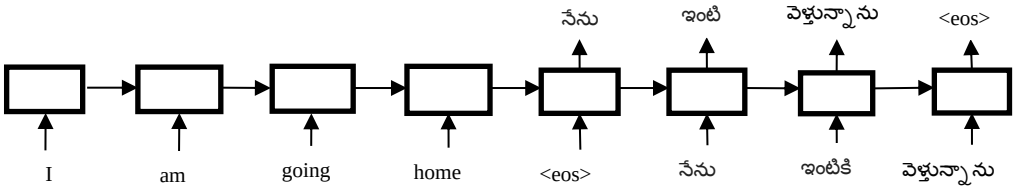

Figure 3. Sequence to sequence model

In the NMT the input sequence is (i_1,...,i_T) and the RNN generates an output sequence of (j_1,…,j_T) by iteratively performing the following equations.

$$h_t = f(W^{hi} i_t + W^{hh} h_{t-1})$$

$$j_t = W^{jh} h_t$$

Here h represents the hidden layer, f represents the activation function, W represents the weights. It is little bit difficult to apply RNN when input and output sequences are of different length. A general sequence model can map the input sequence to a fixed-size vector using one RNN. Now generate the target sequence by using another RNN from this vector. The problem arises in regular RNN when we need to train with long-range dependencies. In such situations, the regular RNN may fail to produce accurate translations. To overcome such problems, we can use LSTMs. LSTMs can handle the long-range dependencies in a better way than the regular RNN. The architecture of LSTM is shown in Figure 4.

The behaviour of LSTM is to hold the information for a longer time. In LSTMs, there are four layers which interact in a special way. LSTM consists of a cell state which is a horizontal line from C_{t-1} to C_t. It runs through the entire chain with some minor linear interactions. LSTM can add or remove the information in the cell state by using gates. It consists of a sigmoid layer and pointwise multiplication. The output of the sigmoid function is f_t.

$$f_t = \sigma(W_f . [h_{t-1}, i_t] + b_f$$

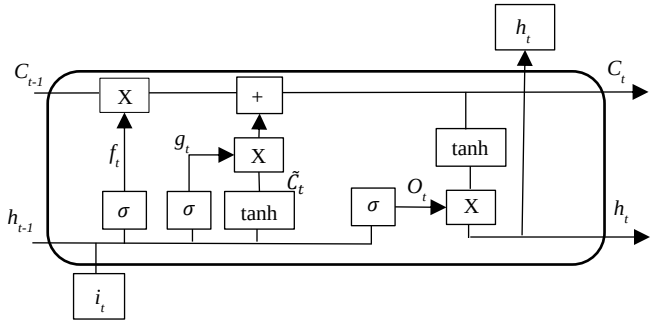

Figure 4. LSTM cell

The next step will decide what is the new information we are going to add to the cell state. The output of the sigmoid layer g_t. Next, the *tanh* will create a new candidate value $\tilde{C}_t$. Now we will update the cell state from C_{t-1} to C_t. Multiply the old cell state by f_t and then add the $g_t * \tilde{C}_t$.

$$g_t = \sigma(W_g . [h_{t-1}, i_t] + b_g)$$

$$\tilde{C}_t = tanh(W_c . [h_{t-1}, i_t] + b_c)$$

$$C_t = f_t * C_{t-1} + g_t * \tilde{C}_t$$

Finally, the outputwill be a filtered version of the cell state. The output of the sigmoid layer is o_t.

$$o_t = \sigma(W_o . [h_{t-1}, i_t] + b_o)$$

$$h_t = o_t * tanh(C_t)$$

In this way, each LSTM cell would function. The goal of the LSTM is to find the conditional probability $p(j_1,\ldots,j_{T'} \mid i_1,\ldots,i_T)$ where the length of both input sequence and output sequence may vary. The length of the input sequence is T, and the length of the output sequence is T'. It computes the conditional probability by obtaining the fixed dimensional representation v of the input sequence $(i_1,\ldots,i_T)$ and it is given by the last hidden state of the LSTM. Now compute the probability of $j_1, \ldots, j_{T'}$ with a standard LSTM-LM formulation whose initial hidden state is the representation v of $i_1,\ldots,i_T$.

$$p(j_1,\ldots,j_{T'} \mid i_1,\ldots,i_T) = \prod_{t=1}^{T'} P(j_t \mid v, j_1,\ldots,j_{t-1})$$

After having a rigorous training by using many sentence pairs, now the decoder will produce the correct translation T of the source sentence S by using LSTM.

$$\hat{T} = \underset{T}{\operatorname{argmax}}\, P(T \mid S)$$

The parameters used for the evaluation of the model are accuracy, perplexity, cross-entropy, and bilingual evaluation understudy (BLEU) score. Accuracy represents the amount of correct classification. The model having high accuracy is a better performer. The perplexity is a measure used for finding how well a probability model predicts a sample. The low perplexity score indicates good probability distribution for predicting the sample. The cross-entropy is useful for calculating the loss function. It measures the difference between two probability distributions for a given set of events. The model having less cross-entropy score will be a better performer. The BLEU score is useful for evaluating the predictions made by the machine translation systems. The model having higher BLEU score will perform well in predicting the translations.

4. RESULTS AND DISCUSSION

In this, the parallel corpus with and without replicated data is used. The replicated data will contain repeated sentences, sentences with the same source but different translations and vice-versa. Remove all the

repeated sentences from the dataset because it is difficult to find which source is correct for the translated sentence and vice-versa. If a sentence is repeated more number of times in the database, it will confuse the model in identifying and learning new features. It causes overfitting and will generate wrong results. If the train and test datasets also contain the same sentences, the accuracy will be more during training and testing. It will fail while translating the new sentences to produce the correct translations. Remove these problems from the dataset that helps in increasing the efficiency of the translation.

Apply normal preprocessing to generate word vocabularies, sequences of indices and BPE for data with and without replication. The performance of NMT using BPE for data with replication and without replication is as shown in Figure 5. The training accuracy for databases is shown in Figure 5(a), the training perplexity for databases is shown in Figure 5(b), the training cross-entropy for databases is shown in Figure 5(c), the validation accuracy for databases is shown in Figure 5(d), and the validation perplexity for databases is shown in Figure 5(e). Input the data for NMT. Here both the regular RNN and LSTM are used. NMT consists of two encoder and decoder layers. The size of the RNN is 500. It uses Adam optimizer, with a learning rate of 0.01. The decay is 0.5. The dropout is 0.3, and the training steps are 20000.

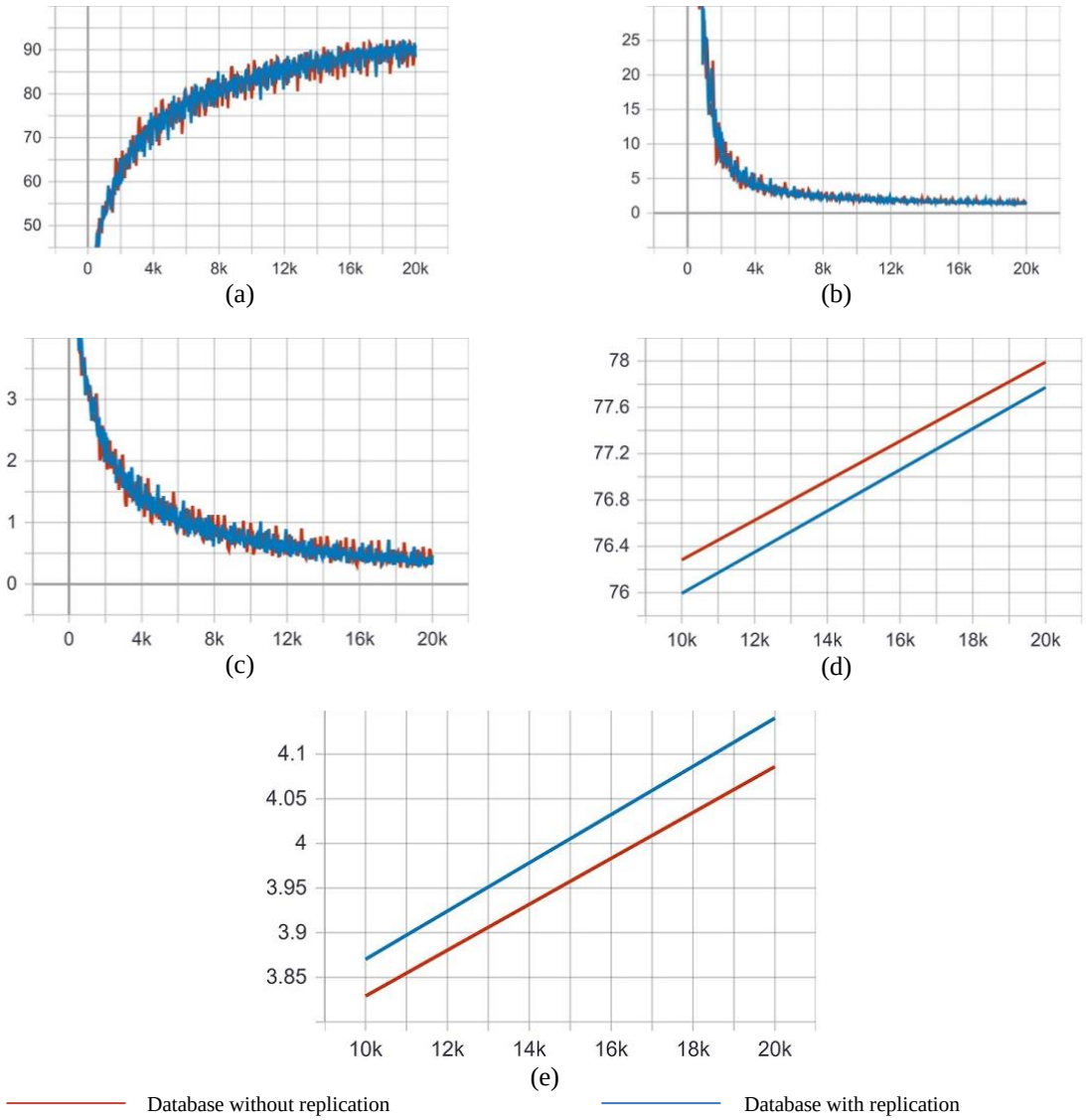

Figure 5. Performance of NMT using BPE for data with replication and without replication. (a) training accuracy, (b) training perplexity, (c) training cross-entropy, (d) validation accuracy, (e) validation perplexity

Now we compared the performance of NMT using different preprocessing techniques like data with and without replication along with BPE. The comparison of parameters like training accuracy, training perplexity, training cross-entropy, validation accuracy, validation perplexity are shown in Table 1. In all parameters, the database without replication has achieved better performance.

Table 1. Comparison of parameters for evaluating the performance of database with and without replication

Parameters	Database without replication	Database with replication
Training Accuracy	91.02	88.32
Training Perplexity	1.394	1.598
Training Cross-Entropy	0.332	0.4686
Validation accuracy	77.99	77.77
Validation Perplexity	4.086	4.14

We have also measured the performance of NMT by using the BLEU metric, and the scores are as shown in Table 2. The performance comparison for different NMT techniques using replicated and non-replicated corpus are as shown in Figure 6. NMT without replication using BPE and LSTM has more accuracy in translation. The preprocessing model used for removing the replication along with BPE is useful for improving the accuracy of NMT.

Table 2. Performance in various techniques for corpus with and without replication using BLEU score

Model	BLEU Score	
	Without replication	With replication
RNN	46.87	46.03
LSTM	47.19	46.38
BPE+RNN	47.70	47.01
BPE+LSTM	48.81	48.27

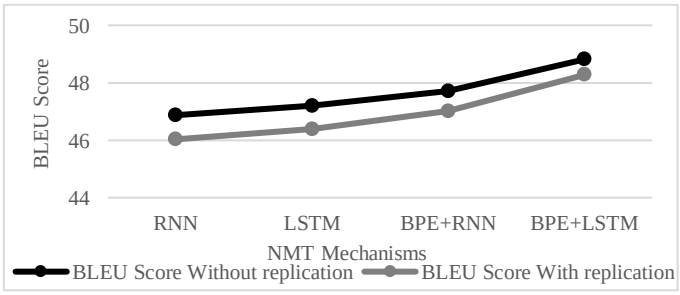

Figure 6. Comparison of the corpus with and without replication using BLEU score

5. CONCLUSION

In NMT, removing replication in the parallel corpus will improve the translation accuracy. Applying BPE solves the unknown words problem like OOV. In this paper, the parallel corpus with and without data replication along with BPE is used in the preprocessing phase. The output of the preprocessing phase is given as input for the encoder and decoder phases. The model was tested for both English-Telugu language pairs. The quality of translation is measured by using the accuracy, perplexity, cross-entropy and BLUE scores. By analyzing the performance of various techniques, it was shown that the model not having replicated corpus is showing better accuracy for translations. So, removing the replicated data in parallel corpus and solving the OOV problems are the two important steps for improving the accuracy of translation in resource-poor languages. Preprocessing has shown a slight improvement in the quality of translation, which is countable. Preprocessing can be considered as one of the essential steps in NMT. So, NMT with effective preprocessing like removal of data replication along with BPE has performed better for the English-Telugu parallel corpus. Thus, by using the NMT with efficient preprocessing for English-Telugu parallel corpus improves the translation accuracy of CLIR.

REFERENCES

[1] Karunesh Kumar Arora, Shyam S. Agrawal, "Pre-Processing of English-Hindi Corpus for Statistical Machine Translation," *Computación y Sistemas*, pp. 725-737, 2017, doi: 10.13053/CyS-21-4-2697.

[2] Adam Lopez, "Statistical Machine Translation, " *ACM Computing Surveys*, Vol. 40, Issue No. 3, Article 8, August 2008, doi: https://doi.org/10.1145/1380584.1380586.

[3] Mikel L. Forcada and Ramon P. Neco, "Recursive hetero-associative memories for translation," *In Mira J., Moreno-Díaz R., Cabestany J. (eds) Biological and Artificial Computation: From Neuroscience to Technology. IWANN, Lecture Notes in Computer Science*, vol 1240. Springer, Berlin, Heidelberg, 1997. doi: https://doi.org/10.1007/BFb0032504

[4] Mussyazwann Azizi Mustafa Azizi, Mohammad Nazrin Mohd Noh, Idnin Pasya, Ahmad Ihsan Mohd Yassin, Megat Syahirul Amin Megat Ali, "Pedestrian detection using Doppler radar and LSTM neural network," *IAES International Journal of Artificial Intelligence (IJ-AI)*, Vol. 9, No. 3, pp. 394-401, 2020, doi: http://doi.org/10.11591/ijai.v9.i3.pp394-401

[5] Nal Kalchbrenner and Phil Blunsom,"Recurrent Continuous Translation Models,"*In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle. Association for Computational Linguistics*. 2013, Web Link: https://www.aclweb.org/anthology/D13-1176.

[6] Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba,"Addressing the Rare Word Problem in Neural Machine Translation,"*In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China. Association for Computational Linguistics* Vol. 1, pp. 11–19, 2015, doi: 10.3115/v1/P15-1002.

[7] Ahmed Y. Tawfik, Mahitab Emam, Khaled Essam, Robert Nabil and Hany Hassan, "Morphology-Aware Word-Segmentation in Dialectal Arabic Adaptation of Neural Machine Translation", In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pp. 11-17, 2019, doi: 10.18653/v1/W19-4602

[8] Pan, Yirong & Li, Xiao & Yang, Yating & Dong, Rui, "Morphological Word Segmentation on Agglutinative Languages for Neural Machine Translation," *ArXiv.org*, 2020, Web Link: https://arxiv.org/abs/2001.01589.

[9] Taku Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pp. 66–75 Vol. 1, 2018, doi: 10.18653/v1/P18-1007

[10] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio,"On Using Very Large Target Vocabulary for Neural Machine Translation,"*In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China. Association for Computational Linguistics*,Vol. 1, pp. 1-10, 2015, doi: 10.3115/v1/P15-1001.

[11] Thang Luong, Richard Socher, and Christopher D. Manning, "Better Word Representations with Recursive Neural Networks for Morphology,"*In Proceedings of the Seventeenth Conference on Computational Natural Language Learning, CoNLL2013, Sofia, Bulgaria*, August 8-9, pp. 104-113, 2013, Web Link: https://www.aclweb.org/anthology/W13-3512.

[12] 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 Association for Computational Linguistics*, August 7-12, pp. 1715-1725, 2016, doi: 10.18653/v1/P16-1162

[13] Mai Oudah, Amjad Almahairi and Nizar Habash, "The Impact of Preprocessing on Arabic-English Statistical and Neural Machine Translation,"*ArXiv.org*, Aug. 19-23, pp. 214-221, 2019, Web Link: https://www.aclweb.org/anthology/W19-6621.

[14] Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov, "Enriching Word Vectors with Subword Information", *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135-146, 2017, doi: 10.1162/tacl_a_00051.

[15] Carlos Escolano, Marta R. Costa-jussà, and José A. R. Fonollosa, "The TALP-UPC Neural Machine Translation System for German/Finnish-English Using the Inverse Direction Model in Rescoring," *In Proceedings of the Second Conference on Machine Translation*, Vol. 2, pp. 283–287, 2017, doi: 10.18653/v1/W17-4725.

[16] Noe Casas, José A. R. Fonollosa, Carlos Escolano, Christine Basta, and Marta R. Costa-jussà, The TALP-UPC Machine Translation Systems for WMT19 News Translation Task: Pivoting Techniques for Low Resource MT, *In Proceedings of the Fourth Conference on Machine Translation*, Vol. 2, pp. 155–162, 2019, doi: 10.18653/v1/W19-5311.

[17] B.N.V Narasimha Raju, M S V S Bhadri Raju, "Statistical Machine Translation System for Indian Languages,"*6th International Advanced Computing Conference*, pp. 174-177, 2016, doi: 10.1109/IACC.2016.41.

[18] Duygu Ataman, M. N., Marco Turchi, Marcello Federico, "Linguistically Motivated Vocabulary Reduction for Neural Machine Translation from Turkish to English,"*The 20th Annual Conference of the European Association for Machine Translation (EAMT)*, pp. 331-342, 2017, doi: 10.1515/pralin-2017-0031.

[19] Anoop Kunchukuttan, P. M., Pushpak Bhattacharyya, "The IIT Bombay English-Hindi Parallel Corpus,"*European Language Resources Association (ELRA)*, pp. 3473-3476, 2018, Web Link: https://www.aclweb.org/anthology/L18-1548.

[20] Kyunghyun Cho, B. V. M., Dzmitry Bahdanau, Yoshua Bengio, "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches,"*Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103-111, 2014, doi: 10.3115/v1/W14-4012.

[21] Philip Gage, "A New Algorithm for Data Compression,"*CUsers J.*, pp. 23-38, February, 1994, Web Link: https://dl.acm.org/doi/10.5555/177910.177914.

[22] Mattia A. Di Gangi and Nicola Bertoldi and Marcello Federico, "FBK's Participation to the English-to-German News Translation Task of WMT 2017", *Proceedings of the Conference on Machine Translation (WMT)*, Vol. 2, pp. 271-275, 2017, doi: 10.18653/v1/W17-4723.
[23] Graham Neubig, Taro Watanabe, Shinsuke Mori1 and Tatsuya Kawahara, "Machine Translation without Words through Substring Alignment", *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pp. 165-174, 2012, Web Links: https://www.aclweb.org/anthology/P12-1018.
[24] Rohan Chitnis and John DeNero, "Variable-Length Word Encodings for Neural Translation Models", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2088-2093, 2015, doi: 10.18653/v1/D15-1249.
[25] T. Mikolov, A. Deoras, D. Povey, L. Burget and J. Černocký, "Strategies for training large scale neural network language models," *IEEE Workshop on Automatic Speech Recognition & Understanding, Waikoloa, HI, USA*, pp. 196-201, 2011, doi: 10.1109/ASRU.2011.6163930.
[26] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, "Sequence to Sequence Learning with Neural Networks," *NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems*, Vol. 2, pp. 3104-3112, 2014, Web Link: https://dl.acm.org/doi/10.5555/2969033.2969173.
[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, Vol. 9, Issue No. 8, 1997, doi: 10.1162/neco.1997.9.8.1735.

BIOGRAPHIES OF AUTHORS

Mr B N V Narasimha Raju is a Research Scholar in the Department of CSE in K L Deemed to be University. He obtained his master's degree from SRKR Engineering college affiliated to Andhra University-Vishakapatnam. He also has a teaching experience of 6 Years. His area of interest is Information retrieval, Machine Learning, Deep Learning, Machine Translation. He is a Life Member of Computer Society of India (CSI), Institute of Engineers (IE).

Dr. Bhadri Raju M S VS received Ph.D. from JNTU University Hyderabad (JNTUH) and has a total teaching experience of 24 years and research experience of 10 years. He published more than 45 papers in international journal and conferences. His area of interest are Information Security, Machine Learning and Information Retrieval. He is a senior Member of IEE, Member of ACM, Life Member of Computer Society of India (CSI), Indian Society for Technical Education (ISTE), Institute of Engineers (IE), Cryptography Research Society of India (CRSI). Currently he is associated with SRKR Engineering, Bhimavram, AP, as Professor and Head of CSE Department.

Prof. K.V.V. SATYANARAYANA, is working as Professor in the Department of CSE in K L Deemed to be University since 2012 with specialization areas in Bioinformatics and Cloud Computing. He has 32+ years of teaching experience both in UG and PG Engineering Courses. He obtained his Engineering Masters degree from JNTUK-Kakinada and Doctoral Degree from Acharya Nagarjuna University. He worked as Director and Head of the Department at other institutions. He has more than 60 National and International peer reviewed publications. He has organized and attended a many National and International conferences and workshops. He looked after so many administrative rolls in K L University and now he is acting as Coordinator for M. Tech Courses.