Informal to Formal Text Conversion (Telugu)

By

Mancha Vinay
(39/CSE/17046/244)

Chintam Ravi Chandra
(39/CSE/17029/227)

Khasha Pavan Kalyan Raju
(39/CSE/17043/241)

Mentored By
Dr. Sanjay Chatterji

Indian Institute of Information Technology Kalyani

Bachelor of Technology

In

Computer Science and Engineering

Project Code - (CS-614)

Abstract

Code-mixing, use of two or more languages in a single sentence is found everywhere, generated by multi-lingual speakers across the world. The phenomenon presents itself prominently in social media discourse. Consequently, there is a growing need for translating code-mixed hybrid language into standard languages. However, due to the lack of gold parallel data, existing machine translation systems fail to properly translate code-mixed text. In an effort to initiate the task of machine translation of code-mixed content, we present a newly created parallel corpus of code-mixed English-Telugu and Telugu. With the help of the created parallel corpus, we analyzed the structure of English-Telugu code-mixed data and present a technique seq2seq encoder decoder model with attention for machine translation (MT) approaches that can help achieve superior translations of code mixed hybrid language into standard language.

Introduction:

Code mixed sentences are sentences which contain two or more languages or even scripts within them, these kind of sentences are mostly used in the current social media. To be more clear lets see how code mixed sentences are used in the social media, people use english script to simulate their own language along with some native English words which may be also accompained by words written in their own script. Telugu is a language which is mostly used by the people of Andhra Pradesh and Telangana,
it is derived from sanskrit the same as hindi and many other languages. In this project we translate code mixed English-Telugu sentences completely into a Telugu sentence.

We can leave them as they are but there are still many people who don't understand english, some of them can't even read it and it will be easier for other NLP tasks(sentiment analysis, etc.,).

But designing a model to do this stuff is challenging, because there's lack of data to train the model with a reliable accuracy and also the data available to train contains too much noise in it.

Related Work

Several research works has been done in the recent past on code-mixed data, and especially involving language tagging. Jhamtani et al. (2014) created an ensemble model by combining two classifiers to create a Hindi-English code-mixed LID. The first classifier used word frequency, modified edit distance, and character n-grams as features. The second classifier used the output from the former classifier for the current word, along with language and POS tag of neighbouring words to give the final tag. Rijhwani et al. (2017) proposed a generalized language tagger for arbitrary set of languages which is fully unsupervised. With respect to back-transliteration, Bilac and Tanaka (2004) proposed a hybrid approach which combines phoneme, grapheme and segmentation based modules. Luo and Lepage (2015) presented an architecture for back transliteration using an SMT framework described in (Franz et al., 2003). Ravishankar (2017) describes a finite-state based system for back-transliteration of transliterated Marathi words in Roman. The major advantage over statistical models is that its able to model exceptions without being retrained. Sinha and Thakur (2005) took the challenge of translation of Hindi-English code-mixed to English monolingual from a linguistics point of view by using morphological analyzers though they did not perform any in depth analysis or evaluations. In (Dhar et al., 2018), the authors created a code-mixed (Hindi-English) to monolingual (English) parallel corpus consisting of 6096 instances. They also developed an augmentation pipeline which can be utilized for augmenting existing MT systems such that the translation of the systems can be improved without training the MT system specifically for code-mixed text. On

testing the module with Moses, Google NMTS and Bing translator, the BLEU scores improved by 2%, 9.4% and 6.1% respectively.

Procedure

- **Data Collection**

We prepared a dataset of chats which are exported from WhatsApp. It contains around 30k sentences extracted from 7 individual chats. These sentences are of Code Mixed words which are mixed words of English, Telugu in English and Telugu in UTF format. We removed the special characters embedded within.

- **Data Preparation**

The collected data is processed and prepared by the following steps:

- The English words are identified by using a bag of words (around 57000 English words) and are translated to their respective Telugu UTF format using Google translation.
- The words which are not in English are identified as Telugu words and are transliterated to their respective Telugu UTF format using Google transliterate api.
- We removed the NULL, empty sentences, Media omitted tags etc., from the resulted data.
- The Data Obtained is further cleaned manually which resulted in size of 31966.

Data:

https://www.kaggle.com/dataset/dfcc693efdfa57062c373565adfd9174551befeb3102711b710d5ec2cd7fbaf2

hi	హాయ్
who is this	ఎవరిది
guess me	నాకు ఊహించడం
ragunadh ena	రఘునాధ్ ఎన
kaadhu	కాదు
i am ravi	నేను రవి
ravi chandra	రవి చంద్ర
ade ragunadh ye ga	అదే రఘునాధ్ యే గ
le	లే
peru maripoyindhi	పెరు మారిపోయింది
😕	😕
😐	😐
em peeking	ఎమ్ త్వరిత వీక్షణ

Sample Data

- **Data Pre-processing**

The cleaned data is pre-processed by following steps:

- We add the SOS and EOS tokens to each sentence as follows:

  sentence = "[SOS] " +sentence +" [EOS]"
- We generated two tokenizers, one for Input language (Code-Mixed Sentences) and one for target language (Converted Telugu sentences).
- We tokenized the input sentences and target sentences.
- We further padded the sentences such that their length is the same.

- This Data is then split into Train Data and Validation Data with 15% of Validation data.

- **Model Architecture**

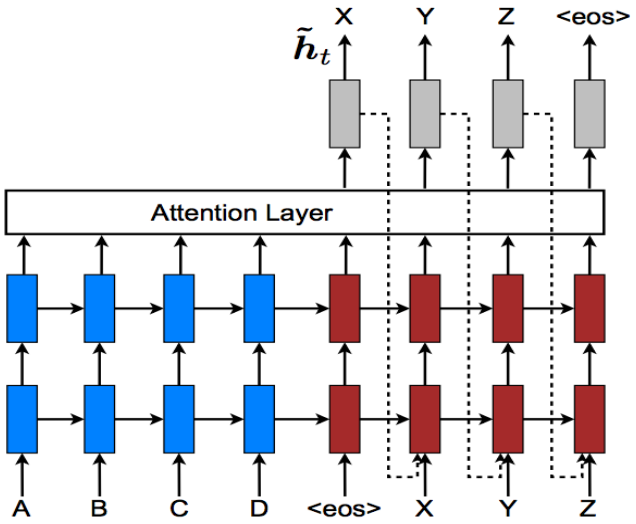
We have used an Encoder-Decoder model with attention to train our Data.

The encoder-decoder architecture for recurrent neural networks is proving to be powerful on a host of sequence-to-sequence prediction problems in the field of natural language processing such as machine translation

- **Encoder**: The encoder is responsible for stepping through the input time steps and encoding the entire sequence into a fixed length vector called a context vector.
  In this model we have initialized the encoder with GRU(Gated Recurrent Unit) layer of 1024 units with batch size of 64

- **Decoder**: The decoder is responsible for stepping through the output time steps while reading from the context vector
  We have initialized the decoder with GRU(Gated Recurrent Unit) layer of 1024 units with batch size of 64 followed by a dense layer and Bahdanau Attention

We have used adam optimizer and Sparse Categorical Crossentropy for loss.

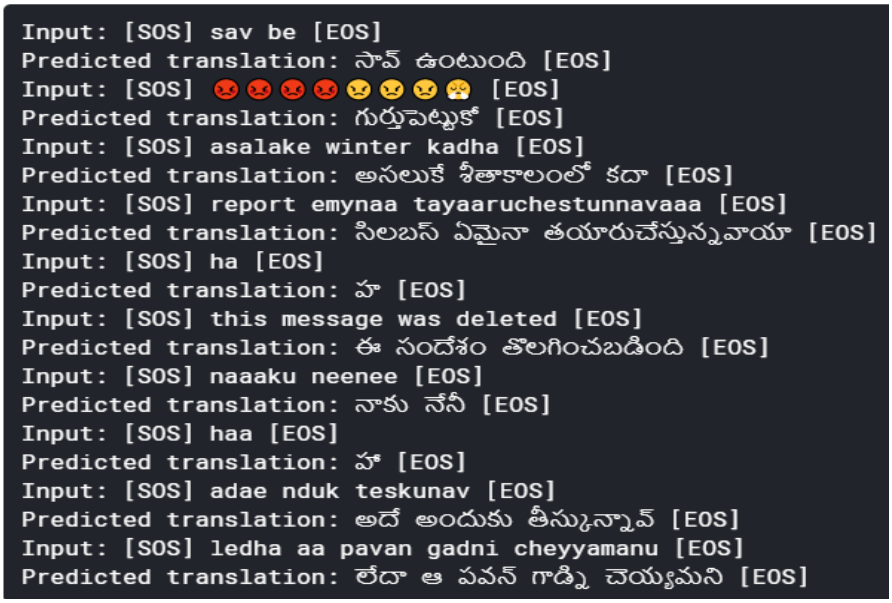


- **Training**

1. Pass the *input* through the *encoder* which return *encoder output* and the *encoder hidden state*.
2. The encoder output, encoder hidden state and the decoder input (which is the *start token*) is passed to the decoder.
3. The decoder returns the *predictions* and the *decoder hidden state*.
4. The decoder hidden state is then passed back into the model and the predictions are used to calculate the loss.
5. We use *teacher forcing* to decide the next input to the decoder.
6. *Teacher forcing* is the technique where the *target word* is passed as the *next input* to the decoder.

7. The final step is to calculate the gradients and apply it to the optimizer and backpropagate.

Link to model: https://www.kaggle.com/vinaymancha/processing-code-mixed



Predictions

Conclusion

In this project we have created a set of 31000 (Approx.) English-Telugu code mixed words and monolingual Telugu gold standard parallel sentences to act as a dataset to our model.

We use an NMT model, because other traditional phrase-based translation models split the source sentences and then translate them phrase by phrase, but using an NMT model is like mimicking the human ability to understand the sentence and translating it.

The NMT model that we use here is the Encoder-Decoder Model which is trained by the created dataset.

Future work

We have achieved translating code-mixed English-Telugu words to their respective Telugu words which are rendered formally. We can take this project further by:

1) Training the Model with more Data and increasing the accuracy of it.
2) Adding different languages to the model i.e. being able to convert languages other than Telugu into the respective selected language.

References

- Google Transliteration API: https://www.google.com/inputtools/try/
- English Bag of Words: http://www.mieliestronk.com/wordlist.html
- Google Translate: https://translate.google.co.in/
- Enocer Decoder with attention: https://www.tensorflow.org/tutorials/text/nmt_with_attention
- Bahdanau attention https://arxiv.org/pdf/1409.0473.pdf