Sanskrit to English Machine translation system

Swarali Purandare, Prof. Jyoti Madake

Department of Electronics and Telecommunication, Vishwakarma Institute of Technology, Pune

***Abstract*— The translation of ancient texts can be considered as a potential source for the content of various digital media and is valuable in fields such as natural phenomena, medicine, and science. Sanskrit is one of the oldest languages in the world. Several popular translation applications, however, do not provide English-Sanskrit translations. There are few public English-Sanskrit translations available. Accurate translations from the model will be challenging to obtain with a small dataset. This paper proposes a Sanskrit to english machine translation system using LSTM.**

I. Introduction

Unambiguousness of Sanskrit language and its neatly laid language rules support this language as the language for processing. Sanskrit is one of the 22 official languages listed in the Indian Constitution's Eighth Schedule. It is a highly significant language in Hinduism, and it is still used in books and chants today. Sanskrit has around five million speakers (as a first, second, or third language) in India, therefore a good, automatic written translation of entire sentences would be quite important.

According to ancient translation statistics, people's names, locations, or institutions take up the majority of the text. Machine translation has evolved along three main lines: rule-based machine translation, statistical-based machine translation, and natural language translation (NMT). However, the majority of study has been on NMT. With the advent of deep learning, NMT has recently been introduced and has shown significant performance improvement [3]. NMT eliminates the drawbacks of rule-based machine translation, such as the complexity of creating rules, dealing with rule collisions, and multiple sub-module conflicts. Furthermore, NMT solves the challenges of displaying lengthier text than a phrase, which is a disadvantage of statistical machine translation. Therefore considering advantages of neural based MT this paper discusses the development of a Sanskrit Machine translation system using neural machine translation.

II. Literature review

Numerous authors [1][2] have stated that Sanskrit is a feasible language for Natural Language Processing since its syntax and grammar are simpler to comprehend than other languages. In Sanskrit, words like "it is" and "I" do not have distinct words but can be simply understood. It lowers a four-letter English term to two letters in Sanskrit due to inflections, making it easier to interpret and taking up less storage space.

There has been a lot of research done in the field of machine translation, and it is still an active area of research. There are already a slew of approaches and models for comprehensive translation that have been tried and proven on a variety of languages. Some of approaches of Machine translation are:

1) Rule based
2)Statistical Machine translation
3) Neural Machine Translation

1) Statistical MT

Young-Suk Lee describes Statistical Machine Translation (SMT)[6], as an older but still efficient translation approach. This approach combined Morphological Syntactic Features, often known as ‘part-of-speech' tags, with SMT, which takes the language and uses its parts-of-speech tags to build features for statistical text analysis. The paper uses prefix and suffix information in its mapping to make the model more powerful [6]

2) Neural MT

Neural Machine Translation has been studied by a number of researchers. Their work is broadly classified as employing sequence-to-sequence learning or Transformer-based learning. [4] Transformer-based Mt not only provides strong BLEU scores, but also does it at a greatly decreased amount of FLOPs (floating-point operations) in training because of its extremely parallelizable nature. Hence, the Transformer is better and faster. Algorithm discussed by Chanjun Park uses shared Vocabulary and Entity Restriction Byte Pair Encoding (SVER BPE). An evaluation of this approach showed it gives a better BLEU score than conventional BPE and SentencePiece . In LSTMs the cell, which is the fundamental building component of LSTM networks, comprises an explicit memory state. The cell's memory state is inspired by digital memory cells seen in ordinary computers. Read, write, and reset operations are supported by digital memory cells.

In LSTM, a digital memory cell stores a real number.

In a LSTM cell flow of information is defined as given by following equations using values of input, memory and output values at time t.

$$memory^t = gate_{input} \times input^t + gate_{forget} \times memory^t$$

$$output^t = gate_{output} \times memory^t$$

In the hidden layes computation is done using activation functions f and input to activation function is output value.

$$h^t = f(output^t)$$

A vector of LSTM cells is present in each LSTM layer. Given the node values for the prior layer x t and the hidden layer values from the previous time step h t1, the input value is a combination of matrix multiplication with weights Wx and Wh and an activation function

$$input^t = g\ (W^x x^t + W^h h^{t-1})$$

LSTM cells include a lot of other parameters. Multiple weight matrices are applied for each gate separately. More parameters mean longer training durations and a higher risk of overfitting. Gated recurrent units (GRU) have been proposed and utilized in neural translation models as a faster approach. LSTM cells seemed to be making a comeback in neural machine.
There is no distinct memory state; instead, there is a hidden state that performs both functions. There are also just two gates. The input and the prior state are used to forecast gate states.The first gate is used to combine the input with the prior state. This combination is identical to standard recurrent neural networks, except that the reset gate scales the influence of the prior state. Because the gate's value lies between 0 and 1, the present input is given priority. The update gate is then utilised to interpolate between the prior state and the newly calculated combination..

III. Methodology

The project can be divided into four parts . One of dataset creation and second of different tokenizers, third of model training and testing and the fourth is named Entity recognition for improving accuracy as well as to reduce ambiguity in translation.

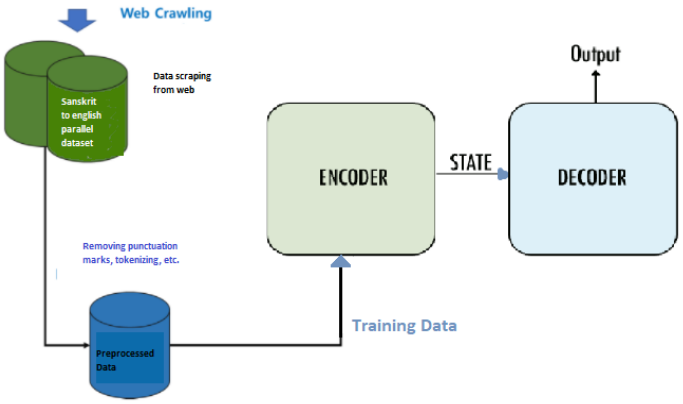

Figure 1. Flow diagram of project

A. Dataset

The first dataset is scraped from http://www.sanskritbible.in/. It includes a complete verse-by-verse translation of the Christian Bible into Sanskrit. As JSON objects, we may query the text pairs (Sanskrit Verse, English Verse). We converted the different languages and verses into their own CSV files after we received this query, so we could compare both languages side by side .Also https://www.holy-bhagavad-gita.org and , https://www.valmiki.iitk.ac.in have Sanskrit to English translations that were scrapped and collected in a database.

The Sanskrit - English corpus is randomly split into training set having 80% data, validation set having 10%, and test set having 10%. The training data contains 24000 Sanskrit to English parallel corpus data sentences, while the validation and test subsets each comprise 6000 parallel corpus data sentences. BLEU measure was used (“Bilingual Evaluation Understudy") to compare the quality of machine translations to supplied target translated sentences in dataset. This metric is extensively used has been shown to correlate well with human assessments of translation quality. We adhere to the most common standard and assign a score between 0 and 100, with 0 being the lowest quality (no match with the candidate translations) and 100 representing the best quality. Sanskrit vocabulary contains 3197 unique tokens and English vocabulary contains 3353 unique tokens.

B. Data cleaning

An NLP model requires large datasets and vocabulary which tells the set of unique words used in the text corpus. It was observed that ASCII did not have all Sanskrit characters for that purpose; unicode representation was used. Then vocabulary was built.

Adding start of sequence and end of sequence tags at the beginning and end of input sequences as this helps the model the encoder ans decoders to know when to start and when to end sequences.This is required as model doesnt know how long the output sequences should be. Also different tokenizers were implemented like tokenization of sentences using split function at spaces, byte pair encoding tokenizers.

Other important preprocessing tasks included removal of punctuation marks, stripping extra spaces, lowering text and most importantly removal of large sentences . Sentences having a number of words greater than 100 were not included in train and test datasets as models like GRU cannot perform well with long sentences. Whereas LSTMs perform quite well with long sentences.

C. Tokenizers

I tried implementing tokenization using various libraries like nltk, inltk, spacy, mosses. INLTK is the library which is developed for asian languages and supports tokenization of sanskrit sentences. I also did tokenization using Byte pair Encoding which gives subwords of words as output of tokenization, which suits the sanskrit language as well.

D. Models implemented

I have implemented two models, one is based on LSTM and other uses GRU architecture. GRU couples forget as well as input gates. GRU requires fewer training parameters and hence less memory than LSTM, and it runs and learns quicker, although LSTM is more accurate on datasets with longer sequences. In summary, LSTM should be used if the sequence is lengthy or precision is critical; otherwise, GRU should be used for decreased memory consumption and faster execution. If you don't have a lot of floating point operations per second to spare, use GRU. LSTM has three output values (output, hidden, and cell), whereas GRU has two output values (output and hidden)
The number of layers and cells required in an LSTM may be determined by numerous factors of the application, including:

1. The dataset's complexity, noise in the dataset, size of dataset, the amount of features, data points, and so on.

2. The accuracy requirement for the application case. This will have a significant impact on the number of memory cells. If the aim is to outperform the state-of-the-art model, more LSTM cells are required.

E. Named Entity Recognition

Named Entity Recognition using a pre-trained BERT model, also using the SpaCy pipeline. This NER module can be later used in the main model for improving accuracy of the model.

IV. Results

Here are some outputs of GRU model:

Sanskrit sentence:
तदानीं यूषफ उत्थाय रजन्यां शिशुं तन्मातरञ्च गृहीत्वा मिसर्देशं प्रति प्रतस्थे,

Translated English Sentence:
Then Jesus answered had spoken to him, and said, I am to the and the and the <EOS>

Some outputs of LSTM model:
Sanskrit sentence:
यीशुस्तत् श्रुत्वा तान् प्रत्यवदत्, निरामयलोकानां चिकित्सकेन प्रयोजनं नास्ति, किन्तु सामयलोकानां प्रयोजनमास्ते।

Translated English Sentence:
But when they heard that Jesus heard that they had not heard that the the to the the the to the the <EOS>

Model	BLEU Score (%)
GRU	18.73
LSTM	15.36

Table showing BLEU accuracies of models implemented

V. Conclusion

I have written about multiple approaches utilized in the implementation of neural machine translation. The resultant conclusion was that LSTM neural machine translation has better BLEU scores than GRU approach for less data. Also statistical and neural machine translation has replaced older methods like rule based ,example based and corpus based machine translation.

VI. References

[1] Promila Bahadur, A.K.Jain, D.S.Chauhan Etrans- A Complete Framework For English To Sanskrit Machine Translation

[2] Sanskrit Machine Translation Systems: A Comparative Analysis Jaideepsinh K. Raulji, Jatinderkumar R. Saini

[3] K. S. Gilda S. S. Dixit S. V. Narote General Structure Of Sanskrit Machine Translation System

[4] Shashank Saxena And Raghav Agrawal Sanskrit As A Programming Language And Natural Language Processing

[5] Abdulmohsen Al-thubaity , Atheer Alkhalifa, Abdulrahman Almuhareb , And Waleed Alsanie Arabic Diacritization Using Bidirectional Long Short-term Memory Neural Networks With Conditional Random Fields

[6] Bensalah Nouhaila , Ayad Habib , Adib Abdellah , And Ibn El Farouk Abdelhamid Lstm Vs. Gru For Arabic Machine Translation

[7] Quang-phuoc Nguyen , Anh-dung Vo , Joon-choul Shin , And Cheol-young Ock Effect Of Word Sense Disambiguation On Neural Machine Translation: A Case Study In Korean

[8] Raj Dabre, Chenhui Chu, Anoop Kunchukuttan A Survey Of Multilingual Neural Machine Translation

[9] Jade Z. Abbott , Laura Martinus Towards Neural Machine Translation For African Languages

[10] Shuoheng Yang, Yuxin Wang, Xiaowen Chu A Survey Of Deep Learning Techniques For Neural Machine Translation

[11] Joseph Mickole James,Sangeetha Jamal Named Entity Recognition In Sanskrit: A Survey

[12] Bogdan Babych, Anthony Hartley Improving Machine Translation Quality With Automatic Named Entity Recognition

[13] Wahab Khan , Ali Daud, Khairullah Khan, Jamal Abdul Nasir, Mohammed Basheri , Naif Aljohani, , And Fahd Saleh Alotaibi3 Part Of Speech Tagging In Urdu: Comparison Of Machine And Deep Learning Approaches

[14] Arabic–chinese Neural Machine Translation:

[15] Fares Aqlan, Xiaoping Fan 1,2, Abdullah Alqwbani, And Akram Al-mansoub3 Romanized Arabic As Subword Unit For Arabic-sourced Translation

[16] Machine Translation Development For

[17] Amruta Godase And Sharvari Govilkar Indian Languages And Its Approaches

[18] Rico Sennrich And Barry Haddow And Alexandra Birch Neural Machine Translation Of Rare Words With Subword Units

[19] Mat ¨iss Rikters Impact Of Corpora Quality On Neural Machine Translation

[20] Ivan Provilkov, Dmitrii Emelianenko,Elena Voita Bpe-dropout: Simple And Effective Subword Regularization

[21] Antonio Valerio Miceli Barone Barry Haddow

[22] Ulrich Germann Rico Sennrich Regularization Techniques For Fine-tuning In Neural Machine Translation

[23] Ancient Korean Neural Machine Translation Chanjun Park , Chanhee Lee, Yeongwook Yang , And Heuiseok Lim .

[24] Whelan, T., Johannsson, H., Kaess, M., Leonard, J. J., & Mcdonald, J. (2013). Robust Real-time Visual Odometry For Dense Rgb-d Mapping. In Proceedings - Ieee International Conference On Robotics And Automation (Pp. 5724–5731).

[25] Raj Dabre, Chenhui Chu,Anoop Kunchukuttan, A Survey Of Multilingual Neural Machine Translation

Link for Plagiarism report:

https://drive.google.com/file/d/1Zb92ILjl05kCG9LLw5Ulfk7swTieWxaL/view?usp=sharing