Milestone 3

CS 6320.002: Natural Language Processing
Siddhartha Sahai - sxs180170
Manas Bundele - mmb180005
Randeep Ahlawat - rsa190000

INSTRUCTIONS

1. Run setup.sh: ./setup.sh
2. Copy all the 3 files in to_replace/ folder to this path (after setup.sh): cp to_replace/* /usr/local/lib/python3.7/site-packages/litcm/
3. Run baseline.py: python3 baseline.py
4. Run evaluate.py: python3 evaluate.py

Writeup Question 1.1:

Write 1-2 paragraphs describing your chosen baseline. What paper was it published in? What machine learning algorithm does it use? What kind of preprocessing does it require? What are the features?

Baseline paper - "Enabling Code-Mixed Translation: Parallel Corpus Creation and MT Augmentation Approach" published in "Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing" - Santa Fe, New Mexico, USA, August 2018.

It compares 3 different models - Moses statistical machine translation system, google translate and Bing translator.
It requires the data to be preprocessed by tagging using the Indic-Trans library. It requires the litcm module - "Language Identification and Transliteration system for Indian Languages" for word-level tagging of languages. The features involved are the multi-language tags for each word in the dataset.

Writeup Question 2.1:

Write 1-2 paragraphs describing the training process. What learning scheme did you use (unsupervised, semi-supervised, or supervised)? Where there any hyperparameters, and if so, how did you tune them? Did you run on a CPU or GPU? How long did it take?

Our baseline model consists of an augmentation module as a preprocessing step to Moses, Google NMTS and Bing Translator.

Augmentation module:

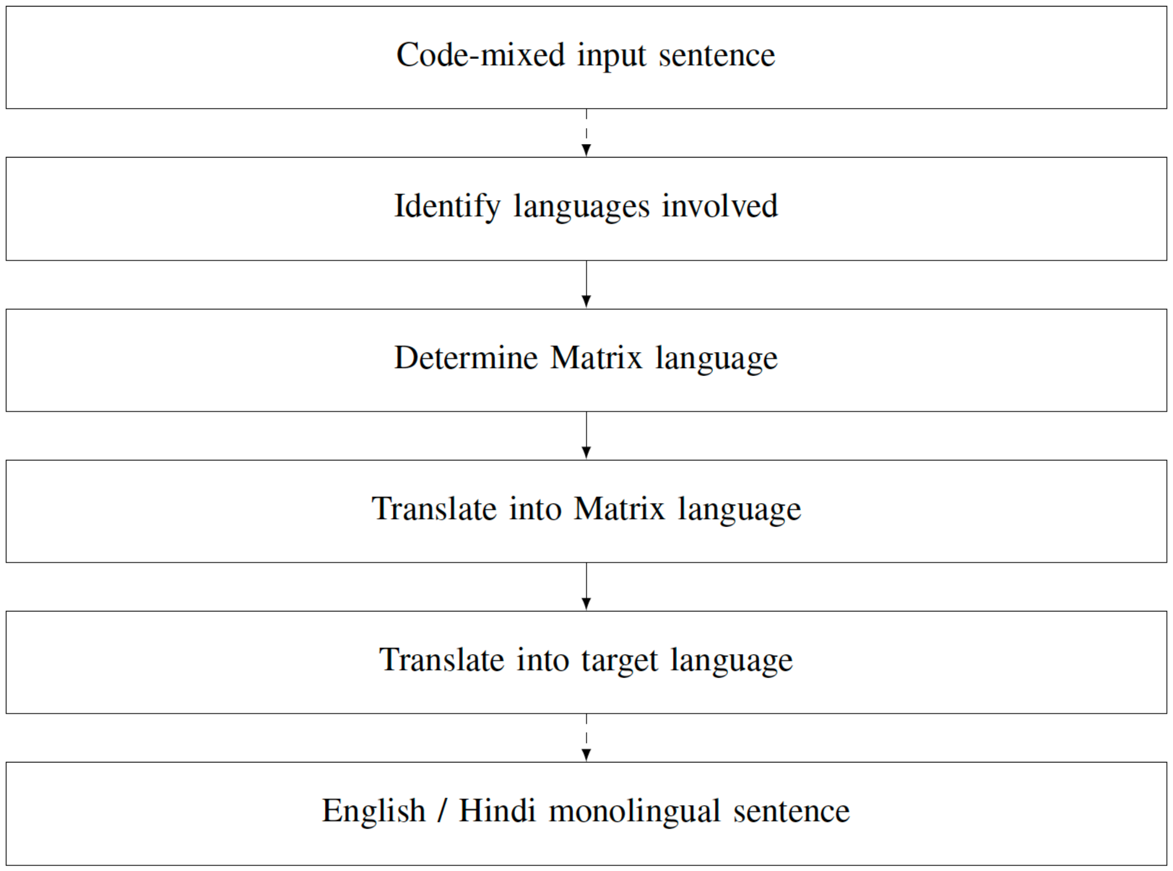
Code-mixed input sentence
Identify languages involved
Determine Matrix language
Translate into Matrix language
Translate into target language
English / Hindi monolingual sentence

Figure 1: Translation augmentation pipeline

Basically, apart from the dataset creation, the contribution of the paper was adding the augmentation module as a preprocessing step to improve the translation results of Moses, GNMTS and Bing translator.

Translation model: After this, we trained Moses on Hindi English parallel corpus by English-Hindi parallel corpus released by Kunchukuttan et al. (2017) as given in baseline. This dataset consists of 1,492,827 parallel monolingual English and monolingual Hindi sentences. The Hindi sentences were in the Devanagari script, and required pre-processing for use with our code-mixed dataset, which is entirely in Roman script using Language Identification and Transliteration in Code Mixing (litcm). Since the training model was of very large size(around 3gb), the system hung up while tuning the hyperparameters and we could not tune the parameters. So, we used the Google NMTS apis using the 'google trans' module in python. Although google blacklists an IP if continuous api calls are made(recognizing as a bot), we made api calls in batches of 300 at regular intervals to translate the phrases(to convert to Matrix Language) and final sentences(from Matrix language to Target Language using Google NMTS).

As the api calls are still highly unreliable, we have provided the output files that will be read directly using code instead of making api calls and causing the program to fail. We skipped Bing translator using apis as api calls are highly unreliable(as per our experience with GNMTS).

The details in the baseline are pretty vague in terms of implementation and hence our results differ alot from the baseline as it highly depends on the api results from Google NMTS.

Writeup Question 3.1:

Give step-by-step instructions for how to run evaluate.py. What arguments does it take?

We may run "python3 evaluate.py" to run the google nmts system. It does not require any arguments.

Writeup Question 3.2:

Write 1-2 paragraphs describing the evaluation. What was the performance of your baseline system? Is that good or bad? Why is the metric appropriate for your task? If there are other metrics commonly used for the task, why did you choose this one?

We used the BLEU metric as the target. The performance of our baseline system was around 25 BLEU. It's good compared to the result in the paper which was around 16 BLEU for google nmts. We think the performance of google nmts may have increased since this paper was published.
Baseline also uses Word Error Rate (WER) and Translation Error Rate (TER) as metrics..
BLEU is an appropriate metric for our task because it measures the similarity between a reference and a hypothesis sentence. For the task of neural machine translation, there can be no perfect sentence - several sentences may express the same meaning hence BLEU is a proven heuristic method which utilizes ngram similarities and other heuristic features. Other commonly used metrics include POS tag accuracy. We chose BLEU as it checks ngram precision. We also plan to use METEOR as it is the harmonic mean of precision & recall.