IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 13, No. 4, December 2024, pp. 4906~4914
ISSN: 2252-8938, DOI: 10.11591/ijai.v13.i4.pp4906-4914

Transliteration and translation of the Hindi language using integrated domain-based auto-encoder

Vathsala M. K.[1], Sanjeev C. Lingareddy[2]
[1]Department of Information Science and Engineering, Cambridge Institute of Technology, Bangalore, India
[2]Department of Computer Science and Engineering, Vijaya Vittala Institute of Technology, Bangalore, India

Article Info

Article history:

Received Nov 22, 2023
Revised Apr 29, 2024
Accepted Jun 1, 2024

Keywords:

Dakshina dataset
Neural network transliteration and translation
Sequence-to-sequence
Translation
Transliteration
Workshop on asian translation 2021

ABSTRACT

The main objective of translation is to translate words' meanings from one language to another; in contrast, transliteration does not translate any contextual meanings between languages. Transliteration, as opposed to translation, just considers the individual letters that make up each word. In this paper, an integrated deep neural network transliteration and translation model (NNTT) based autoencoder model is developed. The model is segmented into transliteration model and translation model; the transliteration involves the process of converting text from one script to another evaluated on the Dakshina dataset wherein Hindi typically uses a sequence-to-sequence model with an attention mechanism, the translation model is trained to translate text from one language to another. Translation models regularly use a sequence-to-sequence model performed on the workshop on Asian translation (WAT) 2021 dataset with an attention mechanism, similar to the one used in the transliteration model for Hindi. The proposed NNTT model merges the in-domain and out-domain frameworks to develop a training framework so that the information is transferred between the domains. The results evaluated show that the proposed model works effectively in comparison with the existing system for the Hindi language.

Corresponding Author:

Vathsala M. K.
Department of Information Science and Engineering, Cambridge Institute of Technology
Bangalore, India
Email: vathsala_12@rediffmail.com

1. INTRODUCTION

In today's world, effective cross-language communication and information access is essential. As the internet continues to expand tremendously, massive amounts of digital information are being generated in several languages, including Hindi and English [1]. Users of cross-language information retrieval (CLIR) can find relevant data in other languages, due to the popularity of these languages and the unambiguous differences in their linguistic frameworks and writing schemes, this is particularly vital for languages like Hindi and English [2]. Translation and transliteration are the two primary techniques employed by CLIR; this study looks at the factors influencing the performance of CLIR's Hindi-to-English transliteration and translation as well as their effectiveness [3]–[5]. More than 40% of Indians are native speakers of Hindi [6], it acts as a lingua franca, uniting individuals from various linguistic and geographic backgrounds all around the nation. In addition to English, Hindi is one of India's 22 officially recognized languages [7], [8]. According to this rating, a substantial quantity of official paperwork, legal papers, and communication is produced in Hindi, necessitating the implementation of efficient and accurate CLIR systems to increase accessibility to this vital information [9], [10]. Due to its broad usage, official position as a language, proliferation of digital content, applications in research and education, as well as its potential for economic and commercial success, Hindi plays a vital role in CLIR at India [11], [12].

For Hindi to English CLIR, transliteration—the process of converting text from one writing system to another while keeping phonetic similarity—is especially helpful due to the variances between the scripts [7], [13]. However, transliteration might not accurately reflect the text's semantic meaning. On the other hand, translation involves altering a text's meaning from one language to another, making it more suitable for transferring semantic information between languages like Hindi and English. Despite the benefits, translation algorithms are prone to errors and often fail to fully convey the meaning of the original text. According to earlier CLIR research, translation-based techniques perform better than transliteration Algorithms. It is currently not obvious if these techniques work across a wide range of language pairs and domains, particularly for CLIR techniques from Hindi to English [14]–[17]. The CLIR methods [18] used today for transliteration and translation from Hindi to English have several problems. Current approaches do not adequately account for language-specific difficulties, such as variances in spelling, grammar, and syntax, which resulted in translation errors and decreased efficacy. Domains must be made more flexible to accept distinct contexts and industry-specific terminology. Additional research is required to decide how to employ advanced deep learning and NLP models, such as BERT, ELMo, and transformer-based systems, in CLIR, these models may potentially enhance outcomes. To correctly analyze and compare the performance of various transliteration and translation techniques, it is crucial to establish appropriate assessment metrics and standards. Researchers can develop Hindi to English-CLIR systems that are more practical and user-friendly by addressing these drawbacks. This research will contribute to the development of comprehensive and user-friendly cross-lingual information retrieval systems that will ultimately benefit numerous industries, such as business, healthcare, education, and government, where accurate and relevant information across languages is crucial. This study will investigate the adaptability and domain-specific effectiveness of deep learning models.

- A framework is proposed, that performs mutual transfer for in and out of the domain learning mechanisms that constantly focus on each other to enhance the overall performance.
- An ensemble-based training mechanism is used for the in-domain, out-domain mechanism, the pre-training is considered for out-domain information, while considering the training process for in-domain, and out-domain knowledge is considered.
- A batch-learning-based mechanism is proposed for training the samples learned adaptively, various samples with difficulties considered while training.
- A model is proposed for the Hindi language that performs transliteration and translation of the given text by utilizing word-to-word embedding.

The BERT model [19] parameters are kept constant when adapters are introduced in between BERT layers and fine-tuned for succeeding tasks. This paper introduces the iterative and length-adjustable non-autoregressive decoder (ILAND) [20], a unique machine translation paradigm that employs a length-adjustable non-autoregressive decoder. The model's superior performance compared to models using a range of non-autoregressive decoders provides empirical support for the model's validity. Several researchers [1], [21]–[23] suggests a knowledge-aware NMT technique that models extra language properties in addition to the word feature. For controlling the quantity of information from various sources that assist in the construction of target words during decoding, we suggest a knowledge gate and an attention gate. A useful and uncomplicated model of the possible cost of each target word should be made available for NMT systems [24], [25].

The research work in this paper is organized as follows: in the first section, a brief introduction is given about the challenges across text-pre-processing, how the transliteration and translation models have been built that overcome the challenges in various languages, and the breakthroughs involved in processing the Hindi language. In section 2 the related work is introduced that gives a brief description of the existing models, in section 3 a neural network without iterations or convolutional operations is developed that focuses on a self-attention mechanism to build an auto-encoder. In section 4 the dataset details and results for transliteration and translation models are shown.

2. PROPOSED METHODOLOGY

The network is a type of neural network without iterations or convolutional operations that focuses on a self-attention mechanism to build an auto-encoder. The input fed is a word embedding or a sequence embedding. Figure 1 shows the proposed workflow. The autoencoder consists of Y embedded layers, multi-self-attention layers, convolutions, and masked multi-self-attention layers model_n. The multi-self-attention layers and the text not generated are masked. An input source is given as v^{n} along the sequence, embedding is transformed into a weight matrix as J^{n}, projection matrix as R^{n} and feature matrix is depicted as L^{n}, self-attention is later applied to $\mathrm{J}^{\mathrm{n}}, \mathrm{R}^{\mathrm{n}}, \mathrm{L}^{\mathrm{n}}$ irrespectively, Softmax is denoted as p.

$$SA(J^n, R^n, L^n) = \text{segment}(J_{n+1}^{n+1} \dots \dots . . J_V^{n+1})P^n \quad (1)$$

$$J_v^{n+1} = pJ_v^n {J_v^n}^X (l_m)^{-2}(L_v^n) \quad (2)$$

$$J_v^n, R_v^n, L_v^n = J^n P_v^j, R^n P_v^r, L^n P_v^l \quad (3)$$

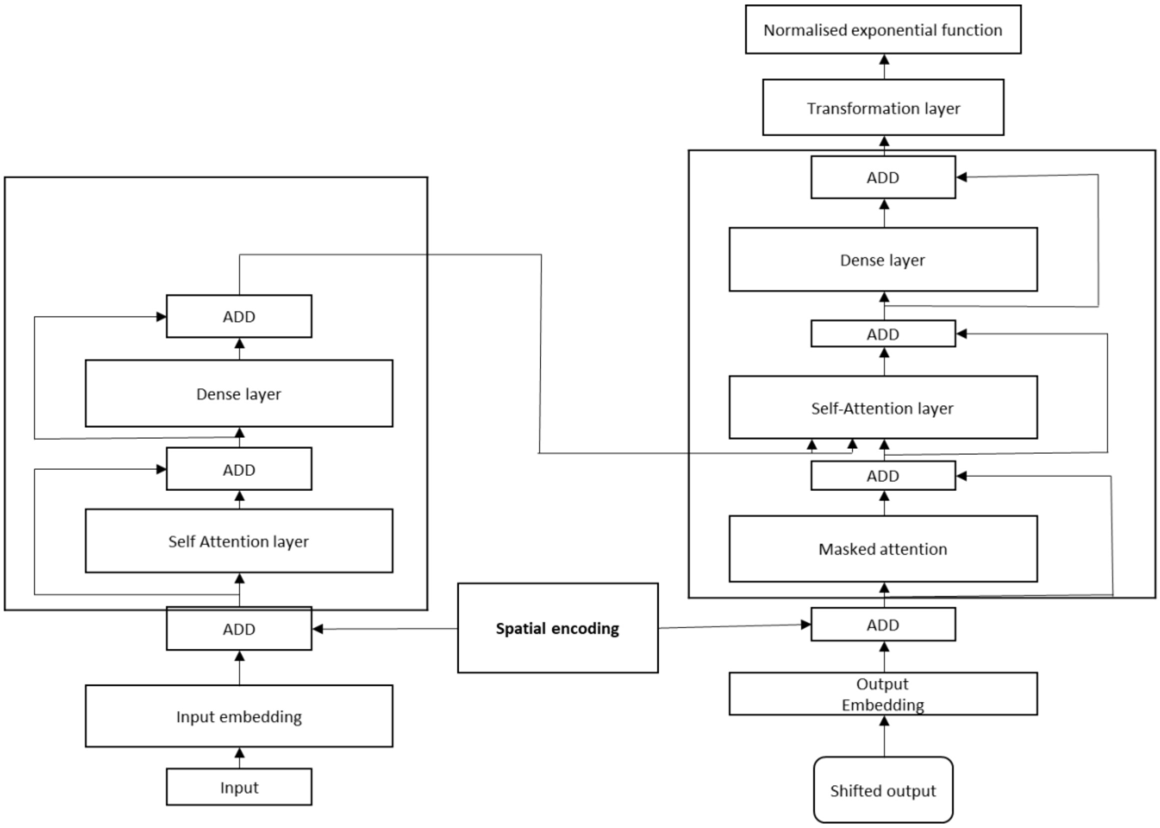

Figure 1. Proposed workflow

Here J_v^n, R_v^n, L_v^n depicts the $v-th$ query and feature matrix of the $n-th$ layer. $\{P_v^j, P_v^r, P_v^l\} \in Q^{mod_{dim}}$ denotes the variable matrix mod_{dim} denotes the dimension of the model respectively. A multi-layer perceptron consists of a fully connected network along with the activation function applied on each position as shown in (4). Here J^{n+1} is the initial source with feature information whereas J^n is added to develop remaining connections that overcome gradient vanishing. The processing sequence is shown as the function g_{SA}, which denotes the source as J^{n+1} as shown in (5). The SA utilizes a set of various layers to learn the source representation as displayed in (6).

$$J^{n+1} = MLP\big(SA(J^n, R^n, L^n)\big) + J^n \quad (4)$$

$$J^{n+1} = g_{SA}^{n+1}(J^n, R^n, L^n) \quad (5)$$

$$[J^x = g_{SA}^x(J^{x-1}, R^{x-1}, L^{x-1})]_X \quad (6)$$

However $[\dots]_X$ ($x \in \{1,2,\dots\dots X\}$) denotes the X similar layers stacked along with each other. The result J_X of the $X-th$ attention layer denotes the final representation transferred to the autoencoder to learn through a translation model that predicts the target. The dissimilarity amid the auto-encoder in the masked layer is shown because the output is developed dynamically. The output decoded estimates the probability $\log c(w_k|w < k, \tau)$ for each word by the softmax function, here τ denotes the variable associated with the auto-encoder.

2.1. Training loss

The proposed model adapts the fine-tuning model in model training. Further, out-model training to focus on the training and perform the KD. Distilling includes two sub-models known as the student and teacher model; the loss of the student model consists of the sum of two components. The loss is determined by probability associated with prediction and the label associated with the negative log loss function. The KD loss function estimates the loss in between the output probability amid the student and teacher model.

$$\mathrm{Loss}(\tau_b; D) = \sum_{(a,b\in D)} \sum_{k=1}^{u} -S(\ w_k) * \log c(\ w_k | w < k, \tau_b) \tag{7}$$

$$\mathrm{Loss}_k(\tau_b; D, \tau_b\hat{}) = \sum_{(a,b\in D)} \sum_{k=1}^{u} -j(\ w_k | w < k, \tau_b\hat{}) * \log c(\ w_k | w < k, \tau_b) \tag{8}$$

Here the $j(w_k|w < k, \tau_b\hat{})$ and τ_b denotes the parameter of the student model, irrespectively. $\tau_b\hat{} = \tau_b *$, the average method $\tau_b = \frac{1}{X}\sum_x \tau_b^{(x)}$. The weighted approach denoted as $\tau_b(x) = \sum_x e - n(Q\ \tau_b^{(x)}))\tau_b^{(x)}$ [here $e - n$] depicts the normalized function for the $x - th$ evaluation for the constraint $\tau_b^{(x)}$ to depict a self-ensemble model. The average and weighted average approaches in the student model here are applicable to have efficient information by the accumulation of data through the previous recursions of the teacher model.

2.2. Domain adaptation

In the proposed approach, the in-domain and out-domain models evaluate by pre-training the model. Each iteration of domain values is beneficial to the preceding iteration for in-domain constraints and vice-versa. These processes are repeated to accomplish mutual transmission of the information. Henceforth the in-domain and out-domain features are transferred across each other at the model level to exchange data thereby ensuring better performance, the quality of the model is evaluated here through source and target domain data D_b,D_f are segmented into training sets D_b^l, D_f^l and model building pair as D_b^{eval}, D_f^{eval} to further train and evaluate the model. Figure 2 shows the training process for in-domain and out-domain learning mechanism.

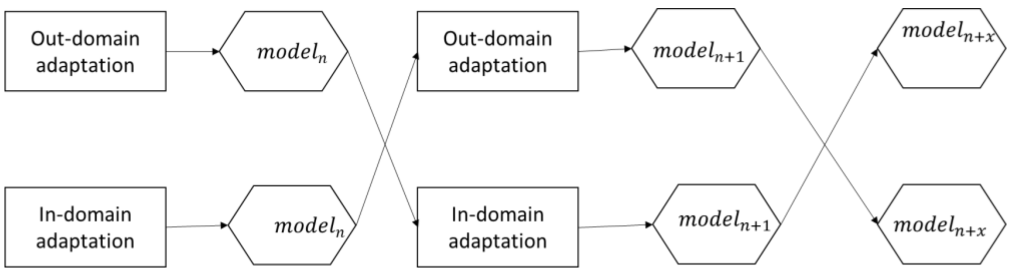

Figure 2. Training process for in-domain and out-domain learning mechanism

Algorithm 1 is the domain adaptation algorithm for the proposed model, this consists of two stages:
- Stage 1: In the preliminary stage, the focus is to finish the preliminary analysis of the in-domain and out-domain model parameters. The **δ** function is used to train the model the objective function is used for training the D_b^l and the parameter used along with it is $\tau_f^{(n+1)}$ initialized on $\mathrm{Loss}_k(\tau_b;\ D_b^l)$, it is retained in for source.
- Stage 2: In the recursion phase, the focus is to finish full data transfer amid the in-domain and out-domain models. The μ function is used to transfer the model, the main aim is to utilize it with the self-knowledge function $\mathrm{Loss}(\tau_f^{(r-1)};\ D_b^l)$ and $\mathrm{Loss}_k(\tau_b^{(r-1)};\ D_f^l\tau_f))$ is used with the training set D_f^l.

To execute model transfers the in-domain parameter set τ_b in a given scenario is initialized through the previous round of the out-domain model parameter set $\tau_f^{(r-1)}$. Once the preliminary analysis is done then the fine-tuning of the model is performed on the in-domain model, and the same is repeated for the source domain. The α model is used for evaluation purposes along with the ensemble function (.) used for the evaluation of the performance of $\tau_b^{(r)}$ for developing the set D_b^{eval} from the ensemble parameter denoted as τ_b. Table 1 displays the Algorithm 1.

Table 1. Algorithm 1 for model transfer

Input	Train the $\{D_b^l, D_f^l\}$, denotes the development lists, $\{D_b^{eval}, D_f^{eval}\}$, with level R
Step 1	In Model training
Step2	$\tau_b^{(n+1)} \leftarrow \text{tr model}(\text{Loss}(\tau_b; D_b^l))$
Step3	Out Model training
Step 4	$\tau_f^{(n+1)} \leftarrow \text{tr model}(\text{Loss}(\tau_f; D_f^l))$
Step 5	Initialize in-model and out-model ensemble model constraints.
Step 6	$\tau_b \leftarrow \tau_b^{(n+1)}, \tau_f \leftarrow \tau_f^{(n+1)}$
Step 7	for $r = 1,2, \ldots ..$ R do
Step 8	In Model training =Transfer training model and evaluation
Step 9	$\tau_b^{(r)} \leftarrow \mu\ (\text{Loss}(\tau_f^{(r-1)};\ D_b^l)\text{Loss}_k(\tau_f^{(r-1)};\ D_b^l\tau_b))$
Step 10	$\tau_b \leftarrow \alpha(D_b^{eval}, \tau_b^{(r)})$
Step 11	out Model training =Transfer training model and evaluation
Step 12	$\tau_f^{(r)} \leftarrow \mu\ (\text{Loss}(\tau_b^{(r-1)};\ D_f^l)\text{Loss}_k(\tau_b^{(r-1)};\ D_f^l\tau_f))$
Step 13	$\tau_f \leftarrow \alpha(D_f^{eval}, \tau_f^{(r)})$
Step 14	end for
output	In Model training τ_b ; outmodel training τ_f

2.3. System design

As a result of the complexities associated with the huge number of words. A neural network based on words offers an end–to–end solution. Henceforth a character level method used as a word-to-word model embedding to evaluate the complexity associated with noise, alterations, and errors is developed.

2.3.1. Pre-processing and post-processing

Input pre-processing: each input word uttered goes through the following steps. All the letters in the word should be in lower case, no more than two times repetition of the character, diacritics are transformed into various versions in the standard 7-bit American standard code for information interchange (ASCII), along the emoji, emoticons involving punctuation are converted into hashtags. Figure 3 shows the sequence-to-sequence architecture.

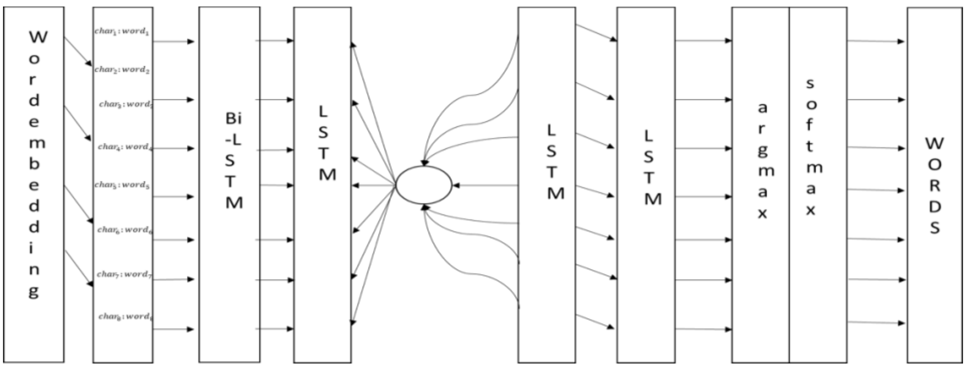

Figure 3. Sequence-to-sequence architecture

Output-side pre-processing: while training the foreign-tagged words into hashtags for the machine learning output. The training input and the output are aligned through the hashtags on the output side. The training input and the output are aligned as the transformation ensures the model learns that identifies foreign words and transfers them into hashtags that are identical to the input-side pre-processing, the output-side free-standing emojis, emoticons, and punctuations are transformed into hashtags during the training, and prediction. Output-side post-processing: on the output side, a post-processing step transforms the hashtags back to words in the source. If the input and output are aligned this step is applicable before removing the tokens [+] and [-]. however, when the final output is reached the words along with the [+] token is merged along with the [-] tokens that are replaced with a white space that splits a word into multiple words.

2.3.2. System architecture

Auto-encoder model: A character-level word-to-word embedding architecture for the model H(a|b)that provides an input b for target a. This auto-encoder consists of two-gated recurrent unit layers; here the first layer is bidirectional. It consists of two gated recurrent unit along with the attention mechanism. The preliminary stage for the auto-encoder involves the attention mechanism, the initial state for the auto-encoder, and recurrent neural networks on the non-recurrent connections required during the training. The final softmax layer for the auto-encoder output to the final of the output sequence a, the loss function is the cross-entropy loss per-time average over a_x. A beam is used during interference via a fixed beam to predict the candidate with the highest log-likelihood at each step, the individual beam along with the highest log-likelihood, in the final step the iterations are decreased by a number, to address the rare case of the autoencoder that stops and produces non-stop repetitions of the text.

3. RESULTS AND DISCUSSION

This section of the paper consists of a results analysis that is obtained by using the neural network transliteration and translation (NNTT) model for translation and transliteration. The accuracy obtained by the performance of the model is evaluated and a comparative study is conducted along different transliteration methods considering accuracy as the measure of performance and results are plotted below. The main aim of this study is to enhance the transfer of information in the Hindi language by improving the model's effectiveness. The dataset details used for transliteration and translation are given below. The simulations are carried out for the proposed model in the INTEL core i7 processor in Python language by utilizing deep learning libraries with 8 GB ransom access memory (RAM) and 64-bit Windows operating system (OS).

3.1. Dataset details

3.1.1. Dakshina (transliteration dataset)

In March 2019, the whole Dakshina dataset [26] was extracted from Wikipedia in 12 South Asian languages. In the dataset four of the twelve languages-kn, ml, ta, and te-are Dravidian, while eight of the twelve are Indo-Aryan. Two of the languages-sd and ur-have texts written in Perso-Arabic scripts or written in Brahmic scripts. Each language has three different data kinds. First, there is Wikipedia material that is written in the language's native orthography and is broken down into training and validation sections. There are specifics on how the compilation's raw data and text are pre-processed. The parallel corpora for the three most important Indian languages-Hindi, Tamil, and Telugu-are included in the Dakshina dataset. It is a useful tool for activities like cross-lingual information retrieval and machine translation, among others. The 2020 workshop on Asian language resources included the publication of the Dakshina dataset, which was produced by Google researchers. There are more than 1.5 million Hindi-English sentence pairings, 1.2 million Tamil-English sentence pairs, and 1 million Telugu-English sentence pairs.

3.1.2. WAT2021 (translation)

The multilingual translation dataset from the workshop on Asian translation (WAT2021) [27] joint project is used. Identical information is given for the English language. The compilation contains almost 1.5 million phrase pairs drawn from diverse literature in the fields of news, communication, information technology (IT), law, and science. Tokenization, normalization, and sentence-level alignment have all been applied to every sentence. It is suitable for developing and testing machine translation models, especially for the aforementioned Asian languages. Researchers looking to enhance the functionality of machine translation systems and develop translation models may find this dataset useful.

3.2. Results

Above we get to know about the datasets, which are being utilized in this research. Further bilingual evaluation understudy (BLEU) scores here are used to evaluate the models. The results here are evaluated in comparison of the existing system with the proposed system.

3.2.1. Transliteration

The publicly available transliteration corpora are compiled for the existing source. The majority of the data comes from the Dakshina corpus [26]. The results here are evaluated in comparison of the existing system with the proposed system for the Hindi language from the Dakshina corpus. The result evaluated is depicted which shows that the accuracy for the existing system is 60.56 whereas the proposed system generates a value of 86.56%. Figure 4 shows the evaluation of the proposed NNTT for transliteration with the existing state-of-art techniques.

3.2.2. Translation

BLEU scores here are used to evaluate the models, The SacreBLEU signatures are included in the Indic-English21 and English-Indic22 assessment annotations to guarantee consistency and repeatability across models, and the publicly available translation corpora are compiled for the existing source. The majority of the data comes from the WAT2021 [27], the results here are evaluated in comparison of the existing system with the proposed system for the Hindi language from WAT2021. The open parallel corpus (OPUS) [28] method generates a value of 13.3, Mbart [29] generates an accuracy value of 33.1, GOOG (Google Inc) [30] generates an accuracy value of 36.7, and microsoft corporation (MSFT) [30] generates a value of 38. Term frequency (TF) [31] generates a value of 38.8, Mt5 (M- Transformer 5) [32] generates a value of 39.2 whereas the existing system generates a value of 40.3 and the proposed neural network translation and transliteration proposed system (NNTT-PS) generates a value of 77.5497%. Figure 5 displays the evaluation of the proposed NNTT for translation with the existing state-of-art techniques.

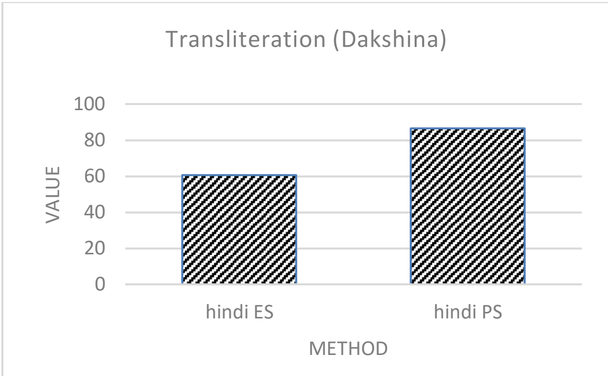

Figure 4. Evaluation of the proposed NNTT for transliteration with the existing state-of-art techniques

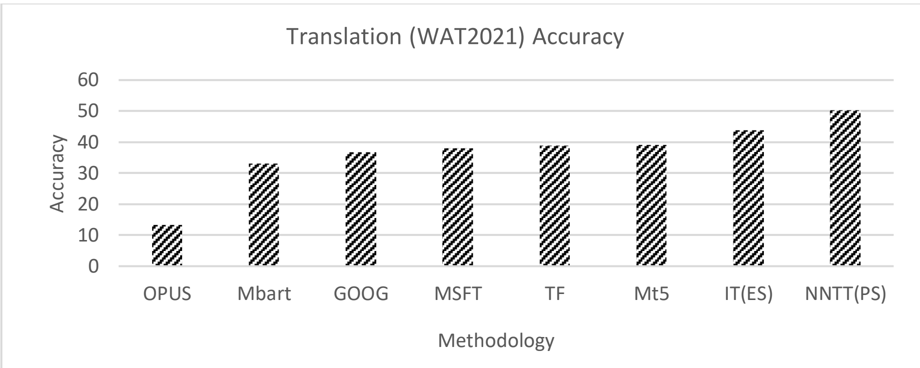

Figure 5. Evaluation of the proposed NNTT for translation with the existing state-of-art techniques

3.3. Comparative analysis

A comparative analysis is carried out for translation and transliteration for the Dakshina dataset and the WAT2021 dataset. The comparison analysis for the transliteration model employed on the Dakshina dataset shows that for the Hindi language, the existing system displays a value of 60.56 and the proposed system depicts a value of 86.56, the percentage of improvisation is 35.3453%, the proposed transliteration model works efficiently generating better accuracy in comparison with the existing system for the Hindi language. The comparison analysis for the translation model employed on the WAT 2021 dataset shows that for the Hindi language, the existing system showcases a value of 40.3 and the proposed system depicts a value of 77.5497. The percentage of improvisation is 63.2156%, the proposed translation model works efficiently generating better accuracy in comparison with the existing system for the Hindi language. Table 2 shows the comparative analysis.

Table 2. Comparative analysis

Dataset	Existing system	Proposed system	Improvisation in %
Transliteration (Dakshina Dataset)	60.56	86.56	35.3453
Translation (WAT 2021 Dataset)	40.3	63.2156	63.2156

4. CONCLUSION

This paper focuses on the use of employing neural networks in transliteration and translation models for information transfer across a framework for in-domain and out-domain models. In the training phase, the autoencoder is responsible to train and deploy efficiently. Pre-training is taken into account for out-of-domain knowledge, and the training process is taken into account for both in- and out-of-domain knowledge. For training, the samples learned adaptively, a batch-learning-based technique is developed, taking into account different samples with problems during training. Word-to-word embedding is used in a model that performs transliteration and translation of the provided text in Hindi. The comparison analysis for the transliteration model employed on the Dakshina dataset shows that for the Hindi language, the existing system showcases a value of 60.56 and the proposed system depicts a value of 86.56, the percentage of improvisation is 35.3453%. The comparison analysis for the translation model employed on the WAT 2021 dataset shows that for the Hindi language, the existing system showcases a value of 40.3 and the proposed system depicts a value of 77.5497. The percentage of improvisation is 63.2156%, the proposed transliteration and translation model works efficiently generating better accuracy in comparison with the existing system for the Hindi language.

REFERENCES

[1] B. Zhang, D. Xiong, and J. Su, "Neural machine translation with deep attention," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 1, pp. 154–163, 2020, doi: 10.1109/TPAMI.2018.2876404.
[2] Y. Nishimura, K. Sudoh, G. Neubig, and S. Nakamura, "Multi-source neural machine translation with missing data," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 28, pp. 569–580, 2020, doi: 10.1109/TASLP.2019.2959224.
[3] H. Moon, C. Park, S. Eo, J. Seo, and H. Lim, "An empirical study on automatic post editing for neural machine translation," *IEEE Access*, vol. 9, pp. 123754–123763, 2021, doi: 10.1109/ACCESS.2021.3109903.
[4] Y. Fan, F. Tian, Y. Xia, T. Qin, X. Y. Li, and T. Y. Liu, "Searching better architectures for neural machine translation," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 28, pp. 1574–1585, 2020, doi: 10.1109/TASLP.2020.2995270.
[5] Y. Zhao and H. Liu, "Document-level neural machine translation with recurrent context states," *IEEE Access*, vol. 11, pp. 27519–27526, 2023, doi: 10.1109/ACCESS.2023.3247508.
[6] K. Mrinalini, P. Vijayalakshmi, and N. Thangavelu, "SBSim: a sentence-BERT similarity-based evaluation metric for indian language neural machine translation systems," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 30, pp. 1396–1406, 2022, doi: 10.1109/TASLP.2022.3161160.
[7] A. Kumar, A. Pratap, and A. K. Singh, "Generative adversarial neural machine translation for Phonetic languages via reinforcement learning," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 1, pp. 190–199, 2023, doi: 10.1109/TETCI.2022.3209394.
[8] S. Bhatia, A. Kumar, and M. M. Khan, "Role of genetic algorithm in optimization of Hindi word sense disambiguation," *IEEE Access*, vol. 10, pp. 75693–75707, 2022, doi: 10.1109/ACCESS.2022.3190406.
[9] S. Saini and V. Sahula, "Neural machine translation for English to Hindi," *Proceedings - 2018 4th International Conference on Information Retrieval and Knowledge Management: Diving into Data Sciences, CAMP 2018*, pp. 25–30, 2018, doi: 10.1109/INFRKM.2018.8464781.
[10] F. Aqlan, X. Fan, A. Alqwbani, and A. Al-Mansoub, "Arabic-Chinese neural machine translation: romanized Arabic as subword unit for Arabic-sourced translation," *IEEE Access*, vol. 7, pp. 133122–133135, 2019, doi: 10.1109/ACCESS.2019.2941161.
[11] Z. Tan *et al.*, "Neural machine translation: A review of methods, resources, and tools," *AI Open*, vol. 1, pp. 5–21, 2020, doi: 10.1016/j.aiopen.2020.11.001.
[12] Q. Li *et al.*, "Linguistic knowledge-aware neural machine translation," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 26, no. 12, pp. 2341–2354, 2018, doi: 10.1109/TASLP.2018.2864648.
[13] I. J. Unanue, E. Z. Borzeshi, and M. Piccardi, "Regressing word and sentence embeddings for low-resource neural machine translation," *IEEE Transactions on Artificial Intelligence*, vol. 4, no. 3, pp. 450–463, 2023, doi: 10.1109/TAI.2022.3187680.
[14] C. Zhou *et al.*, "A multi-task multi-stage transitional training framework for neural chat translation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 7, pp. 7970–7985, 2023, doi: 10.1109/TPAMI.2022.3233226.
[15] C. Duan *et al.*, "Modeling future cost for neural machine translation," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 29, pp. 770–781, 2021, doi: 10.1109/TASLP.2020.3042006.
[16] M. Maimaiti, Y. Liu, H. Luan, and M. Sun, "Enriching the transfer learning with pre-trained lexicon embedding for low-resource neural machine translation," *Tsinghua Science and Technology*, vol. 27, no. 1, pp. 150–163, 2022, doi: 10.26599/TST.2020.9010029.
[17] O. Sen *et al.*, "Bangla natural language processing: A comprehensive analysis of classical, machine learning, and deep learning-based methods," *IEEE Access*, vol. 10, pp. 38999–39044, 2022, doi: 10.1109/ACCESS.2022.3165563.
[18] J. A. Ovi, M. A. Islam, and M. R. Karim, "BaNeP: an end-to-end neural network based model for Bangla parts-of-speech tagging," *IEEE Access*, vol. 10, pp. 102753–102769, 2022, doi: 10.1109/ACCESS.2022.3208269.
[19] U. K. Acharjee, M. Arefin, K. M. Hossen, M. N. Uddin, M. A. Uddin, and L. Islam, "Sequence-to-sequence learning-based conversion of pseudo-code to source code using neural translation approach," *IEEE Access*, vol. 10, pp. 26730–26742, 2022, doi: 10.1109/ACCESS.2022.3155558.
[20] Q. Du, N. Xu, Y. Li, T. Xiao, and J. Zhu, "Topology-sensitive neural architecture search for language modeling," *IEEE Access*, vol. 9, pp. 107416–107423, 2021, doi: 10.1109/ACCESS.2021.3101255.
[21] O. Firat, K. Cho, and Y. Bengio, "Multi-way, multilingual neural machine translation with a shared attention mechanism," *2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016*, pp. 866–875, 2016, doi: 10.18653/v1/n16-1101.
[22] B. Zhang, D. Xiong, J. Xie, and J. Su, "Neural machine translation with gru-gated attention model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4688–4698, 2020, doi: 10.1109/TNNLS.2019.2957276.
[23] Z. Tan, Z. Yang, M. Zhang, Q. Liu, M. Sun, and Y. Liu, "Dynamic multi-branch layers for on-device neural machine translation," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 30, pp. 958–967, 2022, doi: 10.1109/TASLP.2022.3153257.

[24] J. Guo, Z. Zhang, L. Xu, B. Chen, and E. Chen, "Adaptive adapters: an efficient way to incorporate BERT into neural machine translation," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 29, pp. 1740–1751, 2021, doi: 10.1109/TASLP.2021.3076863.

[25] Y. S. Lim, E. J. Park, H. J. Song, and S. B. Park, "A non-autoregressive neural machine translation model with iterative length update of target sentence," *IEEE Access*, vol. 10, pp. 43341–43350, 2022, doi: 10.1109/ACCESS.2022.3169419.

[26] Y. Madhani *et al.*, "Aksharantar: open indic-language transliteration datasets and models for the next billion users," *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 40–57, 2023, doi: 10.18653/v1/2023.findings-emnlp.4.

[27] G. Ramesh *et al.*, "Samanantar: the largest publicly available parallel corpora collection for 11 indic languages," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 145–162, 2022, doi: 10.1162/tacl_a_00452.

[28] J. Tiedemann and S. Thottingal, "OPUS-MT - building open translation services for the world," *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation, EAMT 2020*, pp. 479–480, 2020.

[29] Y. Tang *et al.*, "Multilingual translation with extensible multilingual pretraining and finetuning," *arXiv-Computer Science*, pp. 1-15, 2020, doi: 10.48550/arXiv.2008.00401.

[30] M. Johnson *et al.*, "Google's multilingual neural machine translation system: enabling zero-shot translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 2017, doi: 10.1162/tacl_a_00065.

[31] A. Vaswani *et al.*, "Attention is all you need," *31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.*, pp. 5999–6009, 2017.

[32] L. Xue *et al.*, "mT5: a massively multilingual pre-trained text-to-text transformer," *NAACL-HLT 2021 - 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* pp. 483–498, 2021, doi: 10.18653/v1/2021.naacl-main.41.

BIOGRAPHIES OF AUTHOR

Vathsala M. K. earned her Bachelor's of Engineering B.E. degree in ISE from VTU, Belagavi in 2007. She obtained her master's degree in M.Tech. (Software Engineering) from R.V. College of Engineering in 2011. Currently she is a research scholar at Vijaya Vittala Institute of Technology, VTU (Belgaum) doing her Ph.D. in Computer Science and Engineering. She has attended many workshops conducted by various universities. Her areas of interest are NLP, machine learning, and block chain technologies. She can be contacted at email: vathsala_12@rediffmail.com.

Sanjeev C. Lingareddy received his Ph.D. in the year of 2012 from JNTU, Hyderabad and currently working as Principal at Vijaya Vittala Institute of Technology, Bengaluru. He has 24 years of rich experience in the academics and 7 years of research experience. His research area includes wireless sensor network, wireless security, cloud computing, and cognitive network. He can be contacted at email: sclingareddy@gmail.com.