sciendo INTERNATIONAL JOURNAL ON SMART SENSING AND INTELLIGENT SYSTEMS

Article | DOI: 10.2478/ijssis-2023-0007 Issue 1 | Vol. 16 (2023)

Performance Comparison of Statistical vs. Neural-Based Translation System on Low-Resource Languages

Goutam Datta[1,2], Nisheeth Joshi[1] and Kusum Gupta[1]

[1]Deparment of Mathematical and Computer Science, Banasthali Vidyapeeth, Rajasthan, India

[2]School of Computer Science, University of Petroleum and Energy Studies, Dehradun, India

*E-mails: gdatta1@yahoo.com; jnisheeth@banasthali.in; gupta_kusum@yahoo.com

Received for publication February 22, 2023.

Abstract

One of the important applications for which natural language processing (NLP) is used is the machine translation (MT) system, which automatically converts one natural language to another. It has witnessed various paradigm shifts since its inception. Statistical machine translation (SMT) has dominated MT research for decades. In the recent past, researchers have focused on developing MT systems based on artificial neural networks (ANN). In this paper, first, some important deep learning models that are mostly exploited in Neural Machine Translation (NMT) design are discussed. A systematic comparison was done between the performances of SMT and NMT concerning the English-to-Bangla and English-to-Hindi translation tasks. Most of the Indian scripts are morphologically rich, and the availability of a sufficient corpus is rare. We have presented and analyzed our work and a survey was conducted on other low-resource languages, and finally some useful conclusions have been drawn.

Keywords

Neural machine translation, statistical machine translation, RNN, deep learning

I. Introduction

Machine translation (MT) translates the source to the target language automatically. Numerous attempts have been made in the past to automate and improve translation performance. The MT industry witnessed many challenges while designing translation systems. The linguistic aspect is one of the key challenges. Corpus-based MT systems learn from the corpus we use to train the model. Statistical machine translation (SMT) is a language-independent and corpus-based translation system. The statistical phrase-based translation system outperforms the word-based method [1]. The classical rule-based approach was used before SMT, which was a very lengthy and tiresome process because of the need for the formulation of numerous rules that required manual designing and implementation. The language model in the SMT system considers any given string of words s_1, s_2…….s_n and attempts to predict the next words based on its preceding words $P(s_n|s_1, s_2……..s_{n-1})$ whereas the translation model is responsible for translating the source to target sentence word by word. From the linguistic aspect, one of the major challenges in NMT is corpus collection. NMT requires a huge amount of data for training the system. Sometimes it is a challenging task to obtain a huge corpus, specifically in the case of low-resource languages. Due to the structural and morphological variances, English to low-resource language translation has many challenges. For instance, the English language's syntactic structure follows the subject-verb-object (SVO) order [2]. But in the case of some of the Indian languages, such as Bengali and Hindi, what is followed is the subject object verb (SOV) syntactic structure. In a traditional encoder–decoder model of the NMT system, the encoder converts the whole input sentence into a fixed-length vector. The decoder then covers that up to produce the target sentence. Such a type of encoder–decoder-based

Open Access. Published by Sciendo. CC BY-NC-ND
© 2023 Datta et al. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 License https://creativecommons.org/licenses/by-nc-nd/4.0/

approach is problematic for longer sentence sizes. This issue is addressed by a novel architecture that allows a model to search for a set of input words that are relevant to target words without requiring the entire sentence to be converted into a fixed-length vector. This type of model uses a bidirectional Recurrent Neural Network (RNN) as an encoder and is able to generate good results for longer sentence sizes [3]. Introducing a deep neural network into SMT has so many advantages and improvements in performance. Deep learning has assumed its role in language models, translation models, word alignment, etc. [4]. Our present case study considers two low-resource Indian languages, viz., Bengali and Hindi. We have trained and tested our NMT model with English to Bangla and English to Hindi language pairs. We have tested our models with two types of optimizers, Adam and stochastic gradient descent (SGD), and evaluated the model's performance with the automatic metrics BiLingual Evaluation Understudy(BLEU)-1, BLEU-2, BLEU-3, and BLEU-4 [5]. Finally, we have drawn some useful conclusions based on the results obtained. In future work, we propose the implementation of a generative adversarial network (GAN) for our low-resource language pairs.

The remainder of this paper is organized as follows. Section II describes various MT architectures in different phases with their merits and demerits. With the help of simulation, we have attempted to briefly describe some important layers of transformer-based architecture. Section III reports a detailed analysis of the performance of NMT systems for various low-resource language pairs as presented by different researchers. Section IV includes our case studies on two resource-poor and highly morphological Indian languages: English ⇒ Bengali and English ⇒ Hindi. We have reported our findings for these two resource-poor Indic scripts. Section V contains an analysis and discussion. Finally, we conclude and propose future directions in this domain in Section VI.

II. Statistical and Neural MT architecture

Automatic translation is not an easy task, due to linguistic differences between the source and target languages. Also, in the case of a morphologically rich language, the same word may have a different meaning based on the context. Earlier, the SMT system was primarily based on a statistical approach. One of the popular SMT systems is Moses. Its translation model can be trained automatically with the help of any parallel corpus. When the model is trained, then an efficient searching algorithm is applied to select the best possible target output out of many [1], [6], [7]. During the training phase in Moses, it requires data that are aligned at the sentence level. There are two important components of SMT: the language model and the translation model; these need to be tuned separately. The language model requires monolingual data in the target language and is used by the decoder to generate fluent output (Figure 1). The decoder tries to find the best target sentence given the input sentence by a probabilistic approach using translation and language models. There are various components

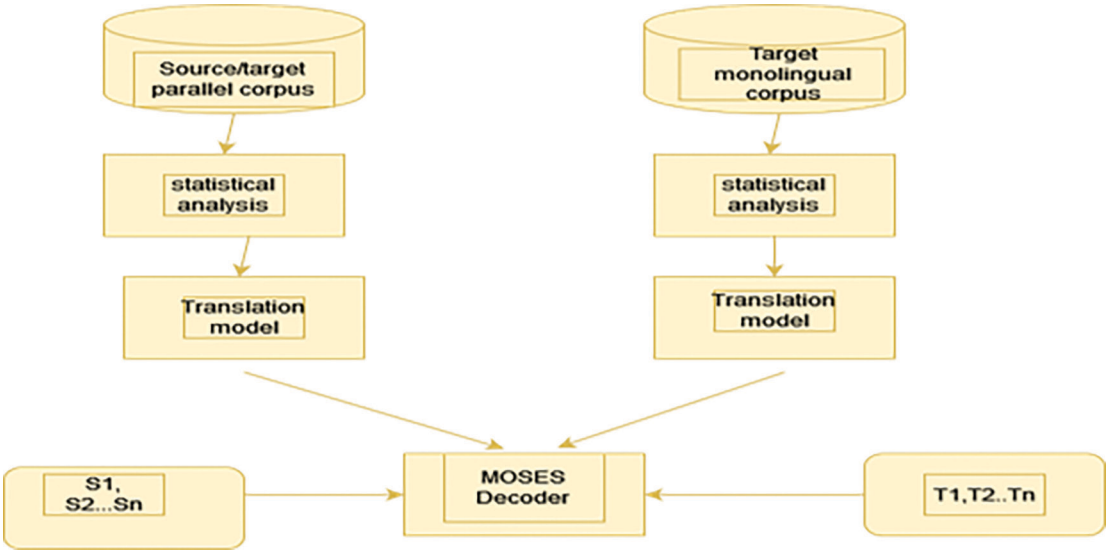


Figure 1: Architecture of SMT system. SMT, statistical machine translation.

that need to be tuned and fine-tuned separately in the language and translation models in SMT systems.

The main objective of SMT is to find a target sentence *t*, which offers a maximum match with the given source sentence *s*, with posterior probability, and can be represented mathematically as follows:

$$P(t|s) = P(s|t)P(t) \quad (1)$$

The above representation indicates, given the source sentence, the maximum probability of finding the target sentence. SMT systems, with different techniques applied in framework level as well as in linguistic level, are explored by different researchers to enhance the performance of the system [8]–[10].

These days we are exploiting artificial neural networks (ANN) in the MT design. NMT has witnessed many paradigm shifts [11]–[15]. One of the models involves sequence-to-sequence learning together with the application of a neural network, where the encoder and decoder of the model have deep Long Short Term Memories (LSTMs) [16]. The attention mechanism plays a very important role in NMT design, namely that of capturing the semantic of the source language when translating it to the target language. In Figure 2, we have encoder–decoder model with attention. The main objective of having attention is to obtain the significance from the sequence of texts. There are two types of attention, namely local and global. In case of local attention, only a subset of the hidden state vectors is taken to generate a context vector. On the other hand, in case of global attention, all the hidden state vectors of the encoder are passed to obtain the context vector. There is much work that has taken place on the various under-resource languages where an attempt has been made toward the performance improvement of NMT with various approaches [17]–[20].

A transformer-based model is the most recent and popular approach (Figure 3). During training, transformer models outperform traditional RNN and convolutional models [21]. In large-scale language modeling applications, the transformer model's Bidirectional Encoder Representation from Transformers (BERT) demonstrated significant performance improvement [22]. RNN's major drawbacks include its inherent sequential nature, i.e., one word at a time stamp, which makes decision-making slow and inefficient. The transformer model, on the other hand, replaces this sequence nature with parallelization, i.e., it considers all sets of input sequences in parallel, resulting in a fast and accurate prediction. The transformer model considers all inputs and their corresponding output sequence to be produced in O(1) operations thanks to its powerful attention mechanism. Transformer models are encoder–decoder-based, have strong attention mechanisms, and eliminate the need for slow RNN, making them ideal for NMT. Transformers or any other deep learning model's attention mechanism can be thought of as a weight vector that determines which of several words in a sequence/sentence is more important and is responsible for determining the context of the entire representation. The attention mechanism scans the entire sequence, and attempts to capture that word to which the highest weightage is given.

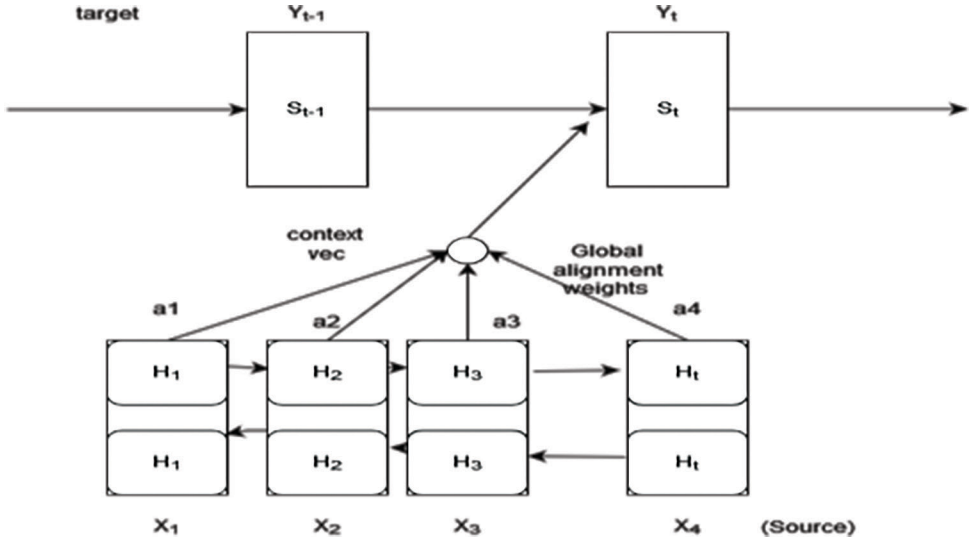


Figure 2: Encoder–decoder model with attention [3].

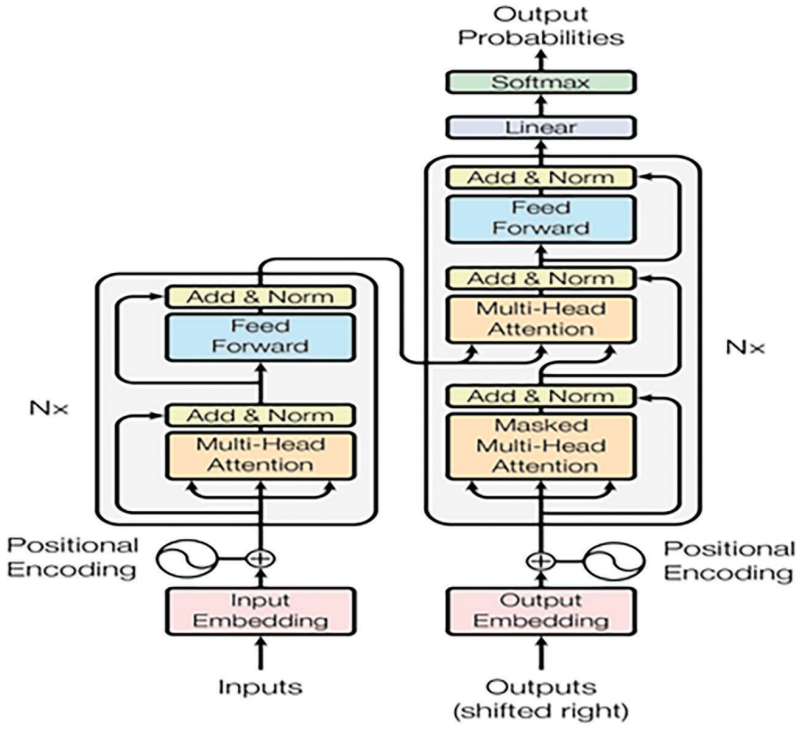


Figure 3: Transformer model [21].

There are many techniques, such as byte-pair encoding (BPE), transfer learning, and so on, that are exploited extensively by researchers to enhance the performance of NMT, specifically during training with low-resource languages [23]–[27]. Simplification of some linguistic level features on the source and the target language was done to improve the performance of NMT system was also explored by the researchers in this domain [28].

The GAN is a new paradigm in the deep learning framework [29]–[31]. Two neural networks compete with each other in GAN. The first network is a generator, and the second is a discriminator. The data generated by the generator are plausible, and in charge of producing plausible example data in the given domain space using a fixed-length random vector with a Gaussian distribution. The discriminator attempts to distinguish between real and fake data (generated by the generator). Nowadays, GAN is widely used by researchers in various cases of use of natural language processing (NLP). Figure 4 depicts GAN in NMT. In this case, the adversary network (D) is a discriminator that receives input from a generator that is our NMT model. The discriminator receives the following inputs: generator's (the NMT model) candidate translation, reference translation, and source text. The discriminator attempts to identify only correct translations for any given input source sentence by comparing the candidate and its reference translation. This comparison either produces a reward (if it matches the reference) or an error signal (if a mismatch happens to be identified). This error signal will be treated as feedback, and the NMT model will be further trained so that it can generate correct translation in the future. Below is an algorithmic representation of how NMT can be implemented with GAN.

Algorithm: GAN in NMT

Input: Source sentence x

Output: Target sentence y

Step 1: NMT model will generate x to y ;

Step 2: Adversary network will try to predict correct translation from its inputs, i.e., ground truth reference translation, candidate/hypothesis translation, and source sentence;

Step 3: Based on the result/prediction by adversary network, rewards will be generated (i.e., successfully predicting correct translation of source and its corresponding target translation, [candidate] translation D(x, y));

Step 4: Now this reward (i.e., correct translation) will be further used for comparison with candidate translation. The difference (if any) between reward and candidate translation will be the error signal;

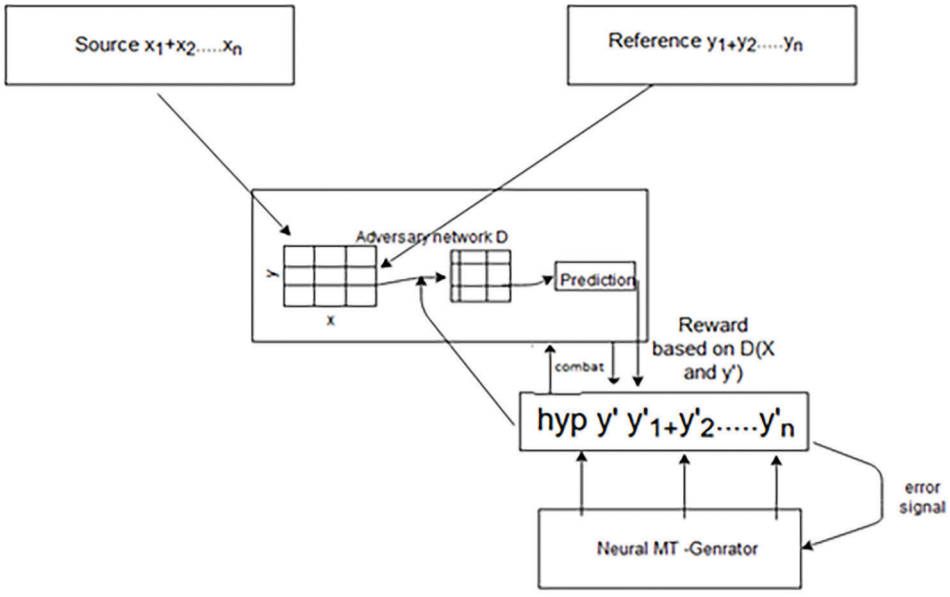


Figure 4: GAN model in the case of NMT use. GAN, generative adversarial network.

Step 5: This signal can be utilized to further train and modify the NMT model, thereby contributing toward its capability for correct translation in the future.

a. Module-wise experimentation with transformer layers

In this section, an attempt has been made to perform some module-wise experimentation using different important layers of a pre-trained transformer model.

Transformer architecture with its various layers is discussed in Section II. Here we have experimented with its two important layers, namely word embedding and attention model. These two are very important phases of any language modeling use cases, such as designing NMT systems. For our experimentation (simulation), we have used NVIDIA's cloud Graphics Processing Unit (GPU) accelerated environment, where they have a pre-trained transformer model in PyTorch for English to German translation. On the encoder side, input words are being embedded and represented in the form of a vector. Word embedding occurs at the bottom layer of the encoder. In the original paper [21], the transformer model uses d_{model} = 512. However, in the NVIDIA environment, d_{model} = 1,024 is taken. When we pass an input string, the tokenizer assigns a number to each word. The last number is 2, which indicates an end of the sentence. This tokenized representation is combined with a positional encoder to ensure representation in the form of an embedded vector. This embedded vector is then fed into the self-attention layer.

As an input, we passed the following string, and as a consequence, the tokenizer assigned a unique number to each word.

We supplied the following strings as input, and obtained the corresponding tokenized representations as follows:

Input Text 1: Bob is intelligent and hard-working.
Tokenized output: [28443 16 11040 9 1749 791 5 2]
Input Text 2: Bob and Greg are intelligent and hard-working.
Tokenized output: [28443 9 5978 492 37 11040 9 1749 791 5 2]

It can be seen from any one of the above patterns that each tokenized word has a unique number, irrespective of its position.

Attention visualization is a way by which one word in a sentence can look for other words and carry out a comparison; and this comparison, in turn, determines a score, i.e., the attention score. This attention score helps to decide what the next word in a sentence will be. This attention score varies from lowest to highest based on the relationship among different words in a sentence (Figure 5).

The decoder part of the transformer is somewhat similar to the encoder except for the addition of an extra sublayer, which acts as multi-head attention over the output of the encoder in the

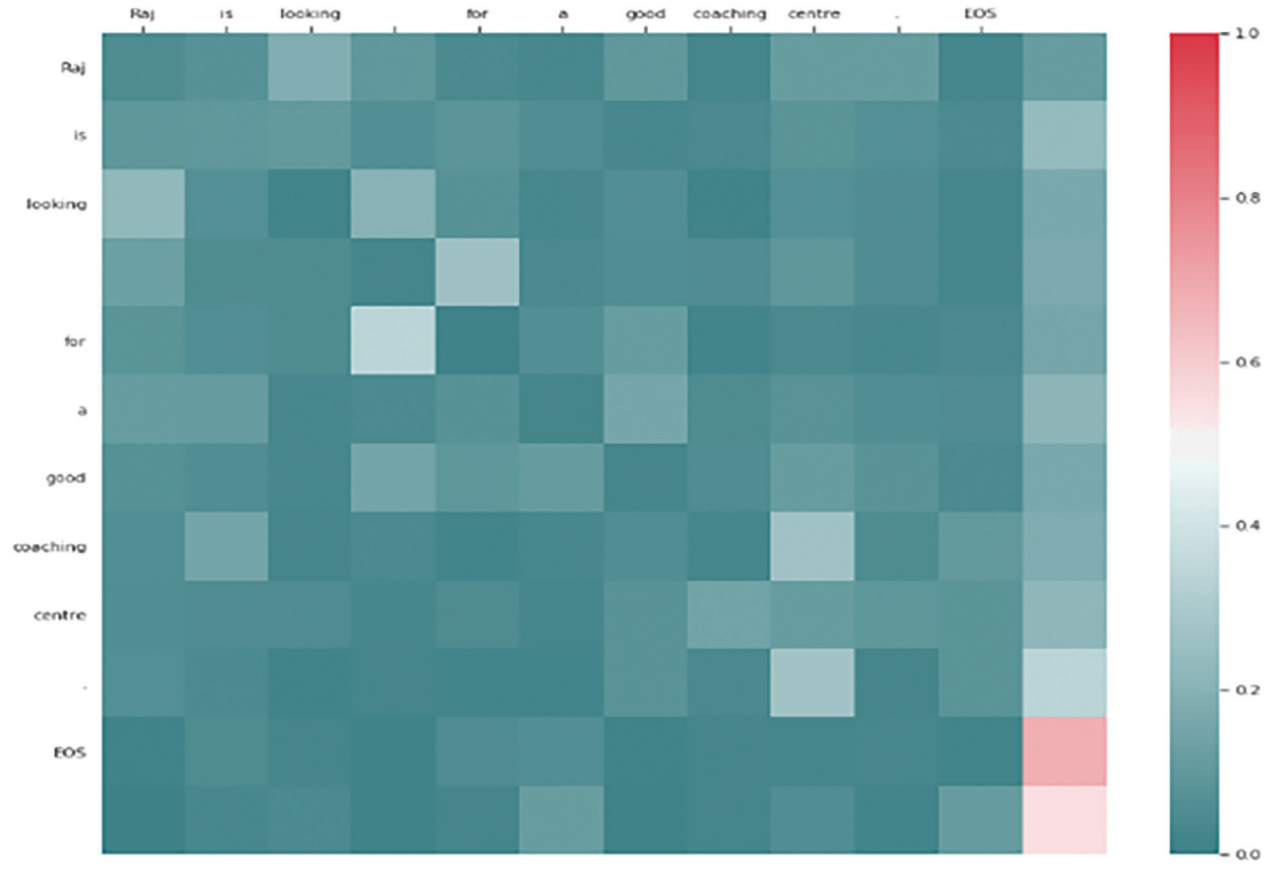


Figure 5: Heat map representation of attention visualization.

encoder–decoder stack. The embedded tokens are fed into a self-attention mechanism. In the decoder, there is a connection between self-attention and a fully connected neural network, which can make it possible to multi-head attention, and through such a contrivance the capability is derived for scanning the entire embedding vector representation *x*, thereby, in turn, enabling the generation of the desired output. In the decoder, words are generated from left to right, one at a time. There are residual connections followed by layer normalization around each sublayer in the encoder and decoder side.

b. Masked multi-head attention

One of the very important layers in the decoder side is the masked multi-head attention layer. In a straight and simple way, its main purpose is to suppress future words so that these cannot be part of the attention. It is masking operation zeros-out the words that appear in the future (in the source words). This mechanism helps to consider only similar preceding terms and not the future terms in MT. Figure 6 is the matrix representation of the masking operation. In the upper right half, all tensor values are zero, representing masked values. Figure 7 presents the operation's graphical representation. Masking helps during source-to-target translation. In fact, in three different places, attention is used in the transformer model. One is the self- attention towards encoder where input sequence pays attention to its own. Decoder side self-attention is responsible for paying attention to its target side sequence. There is another attention in the decoder side called encoder–decoder attention which is responsible for paying attention to the decoder side representation with the help of capturing attention score from the encoder side sequence.

III. Previous work

In this section, we have summarized the work of various researchers in terms of their models' performances for low-resource languages. We have presented our NMT model's result for translations of

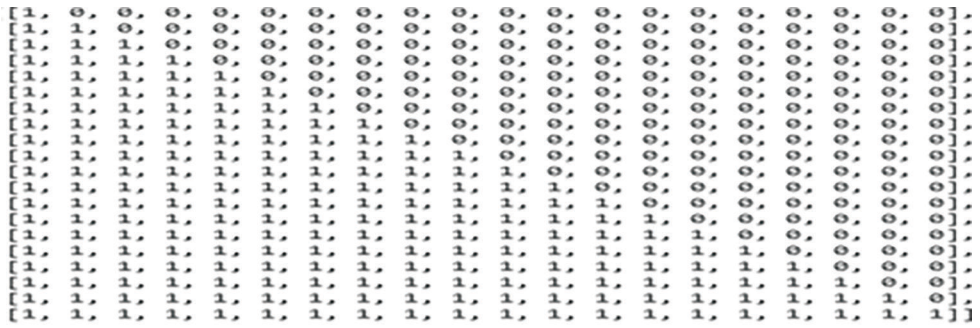


Figure 6: Masked values are represented with zeros.

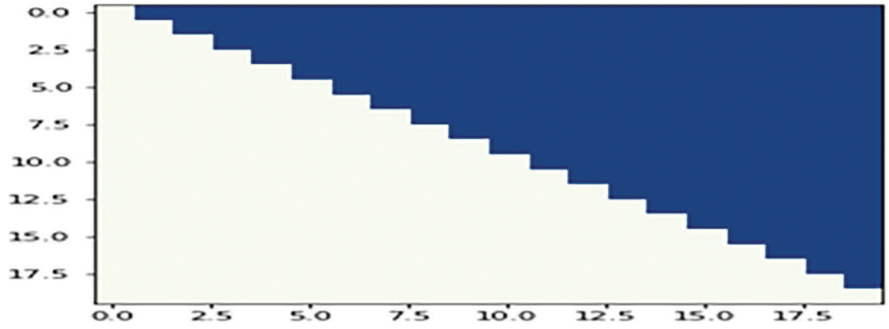


Figure 7: Graphical representation of masking operation; the colored right half is the masked part.

English content into Indian low-resource languages, namely English ⇒ Bengali and English ⇒ Hindi, in the subsequent part.

Ahmadnia et al. [17] presented their finding for the low-resource language pair, Spanish–Farsi, collected from different domains. Results are shown for different NMT systems when different domain data are combined and separated. Finally, these NMT results are compared with the best baseline NMT system. SMT outperformed NMT for low-resource languages. Their findings are summarized in Table 1.

In the study of Zoph et al. [27], the researchers found that transfer learning helps significantly to improve BLEU scores for low-resource languages. The experiment was conducted for low-resource languages such as Hausa, Urdu, Uzbek and Turkish, Uzbek into English. The experiment's method of handling translation tasks involving resource-rich languages, such as French ⇒ English, was taken as the parent model. The first parent model, i.e., the resource-rich language pair model, was trained, followed by the low-resource model. Learned parameters from the parent model were then transferred to the child (low-resource pair) model. This transfer learning process increases the average BLEU score of the child model, by 5.6, to that of its baseline NMT model. Further, when an ensemble of models and unknown words were replaced, it increased 2 more points, and the average BLEU score was resultantly improved by 7.6. Overall, the model achieved significant performance improvement for a low-resource language

Table 1. For various low-resource corpus SMT outperformed NMT (Ahmadnia et al. [17])

Corpus	SMT	NMT	NMT*	NMT**
Gnome	20.54	15.49	17.26	18.76
KDE4	15.64	13.36	14.29	15.71
Subtitles	18.82	18.62	19.51	22.54
Ubuntu	16.76	14.27	15.14	15.87
Tanzil	17.69	15.14	16.53	17.72
Overall	17.06	15.25	16.67	17.32

SMT, statistical machine translation.

pair, and was able to surpass the performance of the syntax-based machine translation (SBMT) model (Table 2).

Das et al. [32], in their paper, used the low-resource Indian language pair Bengali–Hindi in their attention-based NMT model and found a better performance compared to the SMT model. Bengali and Hindi both are morphologically rich languages. Their NMT model started performing better after only 5 out of 25 epochs. They experimented with 50,000 corpora, out of which 49,000 were used for training, and 1,000 were used for testing. The BLEU score details are summarized in Table 3. The researchers also reported that out of 1,000 test sentence pairs, they picked 8 sentence pairs randomly, and that out of these 8 sentence pairs, 5 produced a better result in NMT than in SMT.

Haque et al. [33] conducted their research on terminology translation in a low-resource language domain. Terminology translation is a tough and challenging job in the MT industry. Terminology translation requires context as a very important criterion during translation. Its evaluation again requires domain experts and hence amounts to a very time-consuming process. Their research finding is great and novel in this direction. We have summarized their results in Table 4.

Rubino et al. [20] performed an experiment on extremely low-resource Asian languages and suggested some techniques to improve NMT performance. They performed an experiment on self-attention-based NMT and reported that normal self-attention-based NMT could not achieve a satisfactory result in low-resource languages. To improve performance, they used a transformer-based model, performed a detailed hyper-parameter search in the transformer model, used synthetic data, applied different techniques to generate synthetic data, and employed other techniques to improve the performance of NMT in extremely low-resource languages.

Table 2. NMT outperformed SMT with transfer learning, ensemble, and further processing of data (Zopth et al.)

Language	SBMT	NMT	Transfer	Final
Hausa	23.7	16.8	21.3	24.0
Turkish	20.4	11.4	17.0	18.7
Uzbek	17.9	10.7	14.4	16.8
Urdu	17.9	5.2	13.8	14.5

SBMT, syntax-based machine translation; SMT, statistical machine translation.

IV. Experimental study

In our work, we have selected two pairs of parallel corpora from the low-resource domain, viz., English–Bengali and English–Hindi. Bengali and Hindi are both Indian languages and resource-poor. The source and statistics of our corpora are described in the subsequent part of this section. For both the language pairs, we have taken a subset of the entire corpus in our case studies. We have compared the performance of models such as attention-based NMT [3] with baseline SMT [1]. Hyper-parameter tuning has a major role in any machine learning model [34]–[36]. Further, in the NMT model, we experimented with two

Table 3. Attention-based NMT outperforms SMT for the Bengali–Hindi language pair (Das et al. [32])

Translation model	BLEU score	Iterations
Attention-based NMT model	20.41	25
MOSES (SMT)	14.35	-

SMT, statistical machine translation.

Table 4. NMT system with transformer model and BPE outperformed phrase-based SMT for English–Hindi and Hindi–English language pairs (Haque et al. [33])

MT model	BLEU	METEOR	TER
Eng.Hindi-PBSMT	28.8	30.2	53.4
Eng.Hindi-NMT	36.6	33.5	46.3
Hindi–Eng.PBSMT	34.1	36.6	50.0
Hindi–Eng.NMT	39.9	38.5	42.0

METEOR, Metric for Evaluation of Translation with Explicit Ordering; MT, machine translation; PBSMT, Phrase-Based Statistical Machine Translation System; SMT, statistical machine translation; TER, Translation Error Rate.

different types of optimizers (hyper-parameter): Adam and SGD. Finally, we have reported our observations.

Whenever we select any nonlinear optimizer, it is very difficult to predict when our training process would need to be stopped. We also encountered this difficulty up to a certain extent. Some of the important approaches to overcoming this problem could be to stop training after a certain number of epochs. But here, it is difficult to predict what the appropriate number of epochs will be. Another approach may be to stop training when the error value falls to a certain level. Here, the problem is that this level may never be achieved [37]. We have used English–Bengali parallel sentences (corpus) taken from the Indian Language Technology Proliferation and Deployment Centre (http://tdil-dc.in/). This corpus is from the tourism domain.

This corpus contains 11,976 sentence pairs. We have also collected Corpus Beng.–Eng. from an open Source Parallel Corpus (OPUS) [38]. The corpus name is Tatoeba Challenge data-V2020-07-28. Here, total sentences = 5.2 K, Beng. token = 38.3 K, Eng. token = 13.5 M, test set = 2,500, development set = 2,637, and training set = 2,681,705. We swapped this corpus to obtain Eng.–Beng. We combined all the corpora collected from both sources and shuffled all the data.

As mentioned before, we have trained our NMT models using two different types of optimizers, viz., Adam [39] and SGD.

These two optimizers have their own merits and demerits. Since gradient descent is slow because it uses the entire training data to update the weights and bias of the model, we have used SGD. SGD solved this problem by considering a single record to update weights and other parameters. But convergence in SGD is slow due to its forward and backward propagation, and the path to reaching global minima is also noisy. And the second optimizer, Adam, is a very popular and widely used optimizer because of its lower memory requirements. It is also computationally efficient [39].

We started by setting the learning rate to be 0.01 for both optimizers. For Adam, the batch size was taken as 1. With the Adam-Optimizer, we achieved a BLEU-4 score of 10.78, while with the SGD-Optimizer, we obtained a BLEU score of 11.17. For this English–Bengali language pair, the baseline, which is the SMT model, returned a BLEU score of 14.58 (Table 5). Table 5's BLEU scores are

Table 5. English–Bengali translation BLEU scores using different optimizers

Language pairs	Optimizer	BLEU-4 score	MT model	No. of epochs
Eng.–Beng.	Adam	10.78	NMT with attention	12
Eng.–Beng.	SGD	11.17	NMT with attention	12
Eng.–Beng.		14.58	MOSES	

MT, machine translation; SGD, stochastic gradient descent.

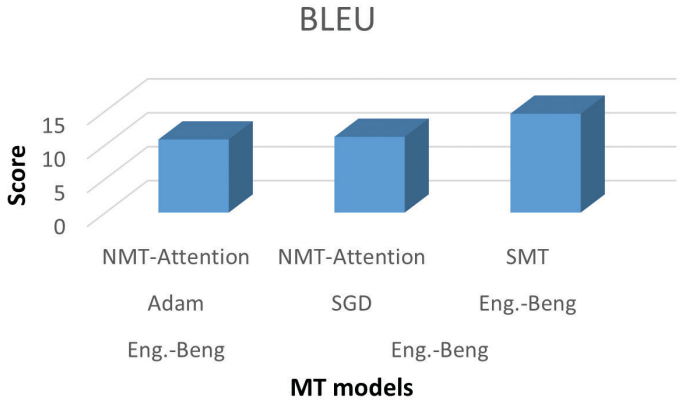


Figure 8: BLEU score generated by NMT and SMT for Eng.–Beng. language pairs. MT, machine translation; SGD, stochastic gradient descent; SMT, statistical machine translation.

represented graphically in Figure 8. For English to Hindi, we have used the data set of IIT Bombay (https://www.kaggle.com/).

It has around 1,561,840 sentence pairs. Using Adam and SGD, we obtained BLEU-4 scores of 12.25 and 11.50, respectively. In the case of English to Hindi, the BLEU score that reported by the baseline SMT model was 16.64 (Table 6). Figure 9 depicts the graphical representation of Table 6.

BLEU scores of other *n*-grams for these two language pairs are summarized in Table 7. Their graphical representation is presented in Figure 10.

Table 6. English–Hindi translation using different optimizers

Language pair	Optimizer	BLEU-4 score	NMT model	No. of epochs
Eng.–Hindi	Adam	12.25	NMT with attention	14
Eng.–Hindi	SGD	11.50	NMT with attention	14
Eng.–Hindi		16.64	MOSES	

SGD, stochastic gradient descent.

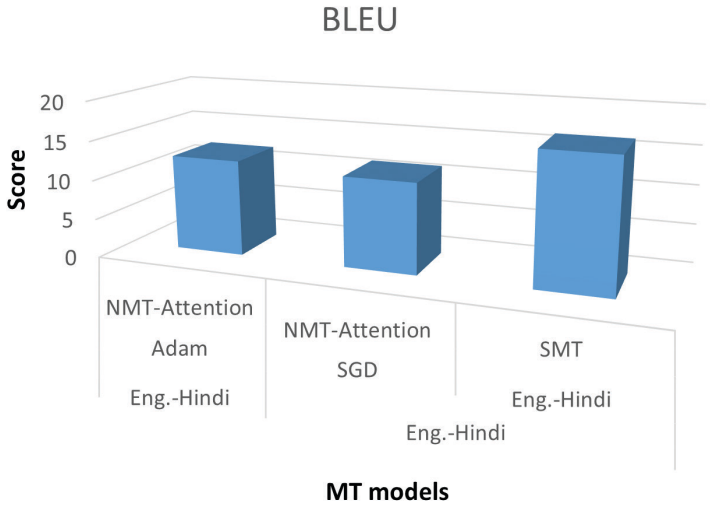


Figure 9: BLEU score generated by NMT and SMT for Eng.–Hindi language pairs. MT, machine translation; SGD, stochastic gradient descent; SMT, statistical machine translation.

Table 7. BLEU-1, 2, and 3 scores are summarized for Eng.–Beng. and Eng.–Hindi language pairs using Adam- and SGD-Optimizers

BLEU	Eng.–Beng.-NMT (Adam-Optimizer)	Eng.–Beng.-NMT (SGD-Optimizer)	Eng.–Hindi (NMT-Adam)	Eng.–Hindi (NMT-SGD)
BLEU-1	14.15	13.91	15.77	14.18
BLEU-2	12.65	13.11	14.12	13.33
BLEU-3	11.83	12.17	13.95	12.19

SGD, stochastic gradient descent.

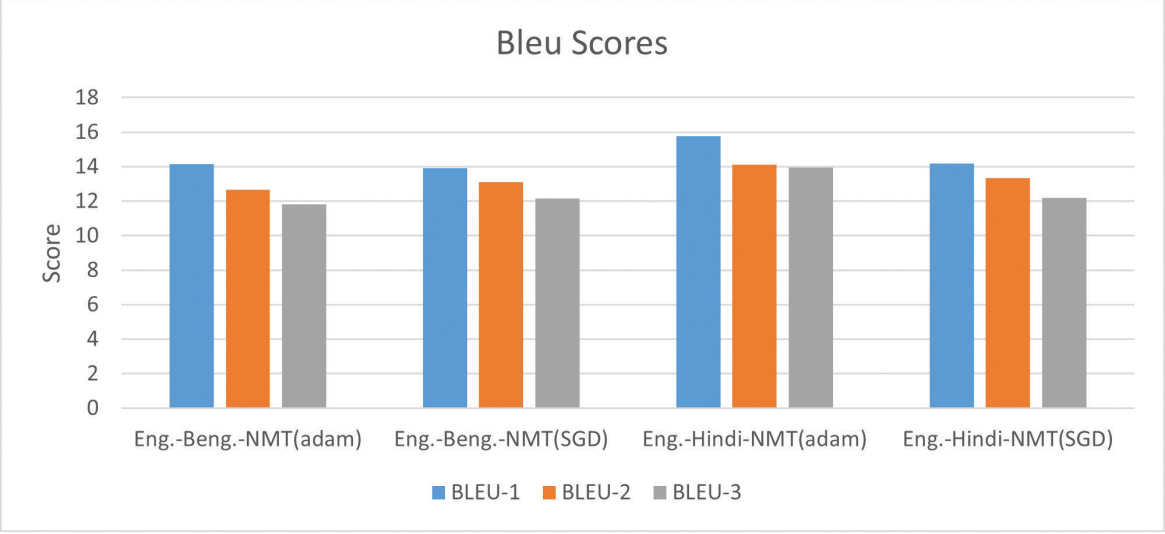


Figure 10: BLEU with minimum n-gram having maximum score. SGD, stochastic gradient descent.

V. Analysis and discussion

The BLEU score that we have achieved for our baseline SMT model and attention-based NMT model can be explained, since we have used the limited parallel corpus for both the language pairs, English–Beng. and English–Hindi. Due to our resource constraints, the NMT model always underperformed compared to its baseline SMT model. For both the hyperparameters, Adam and SGD, we did not notice any major difference in either of the case studies. Compared to different *n*-grams of BLEU, BLEU-1 generated the highest score, but it is only because of BLEU's inherent matching criteria. In our two case studies, we have not focused on any specific linguistic aspect of our corpora during preprocessing. We simply tokenized the sentences. Additionally, we have also not used any other specialized model or any hybrid model.

VI. Conclusion and future work

We can take the following findings from the above survey and our case studies on English–Bengali and English–Hindi language pairs when NMT employs low-resource languages:

- Low-resource languages should be appropriately processed with adequate linguistic feature inclusion before being supplied to NMT for training in order for it to perform well as compared to SMT.
- The pre-trained model with a resource-rich language may be used in low-resource language scenarios with transfer learning (parent–child approach) to achieve better performance in NMT. The parent model is trained using language from a large corpus. Also, back translation, sub-word level (BPE) training, and suitable use of the latest architectures like the transformer model with BERT helped to enhance the performance of the NMT system.

Finally, as future work, we propose to exploit GAN in resource-poor language pairs for exploring the performance of NMT systems.

References

[1] P. Koehn et al., "Moses: open source toolkit for statistical machine translation," *Proc. 45th Annu. Meet. ACL Interact. Poster Demonstr. Sess. - ACL '07*, no. June, p. 177, 2007, doi: 10.3115/1557769.1557821.

[2] A. Banerjee et al., "BENGALI-ENGLISH RELEVANT CROSS LINGUAL INFORMATION ACCESS USING FINITE AUTOMATA," 2010, pp. 595–599, doi: 10.1063/1.3516373.

[3] D. Bahdanau, K. H. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–15, 2015.

[4] M. R. Costajussà, "From feature to paradigm: Deep learning in machine translation," *IJCAI Int. Jt. Conf. Artif. Intell.*, vol. 2018-July, pp. 5583–5587, 2018.

[5] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a Method for Automatic Evaluation of Machine Translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Jul. 2002, pp. 311–318, doi: 10.3115/1073083.1073135.

[6] U. Manual and C. Guide, "Statistical Machine Translation SystemUser Manual and Code Guide," *Univ. Edinburgh*, pp. 1–267, 2012, [Online]. Available: http://www.statmt.org/moses/?n=Moses.Background%5Cnpapers2://publication/uuid/6CE5E8D9-FFCB-4D77-A9A0-EDB52F76AB11%5Cnpapers2://publication/uuid/A34532BD-1847-4471-9C35-B6C8F3C2C108.

[7] J. Kong, Y. Yang, L. Wang, X. Zhou, T. Jiang, and X. Li, "Filtering Reordering Table Using a Novel Recursive Autoencoder Model for Statistical Machine Translation," *Math. Probl. Eng.*, vol. 2017, 2017, doi: 10.1155/2017/3492587.

[8] E. Satir and H. Bulut, "Preventing translation quality deterioration caused by beam search decoding in neural machine translation using statistical machine translation," *Inf. Sci. (Ny).*, vol. 581, pp. 791–807, 2021, doi: 10.1016/j.ins.2021.10.006.

[9] P. Martínez-Gómez, G. Sanchis-Trilles, and F. Casacuberta, "Online adaptation strategies for statistical machine translation in post-editing scenarios," *Pattern Recognit.*, vol. 45, no. 9, pp. 3193–3203, 2012, doi: 10.1016/j.patcog.2012.01.011.

[10] D. Xiong, F. Meng, and Q. Liu, "Topic-based term translation models for statistical machine translation," *Artif. Intell.*, vol. 232, pp. 54–75, 2016, doi: 10.1016/j.artint.2015.12.002.

[11] Y. Zhao, M. Komachi, T. Kajiwara, and C. Chu, "Region-attentive multimodal neural machine translation," *Neurocomputing*, vol. 476, pp. 1–13, 2022, doi: 10.1016/j.neucom.2021.12.076.

[12] S. A. Mohamed, A. A. Elsayed, Y. F. Hassan, and M. A. Abdou, "Neural machine translation: past, present, and future," *Neural Comput. Appl.*, vol. 33, no. 23, pp. 15919–15931, 2021, doi: 10.1007/s00521-021-06268-0.

[13] H. Wang, H. Wu, Z. He, L. Huang, and K. W. Church, "Progress in Machine Translation," *Engineering*, vol. 18, pp. 143–153, 2022, doi: 10.1016/j.eng.2021.03.023.

[14] S. Chauhan, S. Saxena, and P. Daniel, "Improved Unsupervised Neural Machine Translation with Semantically Weighted Back Translation for Morphologically Rich and Low Resource Languages," *Neural Process. Lett.*, vol. 54, no. 3, pp. 1707–1726, 2022, doi: 10.1007/s11063-021-10702-8.

[15] X. Liu, W. Wang, W. Liang, and Y. Li, "Speed Up the Training of Neural Machine Translation," *Neural Process. Lett.*, vol. 51, no. 1, pp. 231–249, 2020, doi: 10.1007/s11063-019-10084-y.

[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 4, no. January, pp. 3104–3112, 2014.

[17] B. Ahmadnia and B. J. Dorr, "Low-resource multi-domain machine translation for Spanish-Farsi: Neural or statistical?," *Procedia Comput. Sci.*, vol. 177, pp. 575–580, 2020, doi: 10.1016/j.procs.2020.10.081.

[18] A. Pathak and P. Pakray, "Neural machine translation for Indian languages," *J. Intell. Syst.*, vol. 28, no. 3, pp. 465–477, 2019, doi: 10.1515/jisys-2018-0065.

[19] S. Jp, V. K. Menon, S. Kp, R. S, and A. Wolk, "Generation of cross-lingual word vectors for low-resourced languages using deep learning and topological metrics in a data-efficient way," *Electronics (Switzerland)*, vol. 10, no. 12. 2021, doi: 10.3390/electronics10121372.

[20] R. Rubino, B. Marie, R. Dabre, A. Fujita, M. Utiyama, and E. Sumita, *Extremely low-resource neural machine translation for Asian languages*, vol. 34, no. 4. Springer Netherlands, 2020.

[21] A. Vaswani et al., "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017.

[22] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, no. Mlm, pp. 4171–4186, 2019.

[23] R. Sennrich and B. Haddow, "Linguistic Input Features Improve Neural Machine Translation," vol. 1, pp. 83–91, 2016, doi: 10.18653/v1/w16-2209.

[24] R. S. and B. H. and A. Birch, "Das Hunderttage-Stadion : Entstehungsgeschichte des Bad Nauheimer Kunsteisstadions unter Colonel Paul R. Knight," *Acl*, pp. 1715–1725, 2016, [Online]. Available: http://www.aclweb.org/anthology/P16-1162.

[25] C. K. Wu, C. C. Shih, Y. C. Wang, and R. T. H. Tsai, "Improving low-resource machine transliteration by using 3-way transfer learning," *Comput. Speech Lang.*, vol. 72, no. February 2020, p. 101283, 2022, doi: 10.1016/j.csl.2021.101283.

[26] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 1–67, 2020.

[27] B. Zoph, D. Yuret, J. May, and K. Knight, "Transfer learning for low-resource neural machine translation," *EMNLP 2016 - Conf. Empir. Methods Nat. Lang. Process. Proc.*, pp. 1568–1575, 2016, doi: 10.18653/v1/d16-1163.

[28] S. K. Mahata, A. Garain, D. Das, and S. Bandyopadhyay, "Simplification of English and Bengali Sentences for Improving Quality of Machine Translation," *Neural Process. Lett.*, 2022, doi: 10.1007/s11063-022-10755-3.

[29] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020, doi: 10.1145/3422622.

[30] Z. Zhang, S. Liu, M. Li, M. Zhou, and E. Chen, "Bidirectional generative adversarial networks for neural machine translation," *CoNLL 2018 - 22nd Conf. Comput. Nat. Lang. Learn. Proc.*, no. CoNLL, pp. 190–199, 2018, doi: 10.18653/v1/k18-1019.

[31] Z. Yang, W. Chen, F. Wang, and B. Xu, "Generative adversarial training for neural machine translation," *Neurocomputing*, vol. 321, pp. 146–155, 2018, doi: 10.1016/j.neucom.2018.09.006.

[32] A. Das, P. Yerra, K. Kumar, and S. Sarkar, "A study of attention-based neural machine translation model on Indian languages," *Proc. 6th Work. South Southeast Asian Nat. Lang. Process.*, pp. 163–172, 2016, [Online]. Available: http://aclweb.org/anthology/W16-3717.

[33] R. Haque, M. Hasanuzzaman, and A. Way, "Terminology translation in low-resource scenarios," *Inf.*, vol. 10, no. 9, pp. 1–28, 2019, doi: 10.3390/info10090273.

[34] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, pp. 1–52, 2018.

[35] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012.

[36] S. Agnihotri, "Hyperparameter Optimization on Neural Machine Translation," *Creat. Components*, vol. 124, 2019, [Online]. Available: https://lib.dr.iastate.edu/creativecomponents/124/.

[37] C. M. Bishop, *Neural Networks for Pattern Recognition*. USA: Oxford University Press, Inc., 1995.

[38] J. Tiedemann, "Parallel data, tools and interfaces in OPUS," *Proc. 8th Int. Conf. Lang. Resour. Eval. Lr. 2012*, pp. 2214–2218, 2012.

[39] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–15, 2015.