CoMeT: Towards Code-Mixed Translation Using Parallel Monolingual Sentences

Team: Amigos (2020101103, 2020102061, 2020101077)

What is Code Mixed Languages?

Code-mixing is the mixing of two or more languages or language varieties in speech. In other words, it is the embedding of linguistic units such as phrases, words, and morphemes of one language into an utterance of another language.

- In a Multilingual society, the increase in the population and increased usage of social media resulted in various forms of informal writing. One of them is code-mixed languages.
- From my personal experience, I grew up speaking Telugu and English and have a fair command of Hindi. Even though I usually speak Telugu at home, I always notice other languages tend to find their way into my conversations.

Purpose/Aim:

Generating a machine translation system for English to Hinglish in supervised learning. In supervised learning, the algorithm “learns” from the training dataset by iteratively making predictions on the data and adjusting for the correct answer. (Here we are making use of supervised learning because we are using labelled data so that we can get the entire context at the encoder.)

- The translation systems could be used to augment datasets for various Hinglish tasks by translating datasets from English to Hinglish.

Problem:

The informal nature of this language (code mixed) makes it difficult to collect code mixed data which is a major contributing factor.

- In addition to the use of language, code-switching also involves switching between dialects, styles of speech, gestures, body language, and vocal registers. So it's difficult to collect data.

Another challenge with Hinglish, in particular, is that there is no standard system of transliteration for Hindi words.

Solution:

Before heading towards the solution let's get a complete idea of BART. It is similar to that of transformers as it includes encoders and decoders. BERT (Bidirectional Encoder Representations from transformers), Random tokens are replaced with masks, and the document is encoded bidirectionally missing tokens are predicted independently. So BERT cannot easily be used for generations. In mBART, we make use of two models, the Masked language model and the Next Sentence Prediction.

Implementation:

- Pre-train the BERT to understand the language.
- Fine-tune BERT to learn a specific task. (Here in our model, that specific task is to translate to Hinglish).
- So, we first fine-tune mBART by first transliterating the Hindi words in target sentences from Roman script to Devanagari scripts to utilize its pre-training. (we make use of Roman scripts because it helps the encoder with more contextual analysis). Then we further translate the English input to Hindi using pre-existing models and show improvements in the translation using parallel sentences as input to the mBART model.

Dataset:

We are using the dataset from the following link:
https://ritual.uh.edu/lince/datasets
This dataset contains English-Hinglish.
Ex: Not very popular it seems; lagta hai bahut popular nahi hai

- In mBART-hien, English sentences and translated Hindi sentences. So, as covered before we use NMT for that.

Statistics of the dataset

	Train (From the train.txt)	**Valid (From the dev.txt)**
# of sentences	8060	942
# of tokens in source sentences	98080	12275
# of tokens in target sentences	101752	12611
# of Hindi tokens in target sentences	68054	8310
# of English tokens in target sentences	21502	2767
# of other tokens in target sentences	12196	1534

Evaluation Metrics:

How to evaluate the performance of the Machine Translation system? The only solution is human evaluation. But the problem is that this is a time taking process. The optimized way is using BLEU.

BLEU: Bi-Lingual Evaluation Understudy

It compares a machine translation with existing human-generated translation (reference translations).

- Calculation: BLEU is calculated based on precision. The ratio of the covered candidate words out of the Total number of candidate words. Then we sum that for the entire sentence i.e., BLEU* = (D(w1) + D(w2) + D(w3) + ... +D(wn))/n
- In coding, we are using SacreBLEU library to calculate BLEU score. https://github.com/mjpost/sacrebleu

BLEU Normalized:

Similar to the calculating process of the BLEU as above but the only difference is calculating the scores on texts where Hindi words are in Devanagari and English words in Roman. But whereas in the above BLEU we calculate the scores on the texts where the Hindi words are transliterated to Roman.

- We are going to use these two evaluation metrics for comparing our systems.

How to run the code:

Firstly we will run the training part and then fine tune the model and later move on to the evals part. We need to make sure we are in the right environment when we run the code. The environments used are:

cmtranslation2: While training the data

indictrans: While evals