Gui at MixMT 2022 : English-Hinglish : An MT approach for translation of code mixed data

Akshat Gahoi Jayant Duneja Anshul Padhi Shivam Mangale
Saransh Rajput Tanvi Kamble Dipti Misra Sharma Vasudeva Varma

International Institute of Information Technology, Hyderabad

{akshat.gahoi,anshul.padhi,saransh.rajput,tanvi.kamble}@research.iiit.ac.in
{dunejajayant,shivammangale}@gmail.com

Abstract

Code-mixed machine translation has become an important task in multilingual communities and extending the task of machine translation to code mixed data has become a common task for these languages. In the shared tasks of WMT 2022, we try to tackle the same for both English + Hindi to Hinglish and Hinglish to English. The first task dealt with both Roman and Devanagari script as we had monolingual data in both English and Hindi whereas the second task only had data in Roman script. To our knowledge, we achieved one of the top ROUGE-L and WER scores for the first task of Monolingual to Code-Mixed machine translation. In this paper, we discuss the use of mBART with some special pre-processing and post-processing (transliteration from Devanagari to Roman) for the first task in detail and the experiments that we performed for the second task of translating code-mixed Hinglish to monolingual English.

1 Introduction

Code Mixing occurs when a multi-lingual individual uses two or more languages while communicating with others. It is the most natural form of conversation for multilinguals. It is often confused with code-switching but there is a slight difference between the two. Both these phenomena include communicating in multiple languages but code switching usually takes place within multiple sentences while code mixing usually refers to words of different languages used in the same sentence. In code mixing, phrases, words and morphemes of one language may be embedded within an utterance of another language. Code mixing is extensively observed on social media sites like Facebook and twitter. With the rapid growth of social media and consequently, increase in the use of code-mixed data, it becomes important to develop systems to process such text. Machine Translation, also known as automated translation, is the process where a software translates text from one language to another without any human involvement. There are multiple forms of machine translation, however, over the past few years, neural machine translation has become extremely popular. The WMT shared task had two subtasks. The first subtask consisted of the translation of Hindi-English parallel sentence pairs to Hindi-English code mixed sentences through machine translation. The second subtask consisted of the translation of Hindi-English code mixed sentences to English.

2 Background

While there is a growing interest in code-mixed text analysis as a research problem, there is one bottleneck that has hindered the growth of such works, and that is the lack of data. Due to this, there aren't many robust models for code-mixed text. To build standardized datasets of code-mixed text, we need to come up with ways of text generation of these code-mixed texts. These texts would be very helpful in training language models for various code-mixed pairs as language models only need unsupervised data.

Code Mixed text generation is a relatively new problem, and so is its initial stage. One of the recent works in this field (Rizvi et al., 2021) tried to use linguistic theories to synthetically build code-mixed text using parallel monolingual corpora of two languages. The Equivalence Constraint Theory (Poplack, 1980) says that code-mixing can only occur at parts of the text where the surface structures of two languages map onto each other. So in these parts, the grammatical rules of both languages are followed. The Matrix Language Theory (McClure, 1995) tries to solve this problem by separating the two languages into a base language and a second language. The grammatical rules of the base lan-

Proceedings of the Seventh Conference on Machine Translation (WMT), pages 1126 - 1130
December 7-8, 2022 ©2022 Association for Computational Linguistics

guage are followed, and parts of the base language are replaced by the corresponding parts of the second language whenever it is grammatically feasible to do so.

Deep Learning and Neural Networks have also been used to build systems for code mixed generation. In these systems, the problem of text generation has been posed as one of machine translation, where monolingual text is translated to code-mixed text. Some of the early work involved using the then state of the art encoder-decoder models like pointer generator networks(Winata et al., 2019) and GANs(Chang et al., 2019) to translate two sets of monolingual corpora into code mixed text. With the rise of multilingual models like mT5 (Xue et al., 2020), mBART, indicBART (Dabre et al., 2021), etc. the task of translation has become much easier as these models understand both languages and this has been shown to outperform previous models in many workshops.

mBART(Liu et al., 2020) is a denoising autoencoder which has been trained on a very large dataset which contains text from 25 languages. It has the same transformer based architecture and training objective as BART, a denoising autoencoder which was shown to be one of the best performing sequence to sequence models at the time. It has been trained to reconstruct original text which has been corrupted as a way to add noise. It can perform various downstream sequence to sequence tasks like machine translation, text summarization, etc. mBART consists of 12 encoder layers and 12 decoder layers. There are 16 heads and a model dimension of 1024.

Another solution to circumvent the data problem is to create translation systems that can translate code-mixed text to monolingual text. This allows us to use robust NLP systems for various downstream tasks.

While we have the above said top performing models at the moment, they are very heavy computational wise due to their large parameter sizes. With resource constraints, it is tough to replicate their performance. Helsinki's OPUS-MT (Tiedemann and Thottingal, 2020) model was of comparabaly smaller size and focused on the initiative of supporting low-resource languages. It does accordingly have lower performance. We have attempted at utilizing this model in our case with further training on provided data to understand whether under the resource constraints, we can observe competitive performance.

The model architecture is based on a standard transformer setup with 6 self-attentive layers in the encoder and decoder network. It has 8 attention heads in each layer. This is hence comparatively low compute seeking as compared to the mainstream models.

Data	Length
Synthetic (Train)	3263
Synthetic (Validate)	396
Human Generated (Train)	1800
Human Generated (Validate)	376

Table 1: Distribution of Sentences in the data

3 System Overview

3.1 Task 1

In this section we propose our system for Task 1 which is English and Hindi to code-mixed text translation

3.1.1 Dataset and Data Preparation

The dataset that we used for Task 1 was the HinGE dataset (Srivastava and Singh, 2021). It is divide into two parts, the synthetic dataset or the machine generated dataset and the human generated dataset. (Table 1) There were 3659 and 2176 sentences respectively.

3.1.2 Model

In this task we finetune the mBART model on the data given to us. Since mBART is a very large model we needed to decrease its size. We do this by reducing the vocabulary of the model as the vocabulary adds to the model size by a lot and we don't need the vocabulary from the rest of the 25 languages. To reduce the vocabulary we create our own vocabulary using the tokens present in the task dataset, IIT-B English-Hindi parallel corpus (Kunchukuttan et al., 2018) and the Dakshina Dataset (Roark et al., 2020) as we feel the two datasets were large enough to create a vocabulary extensive enough to solve the given task. We process the input data from the given task data as explained above to create our input. Using the corresponding code-mixed sentences as the gold output we finetune the mBART model.

3.1.3 Post Processing

The output of our model was in a mixed script (Roman + Devanagari). So the post processing

Post Processing	**ROUGE-L**	**WER**
Normal Output	0.39091	0.81884
With Automated Transliteration	0.48376	0.72561
With Automated Transliteration + Dictionary Based Transliteration	**0.61667**	**0.63342**

Table 2: ROUGE-L and WER scores after different post processing tasks

becomes one of the important step in this task as we wanted our output Hinglish sentences to be only in Roman script. We used transliteration function from indicate library as our first step to see how good the results will be. There were many instances where the transliteration done by indicate was not accurate. So the next step that we did was to create a dictionary of most common words and numbers with their corresponding transliterated Roman text. This dictionary over the automatic transliteration by indicate was used to get the best output of our model in the Roman script.

3.2 Task 2

In this section we propose our system for Task 2 which is Hinglish (code-mixed) to English text translation.

3.2.1 Observations

The data for task 2 are tweets based data. Due to the tweets nature, we observed that:

- The URLs included tended to be at the end of the sentences.
- The mentions (of the form '@<some_user_tag>' for instance @LokSabha) at the beginning and the end of tweets are generally such that the sentences can be translated without them with no-low loss of information.
- Hashtags which are added at the end of the tweets are generally for increasing outreach and exposure.

Based on the above observations, we found that the information provided by these tokens to the translation was not significant as compared to the loss of information due to incorrect translation of these units. Hence, we applied heuristics to appropriately preprocess the input data to exclusively and exhaustively split the tweets into sentences (which will be translated), URLs, mentions and hashtags, which are then concatenated after the translation in postprocessing.

3.2.2 Dataset and Data Preparation

The dataset that we used for Task 2 was the PHINC dataset (Srivastava and Singh, 2020). It contains 13,738 parallel sentences in Hinglish (code-mixed) and English of which we used a train-val-test split of 80-10-10. We transliterated the Hinglish sentences from the Roman script to the Devanagiri script using the Google Transliterate API, to utilise pre-trained Hindi to English translation models. This transliterator was used among others due to it having one of the best performance, it's similarity in the vocabulary space with the input dataset as compared to the other transliterators available and also that PHINC was jointly created using Google Translate.

3.2.3 Model

Due to compute constraints, we decided to utilize pretrained models, that would be efficient for our dataset. To access better models, we went ahead with models trained with a task or a subtask of Hindi to English machine translation. We appropriately processed the data for the same. We hence decided to finetune Salesken.AI's pretrained model provided on Huggingface Transformers. They have finetuned Helsinki's OPUS-MT model on AI4Bharat's Samanantar dataset (Ramesh et al., 2021), a large indic dataset.

4 Experimental Setup

In task 1, we use the fairseq implementation of mBART as our base model which has been trained on 4 Nvidia GeForce RTX 2080 Ti GPUs. The model has been trained using label smoothed cross entropy as the loss criterion. The model uses an Adam optimizer with polynomial decay learning rate scheduling, dropout = 0.3, learning rate = $3 * 10^{-5}$, $\epsilon = 10^{-6}$, $\beta_1 = 0.9$ and $\beta_2 = 0.98$.

The model was trained on 10000 steps with 2500 warm up steps and a batch size of 512 tokens.

We validate the model on each epoch on a validation set and at the end we select the model with the lowest loss.

In task2, for fine-tuning we use the Salesken.AI's pretrained model provided on Huggingface Transformers. The model was trained on Nvidia GeForce RTX 2080 Ti GPUs.The model has been trained using label smoothed cross entropy as the loss criterion. The model uses an Adam optimizer with learning rate = $3 * 10^{-4}$, $\epsilon = 10^{-9}$, $\beta_1 = 0.9$ and $\beta_2 = 0.98$.

5 Results and Evaluation

The test dataset consisted of 500 sentences. These sentences also had both English sentence and its corresponding Hindi sentence. ROUGE-L score and WER score was considered for evaluation. ROUGE-L score considers longest common subsequence for its scoring. It counts the longest subsequence which is shared between both reference and the output. Its different from precision as it only counts the ratio between longest subsequence matched and the number of words matched. It does not take all the words in the reference.

The WER score represents the word error rate. Total errors between the reference and output is considered for this score. It adds up all the substitution, addition and deletion required to convert the output to the reference sentence and treat it as total error of the output. It can be treated same as calculating Levenshtein distance.

So our aim was to maximise ROUGE-L score and minimize WER score. Our score improved as we translitered the output from Devanagari to Roman using indicate library. The score increase significantly after we created a dictionary of words for transliterating most common Hindi words and numbers. We achieved a ROUGE-L score of 0.61667 and WER score of 0.63342 after both the post processing steps.

The test set provided for Task 2 contained 1500 lines, which were processed as mentioned in 5 The results for the evaluation metrics we obtained for the test set provided for Task 2 is available in 3. Using the Google Transliterate API significantly improved the quality of the input data, and also the similarity of vocabulary with the dataset as mentioned earlier. The application of heuristics also bolstered the approach's performance.

Based on qualitative evaluation, it was observed that it struggled to get long sentence translations which can be attributed to the source of the dataset being of of tweets which have a noisy and inconsistent structure. This is alongside the lower parameter size and attention heads.

Metric	Score
ROUGE-L	0.41493
WER	0.80804

Table 3: Results for Task 2

The model was trained till significant learning on a wide array of parameters, till resource permits, in an attempt to provide more opportunities to appropriately fine-tune the model, but even though there was a sign of the model learning, the performance was observed to be not competitive to the current top performers.

6 Conclusion

In this paper, we approached code mixed machine translation problem from both the direction. We used mBART for our first task of translating English and corresponding Hindi sentences to Hinglish sentence. The results were significantly improved through transliterating the output from Devanagari script to Roman script. Two different methods were used for the same. Our model surpassed baseline in ROUGE-L and WER scores by a huge margin.

For the second task of translating Hinglish sentences to English sentence by fine-tuning Salesken.AI's pre trained model. We cleared the baseline but their is still work to be done in that field as we think that it can be further improved. For the future work in this area we would like to work further on the second task in hand of translating codemixed language to a monolingual language. We need to retrieve information about both the languages from the code mixed sentence and try to give a output in a mono lingual langauge without disturbing the word order.

References

Ching-Ting Chang, Shun-Po Chuang, and Hung-Yi Lee. 2019. Code-Switching Sentence Generation by Generative Adversarial Networks and its Application to Data Augmentation. In *Proc. Interspeech 2019*, pages 554–558.

Raj Dabre, Himani Shrotriya, Anoop Kunchukuttan, Ratish Puduppully, Mitesh M. Khapra, and Pratyush Kumar. 2021. Indicbart: A pre-trained model for natural language generation of indic languages.

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2018. The IIT Bombay English-Hindi parallel corpus. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation.

Erica McClure. 1995. Duelling languages: Grammatical structure in codeswitching. carol myers-scotton. oxford: Clarendon press, 1993. pp. xiv 263. *Studies in Second Language Acquisition*, 17(1):117–118.

Shana Poplack. 1980. Sometimes i'll start a sentence in spanish y termino en espaÑol: toward a typology of code-switching 1. *Linguistics*, 18:581–618.

Gowtham Ramesh, Sumanth Doddapaneni, Aravinth Bheemaraj, Mayank Jobanputra, Raghavan AK, Ajitesh Sharma, Sujit Sahoo, Harshita Diddee, Mahalakshmi J, Divyanshu Kakwani, Navneet Kumar, Aswin Pradeep, Srihari Nagaraj, Kumar Deepak, Vivek Raghavan, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh Shantadevi Khapra. 2021. Samanantar: The largest publicly available parallel corpora collection for 11 indic languages.

Mohd Sanad Zaki Rizvi, Anirudh Srinivasan, Tanuja Ganu, Monojit Choudhury, and Sunayana Sitaram. 2021. GCM: A toolkit for generating synthetic code-mixed text. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 205–211, Online. Association for Computational Linguistics.

Brian Roark, Lawrence Wolf-Sonkin, Christo Kirov, Sabrina J. Mielke, Cibu Johny, Isin Demirsahin, and Keith Hall. 2020. Processing south asian languages written in the latin script: the dakshina dataset.

Vivek Srivastava and Mayank Singh. 2020. PHINC: A parallel Hinglish social media code-mixed corpus for machine translation. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 41–49, Online. Association for Computational Linguistics.

Vivek Srivastava and Mayank Singh. 2021. Hinge: A dataset for generation and evaluation of code-mixed hinglish text. pages 200–208.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT — Building open translation services for the World. In *Proceedings of the 22nd Annual Conferenec of the European Association for Machine Translation (EAMT)*, Lisbon, Portugal.

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2019. Code-switched language models using neural based synthetic data from parallel sentences. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 271–280, Hong Kong, China. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer.