A Reflection on Machine Translation Process from Nepali to English

Angel Shrestha[1] Shova Kuikel[2] Samidha Neupane[3] Nabin Lamichhane[4]

[1234]Department of Electronics and Computer Engineering,
Tribhuvan University, Institute of Engineering, Pashchimanchal Campus,
Lamchaur-16, Pokhara, Nepal
Email: [1]73bct604@wrc.edu.np [2]73bct642@wrc.edu.np [3]73bct636@wrc.edu.np [4]babunabin@wrc.edu.np

Abstract. In this present global community, we can find people all around the world in a single platform trying to communicate with their native languages despite having the problem of language gap. This problem was possibly solved using Machine Learning for Natural Language Processing by building a Machine Translator. Tackling the problem of the under-resourced Nepali-English pair, Transformer model is designed, provided with a dataset of 20,000 parallel corpus (Nepali-English) to convert Nepali sentence; preserving the style, technical and grammatical norms to English sentence. This transformer model results in a BLEU Score of 21.04 on a test set with training accuracy 90.76%. In a nutshell this research tokens were processed sequentially; a state vector is maintained that contains a representation of the input Nepali data and converts it to English.

Keywords: Machine Translation, Transformer model, Natural Language Processing, Nepali-English pair, BLEU Score.

1. Introduction

According to the 2011 National census, 44.6 percent of the population speak Nepali as their first language, there are 2.9 million Nepali language speakers in India. Likewise, English is spoken by around 2 billion people all around the world. As per the population speaking Nepali and huge population speaking English, there is an irrefutable need for a translator. There has not been enough work done in this field. Talking now; Google and Facebook provide the Nepali translation from English. So, the theme of our research is to give the English sentence of the inputted sentence in Nepali. This dataset consists of 20000 parallel corpuses. Dataset is visualized before processing. Input lines are vectorized, word context is then derived, <begin> and <end> are added at the starting and the ending of the word context, and finally word embedding and positional encoding is used which is then fed into Multi-Head Attention, that evaluates SoftMax and outputs the probability for predicted English sentences.

2. Related Works

The first machine translation project for the Nepali-English pair was Dobhase. Dobhase was a rule-based system that accepted an input string, parsed it, generated the syntax for the target language and output the translation. It was unable to handle sentences of complex structures (multiple conjunctions, ambiguous words, etc) and has been discontinued. At present, system constitutes bi-lingual dictionary of 22,000 words [1].

Google provides Google Translate; a free service that translates images, speech, text or real-time video incorporating multiple languages from one to another language. Nepali language support was added to Google translate in the 36th stage which was launched in December 2013 and made publicly available. Google Translate has been trying to improve the translation quality by involving native speakers in the correction of translated output and its verification in order to improve the translation quality [12].

NMT is relatively new for the Nepali–English pair. In 2018, Acharya and Bal (2018) used a small portion of the parallel corpus from Nepali National Corpus (NNC) collected by Yadava et al. They applied SMT and NMT techniques. On their test sets, the highest BLEU scores they obtained were 5.27 and 3.28 in SMT and NMT respectively [2].

English to Nepali using SMT was another project that aimed to translate English to the most likely Nepali sentences applying the SMT (Statistical Machine Translation) approach. This project was able to give accuracy of 68% that is 2.7 on 4 [3].

Translation of Nepali to English text using rule based MT taking input as Nepali document and tokenizing to words using Document Tokenize then to Syllabic Tokenizer using Finite State Machine producing equivalent feature structure for target language [13].

Envisioning a Trilingual Machine Translation System for the Language Pairs involves a detailed description for the development of a trilingual parallel corpora and then the technical efforts on developing a Neural Machine Translation model for realizing a Machine Translation System[14].

Hindi - Nepali and viceversa using NMT obtained BLEU of Hindi to Nepali 53.7 and Nepali to Hindi of 49.1 [15].

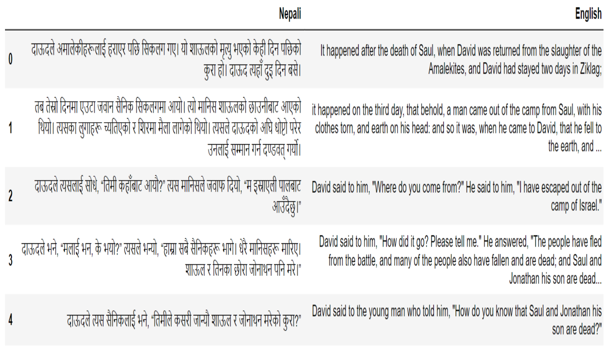
3. Methodology

This research is achieved carefully along the following segments. Every step is carried out specifically with absolute visioning of its precise state.

3.1 Data Extraction

The datasets for the research include the translated Nepali to English in written parallel form. Going through different published research papers and literature references, we used a dataset containing a total number of 16000 valid Nepali-English pairs.

3.2 Data Description



	Nepali	English
0	दाऊदले अमालेकीहरूलाई हराएर पछि सिकलग गए। यो शाऊलको मृत्यु भएको केही दिन पछिको कुरा हो। दाऊद त्यहाँ दुइ दिन बसे।	It happened after the death of Saul, when David was returned from the slaughter of the Amalekites, and David had stayed two days in Ziklag;
1	तब तेस्रो दिनमा एउटा जवान सैनिक सिकलगमा आयो। त्यो मानिस शाऊलको छाउनीबाट आएको थियो। त्यसका लुगाहरू च्यातिएको र शिरमा मैला लागेको थियो। त्यसले दाऊदको अघि धोप्टो परेर उनलाई सम्मान गर्न दण्डवत् गर्यो।	it happened on the third day, that behold, a man came out of the camp from Saul, with his clothes torn, and earth on his head: and so it was, when he came to David, that he fell to the earth, and ...
2	दाऊदले त्यसलाई सोधे, "तिमी कहाँबाट आयौ?" त्यस मानिसले जवाफ दियो, "म इस्राएली पालबाट आउँदैछु।"	David said to him, "Where do you come from?" He said to him, "I have escaped out of the camp of Israel."
3	दाऊदले भने, "मलाई भन, के भयो?" त्यसले भन्यो, "हाम्रा सबै सैनिकहरू भागे। धेरै मानिसहरू मारिए। शाऊल र तिनका छोरा जोनाथन पनि मरे।"	David said to him, "How did it go? Please tell me." He answered, "The people have fled from the battle, and many of the people also have fallen and are dead; and Saul and Jonathan his son are dead...
4	दाऊदले त्यस सैनिकलाई भने, "तिमीले कसरी जान्यौ शाऊल र जोनाथन मरेको कुरा?"	David said to the young man who told him, "How do you know that Saul and Jonathan his son are dead?"

Fig 1: Parallel Nepali-English Corpus

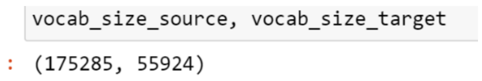
3.3 Data Pre-processing

The parallel corpus consists of two separate text files. One file consists of the Nepali sentences and the corresponding line in the other text file is the translation of those sentences in English. For the pre-processing step the sentences have been lower cased and tokenized into tokens and then the tokenized sentences are used for the purpose of training the NMT system. Input lines are vectorized, word context is then derived, <begin> and <end> are added at the starting and the ending of the word context respectively, and finally word embedding and positional encoding is calculated which is then fed into Multi-Head Attention, that evaluates SoftMax and outputs the probability for predicted English sentences.

Data cleaning:

The data cleaning was done at the very first step which includes tokenized text with white space, words were normalized to lowercase, punctuation removed from each word, non- printable characters were removed.

Tokenization: Tokenization is a common task in Natural Language Processing (NLP) and is a way of separating a piece of text into smaller units called as tokens. Vocabulary is the set of peculiar tokens that is in data. Creating vocabulary is the ultimate goal of tokenization. Word Tokenization splits a piece of text into individual words based on certain delimiter.



```
vocab_size_source, vocab_size_target
: (175285, 55924)
```

Fig 2: Vocabulary size of source(Nepali) and target(English) after tokenization

Vectorization: The process of converting words of sentences into numbers is vectorization. Once we have converted our text samples into sequences of words, we turned these sequences into numerical vectors. Following example shows the indexes assigned to the unigrams generated for inputs and then the sequence of token indexes to which the first text is converted.

Input:'दाऊदले अमालेकीहरूलाई हराएर पछि सिकलग गए।'
Index assigned for each token: {'दाऊदले ': 5, 'अमालेकीहरूलाई ': 3, 'हराएर ': 4, 'पछि ': 6, 'गए': 1,}.
Sequence of token indexes: 'दाऊदले अमालेकीहरूलाई हराएर पछि सिकलग गए' = [5, 3, 4, 6, 1]

Word Embedding: word2vec:

Word embedding is the method that learns a real-valued vector representation for a predefined fixed size vocabulary from the corpus of a language pair in Natural Language Processing. Here words and sentences are mapped into vectors of real-valued numbers. These models are shallow, two-layered neural networks that are trained resulting in reconstruction of words in linguistic contexts[4].

समानता	राजा	रानी	स्याऊ	सुन्तला
व्यक्ति	0.91	0.92	0.02	0.01
रंग	0.02	-0.01	0.04	0.07
फल	0.03	0.04	0.97	0.95

Fig 3: similarity among words using word2vec

Word2vec is an efficient statistical model for learning a standalone word embedding from text data. It takes input from a very large data and produces a vector, consisting of several dimensions, with each unique word in the data are given a corresponding vector in the space. The word vectors are positioned in the equivalent vector space in such a way that words that have common ground in the sentence are located close to one another in a given space.

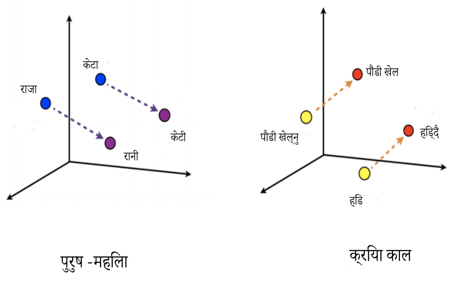


Fig 4: common context of words

4. Model Architecture

4.1 Positional Encoding

The position encoding is inserted into the embedding layer, the position encoder receives input from the embedding layer and applies the relevant position information. Word embedding represents a token value in a n dimension where alike values are close to each other. Positional encoding with word embeddings, results in preservation of position of words in sentence. When spatial coding is added, words come together in a n-dimensional space because of their common ground in sense and relative status in the sentence [5].

In this project, we used cosine for odd and sine functions on even frequencies respectively.

$$PE_{(position,2j)} = sine(position/10000^{2j/dmodel})\ (1)$$

$$PE_{(position,\ 2j+1)} = cosine\ (position\ /\ 10000^{2j/dmodel})\ (2)$$

PE represents positional embedding and j represents frequency dimension. We embodied this work supposing that it will permit the transformer model to effortlessly grasp the addressing by relative positions, since for any set off j, $PE_{position+j}$ can be illustrated to a linear function of $PE_{position}$ [6]. We plotted the graph for positional encoding for our given dataset where position is in y-axis and depth is in x-axis and obtained the following plot.

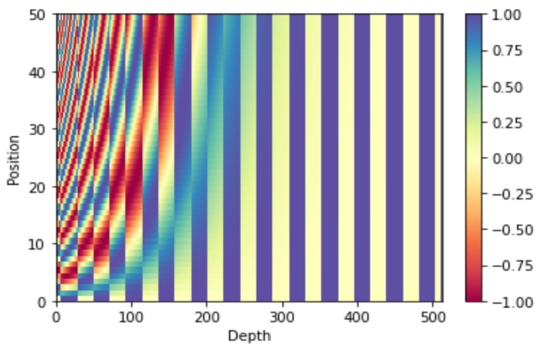


Fig 5: Positional Encoding

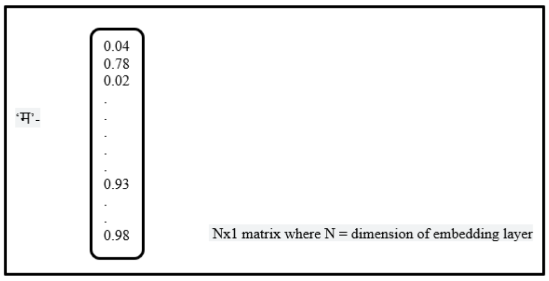


Fig 6: Preservation of position of words

Example: म भात खान्छु is represented by e^0, e^1, e^2 in embedding layer. After adding positional encoding vectors 'p' it is represented as e_p^0, e_p^1, e_p^2

4.2 Scaled dot product attention

The product inserts: Value(C), Query(A), and Key(B) within the transformer model. The queries and keys experience a dot product multiplication, resulting in attention score framework. The score system decides how much weight of a word can be put on other words. So, each word will have a score that compares to other words inside the time-step, the higher the score the more the weight. In this way mapping of keys is done. The scores at that point are down scaled by dividing by the square root of the key. SoftMax of the scaled attention score yields consideration weights which gives likelihood values between 0 and 1. SoftMax results in tincrement of the higher scores and discouragement of the lower one. This increases the certainty of a model to choose the word with higher score and address to the most probable words. At last, attention weights are multiplied by value vectors to yield an output.

$$Att(A, B, C) = SM(AB^T/\sqrt{d}) * C \quad (3)$$

where Att is the attention, SM is the SoftMax; d is the depth of queries and keys and A, B and C are queries, keys and values respectively.

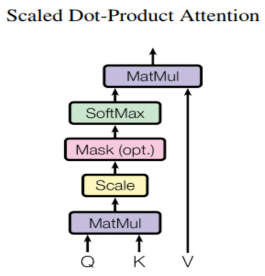


Fig 7: Scaled Dot-Product Attention (*'Attention Is All You Need' by Vaswani et al.'*)

4.3 Multi-head attention

Rather than working with a single head attention with dimension of queries, value and keys; we concluded it advantageous to project keys, values and queries distinctively h times with linear learned direct projections to d_k, d_v and d_q dimensions, respectively. These anticipated versions; then undergo the attention function parallelly, yielding output values of d_v-dimension. The results are then added and again estimated, deriving the final values, as shown in Figure 8. Multi-head attention hence; permits transformer model to mutually address information from diversified sub spaces information from diversified sub spaces representation at distinctive positions. Averaging in single attention head discourages the parallelism.

$$MH(A, B, C) = CnCt(h_1, ..., h_h) W^O \quad (4)$$

where $h_i = Att(QW_i^Q, KW_i^K, V W_i^V)$, MH represents Multi Head Attention and CnCt is summation of parallel attention function of the projection's parameter $W_i^Q \in R^{d_{model} \times d_k}$, $W_i^K \in R^{d_{model} \times d_k}$, $W^V \in R^{d_{model} \times d_v}$ and $W^O \in R^{hd_v \times d_{model}}$.
We have set number of heads equals to 8 that is number of parallel attention head to 8. For each of these we use dimension of keys = dimension of values = (dimensionality of input and output/ number of heads) = 128/8 = 16. As, the dimension reduction is done on each head, the total cost of single-head attention with full dimension is comparably similar to attention of multi head [7].

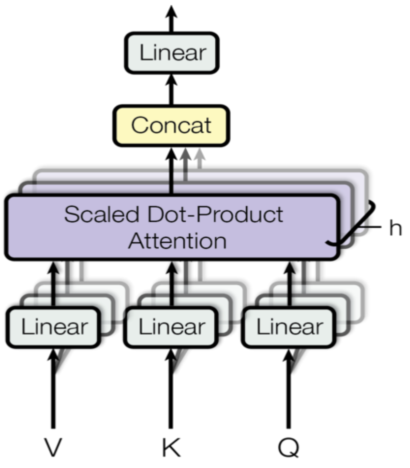


Fig 8: Multi-Head Attention (*'Attention Is All You Need' by Vaswani et al.'*)

4.4 Position wise feed forward network

Every encoder and decoder are applied in every position with fully connected feed-forward network distinctively and uniformly in expansion to attention sub-layers. This consists of 2 transformations linearly with a Rectified Linear Unit in between.

$$F\_F\_N(x) = maximum(0, x * W_1 + b_1) * W_2 + b_2 \quad (5)$$

Different parameters are used at different positions from layer to layer in feed forward network. We have the dimension of output and input that is d_{model} to 128, and size of inner layer (d_{ff}) to 512 [6].

4.5 Encoder and decoder

Encoder:

Our encoder comprises of four uniform layers. Every layer adds 2 inner layers namely multi head attention function, and then a fully connected feed-forward network that considers position. The inner layer is linked to residual connection of previous layer along with normalizing of layers. That is, the result of every inner layer is Layer Normalized (y + Sub_Layer(y)), where Sub_Layer(y) is the result of function executed by inner layers to prevent the loss of features of preceding layers. All sub-layers inside encoder, and embedding layers, too produce result of size d_{model} = 128 set at the beginning to support residual connection [7].

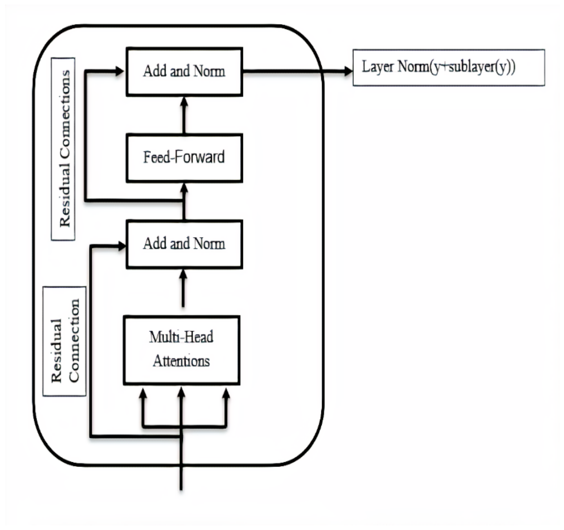


Fig 9: Structure of Encoder

Decoder:

The decoder also comprises 4 and then 2 inner layer, that calculate multi head attention on results of the encoder layer. We add residual connections on each inner layer, along with normalization of layer and self-attention inner layer is adjusted in decoder to ward off positions of the words from addressing to succeeding positions. This look ahead and padding masking merged with the truth that the embeddings are balanced by one, ensuring that the present prediction at time k can rely only on preceding output at (k-1) [8].

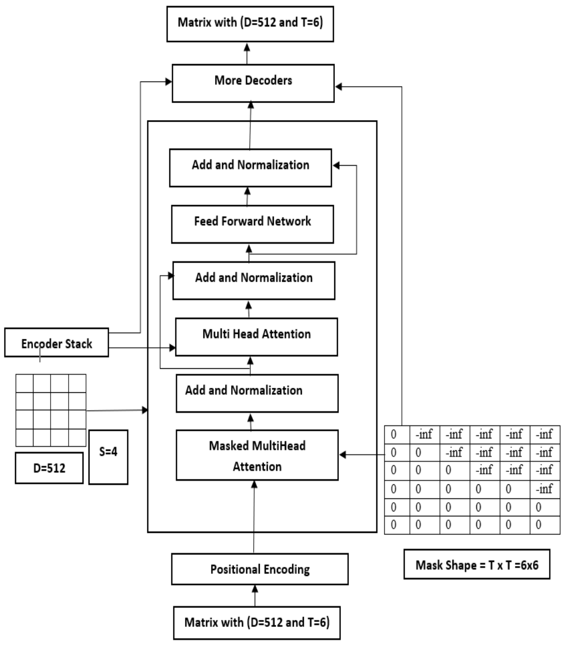


Fig 10: Detailed Structure of Decoder

5. Training

We used 70% of the dataset that is 16000 parallel Nepali-English lines for training and rest of the dataset as the testing data. The Input parameters are: *num_layer, dmodel, dff, numheads, dropout_rate, vocab_size_input, vocab_size_target, input_pe, target_pe*. We have taken 8 lines at a time calculated by dividing input tensor length by batch size. Also setting the checkpoints each at 5 epochs in 70 epochs in total. Optimization is done using the Adam optimizer [9].

optimizer=tf.keras.optimizers.Adam(LearningRate, beta1=0.9, beta2=0.98, e=10^{-9}) (6)

We also varied the learning rate throughout training, implementing the following:

Learning_Rate = $d_{model}^{\ -0.5}$ - minimum ($ns^{-0.5}$, ns * $warmup^{-1.5}$) (7)

We have taken warmup = 4000 and ns is number of steps. The auto-regressive transformer model produces one prediction at a particular time, and take in account its output again to predict future output [9]. While training, the model makes use of teacher-forcing, which then passes the genuine output to another time step in any case of without depending on what model predicts at the current time step.

6. Evaluation and Result

TABLE I: COMPARISON OF MODEL FOR MACHINE TRANSLATION

S. N	Model	BLEU Score on test set
1	Long Short-Term Memory (LSTM)	11.33
2	Bahdanau Attention	15.57
3	Attention Based Transformer Model	21.04

The Attention based Transformer model performed best for machine translation among applied 3 algorithms tested for the scenario. Encoder and Decoder model along with a mechanism of self-attention that employed position wise feed forward network and multi-head attention derived in an eventual increment in performance of the model. Finally, the Bleu Score indicated by Guzman et al in his paper; of our transformer model upgraded to 21.04 with training accuracy 90.76% [11].

7. Discussion and Conclusions

Consequently, the research applies Natural Language Processing to work out the issue of insufficient accuracy in the LSTM model and Bahdanau Attention in Machine Translation by applying Attention based Transformer model. In LSTM and RNN model, sequential processing and past information that are retained through past states were required which makes it difficult to train parallelly and resulted in long dependencies.So, transformer model was implemented which avoids recursion to allow parallel computation and to reduce drop in performance due to long dependencies [10][6].The multi-head attention and positional embedding of transformer model provides the information about the relationship between the different words, such that it does not relies on the past hidden state to capture dependencies with previous words, as they process and translate sentences as whole.

8. Acknowledgement

This research is supported by the Research Management Unit, Pashchimanchal Campus, Pokhara.

References

[1] A. R. Dahal, "Development of a Nepali-English MT System."

[2] P. Acharya and B. K. Bal, "A Comparative Study of SMT and NMT: Case Study of English-Nepali Language Pair.," pp. 90-93, 2018.

[3] A. Paul and B. S. Purkayastha, "English to Nepali Statistical Machine Translation System," in *Proceedings of the International Conference on Computing and Communication Systems*, Singapore, Springer, 2018, pp. 423-431.

[4] A. Dhakal, A. Poudel, S. Pandey, S. Gaire and H. P. Baral, "Exploring deep learning in semantic question matching," *IEEE 3rd International Conference on Computing, Communication and Security (ICCCS),* pp. 86-91, 2018.

[5] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le and R. Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860,* 2019.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jone, A. N. Gomez, L. Kaiser and I. Polosukhin., "Attention is all you need," *arXiv preprint arXiv:1706.03762,* 2017.

[7] P. Shaw, J. Uszkoreit and A. Vaswani, "Self-attention with relative position representations," *arXiv preprint arXiv:1803.02155,* 2018.

[8] K. Irie, A. Zeyer, R. Schlüter and H. Ney., "Language modeling with deep transformers," *arXiv preprint arXiv:1905.04226,* 2019.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980,* 2014.

[10] K. Cho, B. V. Merriënboer, D. Bahdanau and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259,* 2014.

[11] F. Guzmán, P.-J. Chen, M. Ott, J. Pino, G. Lample, P. Koehn, V. Chaudhary and M. Ranzato, "The FLoRes evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english," *arXiv preprint arXiv:1902.01382,* 2019.

[12] Wu, Yonghui, et al. &quot;Google&#39;s neural machine translation system: Bridging the gap between human and machine translation.&quot; arXiv preprint arXiv:1609.08144 (2016).

[13] H. K. Shrestha, "Rule based machine translation system in the context of Nepali text to English text.",2005.

[14]Envisioning a Trilingual Machine Translation System for the Language Pairs – Bal Krishna Bal, Amrit Yonjan Tamang, Lasang Jimba Tamang

[15] Laskar, S. Rahman, P. Partha, S. Bandyopadhyay, "Neural Machine Translation: Hindi-Nepali", in *Proceedings of the Fourth Conference on Machine Translation*, pp. 202-207,2019.