Enhancing Code-mixed Text Generation Using Synthetic Data Filtering in Neural Machine Translation

Dama Sravani
Language Technologies Research Centre, IIIT Hyderabad
dama.sravani@research.iiit.ac.in

Radhika Mamidi
Language Technologies Research Centre, IIIT Hyderabad
radhika.mamidi@iiit.ac.in

Abstract

Code-Mixing[1], the act of mixing two or more languages, is a common communicative phenomenon in multi-lingual societies. The lack of quality in code-mixed data is a bottleneck for NLP systems. On the other hand, Monolingual systems perform well due to ample high-quality data. To bridge the gap, creating coherent translations of monolingual sentences to their code-mixed counterparts can improve accuracy in code-mixed settings for NLP downstream tasks. In this paper, we propose a neural machine translation approach to generate high-quality code-mixed sentences by leveraging human judgements. We train filters based on human judgements to identify natural code-mixed sentences from a larger synthetically generated code-mixed corpus, resulting in a three-way silver parallel corpus between monolingual English, monolingual Indian language and code-mixed English with an Indian language. Using these corpora, we fine-tune multi-lingual encoder-decoder models *viz*, mT5 and mBART, for the translation task. Our results indicate that our approach of using filtered data for training outperforms the current systems for code-mixed generation in Hindi-English. Apart from Hindi-English, the approach performs well when applied to Telugu, a low-resource language, to generate Telugu-English code-mixed sentences.

1 Introduction

Code-mixing (CM) is a phenomenon of mixing two or more languages in an utterance of a speech or text (Bokamba, 1989). This form of communication is prevalent in multi-lingual communities owing to socio and psycho-linguistic reasons. With the advent of social media, code-mixing has become a common phenomenon of communication on social platforms like Facebook, Twitter, Reddit, etc. The extensive use of code-mixing has led to interesting computational multi-lingual NLP research directions.

Linguistic research on code-mixing has proposed multiple theories for generating code-mixed sentences. The Equivalence Constraint (EC) Theory, introduced by (Poplack, 1980), posits that code-switching occurs when there is functional equivalence between the source and target languages, indicating similarity in meaning, pragmatics, or discourse function. The Matrix Language Frame (MLF) theory, proposed by (Myers-Scotton, 1997), suggests that bilingual individuals incorporate words or phrases from a non-dominant language into a dominant language or "matrix language" structure.

Recently, pre-trained models (Liu et al., 2020a; Devlin et al., 2019) have become the state-of-art models for multi-lingual language analysis and generation systems. Availability of large monolingual text corpora from sources like news, Wikipedia, books, has enabled researchers to train large language models at scale. However, building Natural Language Processing (NLP) systems for code-mixed text or speech has become challenging due to its resource poor nature. While code-mixed text is prevalent in various online platforms, such text often co-exists with monolingual data. Thus, identifying code-mixed sentences and building large-scale corpus is challenging. Recently researchers have used multiple approaches to translate between monolingual and their code-mixed counterparts. GCM (Rizvi et al., 2021) proposed an open-source toolkit which leverages EC and MLF theories of code-mixing to generate multiple synthetic code-mixed sentences for a given set of parallel monolingual sentences. However, a limitation of GCM is that the generated code-mixed sentence need not always be a natural sentence.

[1]``Code-Mixing" usually refers to the phenomena of switching between two or more languages within a sentence boundary, and Code-Switching is used to refer to cases where such switching happens at a sentence boundary. In this paper we have used both the terms interchangeably.

Tarunesh et al. (2021) proposed neural machine translation methods to generate a code-mixed sentence given a monolingual input, where the synthetic data is created using clausal substitutions based on MLF theory. All the previously proposed approaches to create synthetic code-mixed data to train machine translation systems have not considered the quality of the synthetic code-mixed data.

In this paper, we propose a novel approach for generating natural code-mixed sentences, by fine-tuning multi-lingual encoder-decoder models. The main focus of the paper is to train these models with good quality code-mixed sentences and their monolingual counterparts. In order to create this silver parallel corpus, we use code-mixed quality filters that are created from the human judgements on minimal gold-standard text.

The main contributions of this paper are summarized as below :

1. In this study we introduce two mechanisms for **Quantitative filtering** of synthetically generated code-mixed texts, leveraging human knowledge.

2. We created a dataset of 3500 manually annotated Telugu-English code-mixed sentences rated for their quality, where each sentence was rated by two annotators to ensure consistency and accuracy of the annotations. We also release parallel test data for English-Telugu, comprising of 1250 samples.

3. We demonstrate the robustness of our proposed approach by applying the generation mechanism on two code-mix language pairs : English-Hindi and English-Telugu (for which there are no prior machine translation resources).

4. Our best model for Hindi-English code-mixed text generation outperforms the (Tarunesh et al., 2021) architecture which is trained on much larger synthetic data[2].

2 Related Work

Recently, various tasks and datasets have been proposed for code-mixed text. Language Identification has been the most popular task in context of computational research for code-mixed text. Code-mixed data comprises of multiple languages, it is essential to identify the language of each segment of text in order to perform appropriate language-specific processing or analysis. Gundapu and Mamidi (2018) proposed various models - Naive Bayes Classifier, Random Forest Classifier, Hidden Markov Model (HMM) and Conditional Random Field (CRF) for Language identification of Telugu-English code-mixed data. Shekhar et al. (2020) proposed a method using LSTM to identity languages in Hindi-English social media text. (Gupta et al., 2021) proposed a Unsupervised Self-training approach for sentiment analysis of code-mixed data. To tackle the problem of scarce annotated code-mixed data this approach used minimal data to start fine-tuning mBART and then use pseudo labels obtained by zero-shot transfer for further training. However, resource creation is expensive and time consuming process, which is further complicated by large number of language pairs between which code-mixing is common. Given this context, faithful translation of monolingual text to code-mixed text can assist construction of task-specific and language-pair-specific datasets - either for training or evaluation.

Guzmán et al. (2017) proposed various code-mixed metrics to quantify degree of code-mixing in a code-mixed sentence/corpus. Code-mix metrics quantify the ratio of tokens contributed by different languages (CMI, M-Index), and probability of switching between two languages (I-Index, Switch Point Average) and the time ordering of switch points in code-mixed text (Burstiness). All the metrics are computed based on the token wise language tag.

Rizvi et al. (2021) proposed, GCM, a toolkit to generate synthetic code-mixed text which are grounded in grammatical theories (Equivalence Constraint Theory and Matrix Language Framework) of code-mixing. Sentences generated using GCM when used to train a RNN-based language model have been shown to significantly reduce the

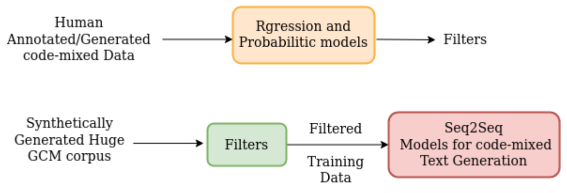
Human Annotated/Generated code-mixed Data
Rgression and Probabilitic models
Filters
Synthetically Generated Huge GCM corpus
Filters
Filtered
Training Data
Seq2Seq Models for code-mixed Text Generation

Figure 1: Methodology for Code-mixed Generation

[2]ALL-CS data is used to compare two approaches: https://github.com/ishan00/translation-for-code-switching-acl

perplexity of the language model. Jawahar et al. (2021) use curriculum training to generate code-mixed Hindi-English data. In the curriculum training training the pre-models are fine-tuned by first training them on synthetic data and then on gold code-mixed data. This architecture has achieved a BLEU score of 12.67 and was place first the overall ranking of CALCS shared task[3]. Gautam et al. (2021) have explored mBART, a pre-trained multilingual encoder-decoder model, to generate Hindi-English text. This methods illustrates the improvement in performance by converting the Hindi roman script to Devanagari script and concatenating Hindi and English sentences for training. Recently, Srivastava and Singh (2021) proposed a dataset capturing quality ratings for synthetically generated code-mixed English-Hindi text. A shared task was also conducted using the dataset. However, the availability of such resources for other code-mix language pairs is limited.

While synthetic data has been used to train machine translation models to generate code-mixed, the quality of those synthetically generated sentences has not been analyzed. We hypothesize that controlling the quality of synthetic code-mix sentences before using them to train translation models can lead to more natural code-mix sentences, and can even be compute efficient. To the best of our knowledge, this is the first work to use human judgements for quality of code-mixed text to create silver parallel data, and use the data to train neural machine translation models for code-mixed text generation.

3 Methodology

In our methodology, as illustrated in Figure 1, we propose models for generating code-mixed text which trained using a silver parallel corpus created by filtering a large synthetic code-mixed corpus. For training the quality filters, we leverage human annotations capturing the quality of code-mixed sentences (Sec.3.1.1) and distributions in human-generated code-mixed sentences (Sec.3.1.2). Using the trained filters we create silver parallel corpus (Sec.3.2). We use the filtered sentences to train machine translation models that will enable generation of code-mixed text (Sec.3.3). In this study we experiment with two language pairs - Hindi-English and Telugu-English.

[3]https://code-switching.github.io/2021#shared-task

3.1 Training Code-Mixed Sentence Quality Predictors : Filtering Mechanism

In this step, we create filters to select the high-quality data from a larger set of synthetic code-mix corpus created by GCM. A sample in GCM consists of English sentence, Hindi/Telugu sentence and Hindi-English/Telugu-English sentence.

We use the following approaches to train our filters.

3.1.1 Regression Filter

In this method, the regression models are trained to predict the rating of the code-mixed sentences. Code-mixing is not an arbitrary mixing of linguistic units from two or more languages. Multilingual speakers possess a strong instinct of when and how to mix. Certain code-mixed structures are preferred by native speakers. The datasets used for training should contain all types of code-mixing, for enabling regression model to filter out good quality code-mixed sentences. To build a regression model, we leverage the following datasets containing human annotations to test the quality of code-mixed sentence.

Hindi-English: Hindi-English regression models are trained (Srivastava and Singh, 2021) HINGE dataset comprising of 4000 Hinglish code mixed sentences. These code-mixed sentences are generated by using two rule-based methods *viz*, Word-aligned code-mixing (WAC) and Phrase-aligned code-mixing (PAC) corresponding to the parallel monolingual Hindi and English sentences. Each of these code-mixed sentences are rated on a scale of 10 by two different annotators.

Telugu-English: Due to the lack of Telugu-English code-mixed datasets that have been evaluated by humans for their quality, we create a new dataset.

We use GCM (Rizvi et al., 2021) to generate synthetic code-mixed sentences. GCM needs monolingual parallel sentences. We feed English-Telugu parallel sentences from Samantar corpus (Ramesh et al., 2022). We randomly select 3,500 such sentences from GCM output for annotation.

An annotator then rates each sentence on a scale of 1-5 based on readability, grammatical correctness, and semantic correctness. A rating of 5 is given to a sentence if the code-mixed sentence sounds fluent and makes semantic sense. Each sample was rated by two annotators to ensure the validity and reliability of the dataset.

In both of the aforementioned datasets, each code-mixed sentence was annotated by two annotators. The ratings given by the annotators were then averaged to obtain the average rating for the sentence. We use the average rating to train our regression predictor models. The features chosen for training are:

- BLEURT scores : BLEURT (Sellam et al., 2020) score, which is reference-based text generation metric, aids us to capture the semantic similarity between a source and a reference sentence. We translate a code-mixed sentence to monolingual English using Google Translate. We compute BLEURT score between the translated English sentence and the actual English sentence that was fed to GCM.

- Code-mixed (CM) metrics : Code-mixed metrics capture the degree of code-mixing in a sentence. Code-mixed metrics include CMI, M-index, I-index, Burstiness and Language Entropy. Code-mixed metrics are computed using token-wise language tags for their calculation. We compute language tags using the model released by Bhat et al. (2017) for Hindi-English and script based identification is used for Telugu-English, where Telugu tokens are in Telugu script.

Using these input features that capture the semantic and linguistic aspects of code-mixed language, we train multiple regression models to predict the rating of each code-mixed sentence.

The regression models used for training included a) Linear, b) Polynomial, and c) mBERT (Multi-lingual Bidirectional Encoder Representations from Transformers) regressions. For BERT based regressor, we add a regression head on top of BERT model. Input to the mBERT based regressor is the code-mix sentence appended with the other input features described above. We evaluate the performance of regression models using metrics such as Mean squared error (MSE), Root mean squared error (RMSE), Mean absolute error (MAE) and Coefficient of determination (R2) and report the results Table 1 and Table 2 for Hindi-English and Telugu-English respectively.

3.1.2 Probabilistic Filter

The regression filter relied on the ratings assigned to synthetically generated code-mixed sentences by humans, which is a cost and time-intensive resource.

Regression	**MSE**	**RMSE**	**MAE**	**R2**
Linear	2.145	1.464	1.186	0.100
Polynomial(degree-2)	2.141	1.463	1.186	**0.101**
BERT	2.074	1.440	1.158	**0.130**

Table 1: Regression Models for Hindi-English

Regression	**MSE**	**RMSE**	**MAE**	**R2**
Linear	1.308	1.143	0.947	0.274
Polynomial(degree-2)	1.303	1.141	0.943	**0.271**
BERT	1.107	1.052	0.826	**0.383**

Table 2: Regression Models for Telugu-English

We train quality predictors based on the properties of these human generated code-mixed sentences. HINGE dataset in addition to the synthetically generated ones also contains human-generated sentences. We compare features (e.g. code-mixed metrics) of a candidate code-mixed sentence against the distribution of same features for human generated sentences. Computationally, it is done by scoring the code-mixed sentences based on the probabilistic distribution of features observed in human-generated code-mixed sentences.

The score of a code-mixed sentence is calculated as the sum of probabilities of its feature values occurring in the human-generated sentences. The formula used for calculating is as follows:

$$score(CM) = \sum_{f=1}^{n} Prob(f(Value)) \qquad (1)$$

where: $Prob(f(Value)) =$ Probability of feature value

For instance, if a sentence has a CMI of 50, we calculate the probability of code-mixed sentences with a CMI index of 50 being present in our corpus of human-generated code-mixed sentences.

The probability of a feature value is calculated using Kernel Density Estimation of the feature. In statistics, **Kernel density estimation (KDE)** is the application of kernel smoothing for probability density estimation. It is a non-parametric method to estimate the probability density function of a random variable based on kernels as weights.

Given a Kernel Density Estimation curve for a feature, probability for a interval of values can only be obtained. We estimate the probability for a particular value by calculating probability for the range of values (featureValue-0.01, featureValue+0.01).

As we are utilizing code-mixed content that has been created by humans, we have opted to utilize the same dataset for filtering in both Hindi-English and Telugu-English code-mixed sentences.

3.2 Data preparation for Encoder-Decoder Models

The data[4] for training Encoder-Decoder models is created using the above filters and applying the trained filters on synthetically generated code mixed generated texts.

From 72,490 Hindi and English parallel sentences GCM toolkit generated 20,00,000 Hindi-English code-mixed sentences, henceforth called GCM-HiEn corpus.

We passed 73,298 Telugu and English parallel sentences to generate 23,37,000 Telugu-English code-mixed sentences, henceforth called GCM-TeEn corpus.

The code-mixed sentences for training Encoder-Decoder models using the above corpora are generated as follows:

- **Random Sampler:** 40,000 sentences are randomly selected from each of GCM-HiEn corpus and GCM-TeEn corpus.

- **Polynomial Filter:** : GCM-HiEn corpus and GCM-TeEn corpus are passed through their respective polynomial regression models and highest rated 40,000 sentences are selected from each corpora

- **BERT Filter :** GCM-HiEn corpus and GCM-TeEn corpus are passed through their respective BERT regression models and highest rated 40,000 sentences are selected from each corpora

- **Probabilistic Filter:** Scores are calculated for all the code-mixed sentences present in GCM-HiEn corpus and GCM-TeEn corpus. 40,000 code-mixed sentences having highest scores are selected from both the corpora.

[4]https://github.com/damasravani19/Enhancing-Code-mixed-Text-Generation-Using-Synthetic-Data-Filtering-in-Neural-Machine-Translation

3.3 Training Encoder-Decoder Models

The filtered data from the above filtering processes is passed through the following Encoder-Decoder models to generate Hindi-English and Telugu-English code-mixed sentences.

- **mT5 :** mT5 (Xue et al., 2021) is a multi-lingual variant of ``Text-to-Text Transfer Transformer" (T5) which is pre-trained on new Common Crawl-based dataset comprising of 101 languages. This model is specifically designed for multi-lingual language processing tasks, including machine translation. The capability of this model with multiple languages and the ability to generate text output from text input makes it suitable for generating code-mixed text.

- **mBART:** mBART (Liu et al., 2020b) is Encoder-Decoder de-noising auto-encoder pre-trained on monolingual corpora in many languages using the BART architecture. It comprises of a shared encoder and language specific decoders allowing it to transfer the knowledge between languages preserving language specific features. It has achieved state-of-art performance on many cross-lingual tasks including machine translation.

4 Experimental Setup

In this section, we present our experiments to examine the effectiveness of each filter and its contribution towards generating high-quality code-mixed sentences. The experimental setup is described in detail, followed by a comprehensive analysis of the results obtained.

In our experimental setup, we performed fine-tuning of pre-trained language models, namely mT5 and mBART, for code-mixed text generation in Hindi-English and Telugu-English.

The input to these models consists of the concatenation of two corresponding monolingual sentences, and the output is a code-mixed sentence. For each language pair, we fine-tuned each model on four different training datasets created using random Sampler, polynomial, BERT, and probabilistic filters, respectively. Using a random sampler as a baseline, our objective was to evaluate the model's performance by using the same hyperparameters and an equal number of samples for both random sampler and other filters.

	CALCS				MrinalDhar				ALLCS			
	mBART		mT5		mBART		mT5		mBART		mT5	
	BLEU	ROUGE-L	BLEU	ROUGE-L	BLEU	ROUGE-L	BLEU	ROUGE-L	BLEU	ROUGE-L	BLEU	ROUGE-L
Raw Sampler	1.89	18.02	2.91	15.3	4.58	18.32	8.45	11.73	3.14	10.62	6.75	12.40
Polynomial Filter	2.84	24.30	4.25	21.49	5.90	24.78	11.74	24.02	7.14	23.50	15.04	21.49
BERT Filter	4.92	32.46	5.41	22.44	9.23	33.48	12.63	25.82	13.99	33.02	15.30	22.44
Probabilistic Filter	4.84	28.82	**6.52**	20.67	9.61	28.61	**15.95**	20.69	17.97	28.8	**30.02**	24.84

Table 3: Performance of Hindi-English code-mixed generation models. Best performing models with highest BLEU scores are marked in bold

	SentiDataset				DialogueDataset			
	mBART		mT5		mBART		mT5	
	BLEU	ROUGE-L	BLEU	ROUGE-L	BLEU	ROUGE-L	BLEU	ROUGE-L
Raw Sampler	4.56	18.54	7.86	20.52	4	16.26	3.27	15.43
Polynomial Filter	11.46	34.23	12.54	38.44	7.54	24.63	9.98	25.83
BERT Filter	10.04	47.3	14.05	39.56	9.34	27.75	11.71	28.32
Probabilistic Filter	12.42	9.15	**21.96**	53.56	11.018	31.28	**17.39**	28.77

Table 4: Performance of Telugu-English code-mixed generation models. Best performing models with highest BLEU scores are marked in bold

We fine-tuned mT5 and mBART for Hindi-English and Telugu-English code-mixed text generation using appropriate hyperparameters. For mT5, we trained with a batch size of 64 and a learning rate of 2e-3, while for mBART, we used a batch size of 32 and a learning rate of 3e-6. We used the default Ada-W optimizer (Kingma and Ba, 2014) for training both models, and selected the hyperparameters to minimize the validation dataset loss.

4.1 Test Datasets

For Hindi-English code-mix text generation, we used three different datasets for testing: (a) ALL-CS dataset (b) CALCS-2021 (Chen et al., 2022) shared task validation dataset and (c) A parallel English and English-Hindi code-mixed sentences dataset created by (Dhar et al., 2018), henceforth called MrinalDhar dataset. The Hindi translations for MrinalDhar dataset are obtained from Google Translate of the corresponding English sentences. The ALL-CS test dataset contains code-mixed sentences and their corresponding Hindi translations, while the English translations for this dataset were generated using Google Translate.

For Telugu-English code-mix text generation, we used two datasets a) Sentiment analysis dataset proposed by Kusampudi et al. (2021), which contains code-mixed sentences collected from Twitter. We selected 500 code-mixed sentences from this dataset, henceforth called SentiDataset. b) (Dowlagar and Mamidi, 2023) provided 3005 code-mixed dialogs between doctors and patients. We hand-picked 750 code-mixed sentences, henceforth called DialogueDataset for our evaluation. The monolingual sentences for the corresponding code-mixed sentences in the dataset are generated manually.

We evaluate the performance of our models using standard metrics such as BLEU scores(SacreBLEU) and ROUGE-L scores, and report the results in Table 3 and Table 4 for Hindi-English and Telugu-English, respectively.

5 Results and Analysis

The datasets used for evaluation include code-mixed sentences that are sourced from various social media platforms (CALCS, MrinalDhar, SentiDataset) as well as sentences that are generated by humans(ALL-CS), and those that are transcribed from speech (DialogueDataset). Our models were able to achieve quality results on a variety of code-mixed datasets, despite the differences in sampling and characteristics between the training (synthetically generated) and testing sets. This suggests that our models are **robust and can be applied to a wide range of datasets** with varying characteristics and highlights the effectiveness of our models.

In a similar experiment proposed by (Tarunesh et al., 2021), models when trained on synthetic data and tested on the ALL-CS test dataset achieved a BLEU score of 17.73. However, our mT5 model trained with data after applying probabilistic filtering outperformed it, **achieving a much higher score of 30.02**. This significant improvement highlights the importance of using probabilistic models for code-mixed language translation, as it allows for better modeling of the underlying language patterns and improves the overall performance of the

Hindi : अंजेलिना एक नए उत्साह के साथ इस फाइनल राउंड में उत्तरी हैं
English : Angelina enters this final round with a new spirit
Code-mixed sentence : angelina एक नए उत्साह के साथ इस final round में उत्तरी हैं

Generated code-mixed sentences :

Raw Sampler	: Angelina उत्तरी this final round with a new spirit
Polynomial Filter	: उन final round में उत्तरी हैं
BERT Filter	: अंजेलिना एक new spirit के साथ इस final round में उत्तरी हैं
Probabilistic Filter	: Angelina एक new spirit के साथ इस final round में उत्तरी हैं

Figure 2: Example illustrating Hindi-English code-mixed text generation using multiple filters

Telugu : కానీ అన్ని పాజిటివ్ గానే రిపోర్ట్స్ వచ్చాయి.
English : But all the reports were positive.
Code-mixed sentence: కానీ reports అన్ని positive గానే వచ్చాయి.

Generated code-mixed sentences :

Raw Sampler	: But అన్ని పాజిటివ్ గానే reports
Polynomial Filter	: కానీ అన్ని reports positive గానే రిపోర్ట్స్ వచ్చాయి.
BERT Filter	:.But అన్ని the reports were positive.
Probabilistic Filter	: But అన్ని positive గానే reports వచ్చాయి.

Figure 3: Example illustrating Telugu-English code-mixed text generation using multiple filters

system. Our mT5 model, fine-tuned on probabilistic filtered data, achieves a lower BLEU score of 6.52 compared to the previous work by (Jawahar et al., 2021). Their research reported the highest BLEU score of 14.6 for the CALCS validation dataset in the code-mixed generation task. Notably, we are unaware of any reported results for the code-mixed generation task on the MrinalDhar dataset.

The mBART model with BERT filtering achieved the highest ROUGE-L scores, while the mT5 model with probabilistic filtering achieved the highest BLEU scores, for Hindi test datasets. For Telugu test datasets, models with probabilistic filtering achived higher ROUGE-L and BLEU scores.

The BLEU and ROUGE-L scores demonstrate how **filtering plays a significant role in the model's performance**. All testing datasets are human-generated code-mixed sentences, highlighting how the filters aid in generating code-mixed sentences that closely resemble human-generated ones.

The **Probabilistic filter** uses the feature distribution of human-generated code-mixed sentences to improve the model's performance. This filter was initially created using the Hindi-English dataset but was also applied to the Telugu-English dataset, demonstrating its **language-independent nature**. The good results obtained from Telugu-English code-mixed test generation highlight the power and effectiveness of this filtering mechanism.

5.1 Error Analysis

We have conducted a manual analysis of the outputs generated by the models with the best BLEU scores for the all test datasets. Based on this analysis, we have identified several areas where the models exhibit errors. To better understand these errors, we have categorized them into different groups.

1. **Sentence Truncation :** It is observed that the system generated incomplete sentences when presented with long input lengths.
   - प्रधानमंत्री Manmohan Singh के साथ वाम दलों की आज breakfast meeting
     - Translation : Along with Left parties prime minister Manmohan Singh today breakfast meeting.
     - Explanation : The ending of the sentence is missing.
   - సినిమా లగీ మీ review నెమ్మది గా ఉంది అన్న, I think it n
     - Translation : Movie is also like like our review, I think is n.
     - Explanation : The model stopped generating after generating a character in the last word. It also needs some more information for complete understanding.

2. **Bilingual word overlap :** Some of the generated code-mixed sentences contain both lexical/phrasal equivalents from both languages, which can make the sentences understandable but not natural-sounding as they do not reflect how humans typically code-switch or code-mix in conversation.
   - Who told you this professor thing तुमको किस्सने बोला
     - Translation : 'Who told you this professor thing, Who told you'.
     - Explanation : The English phrase 'Who told you' has same meaning as the Hindi phrase 'तुमको किस्सने बोला'

3. **Pseudo code-mixing :** It appears that in some cases, although the script is written in one language it is actually a word from another language in code-mixed sentences, actual code-mixing may not be occurring to the extent that it appears.

- The code-mixed sentence ``ఓ మై గాడ్, I got reply" generated by model, translates to ``oh my god, I got reply".
  - Explanation : The phrase 'ఓ మై గాడ్' is English phrase 'oh my god' written in Telugu script. So, the code-mixed sentence is actually a monolingual sentence in English.

4. **Lack of intra word code-switching** Our analysis has revealed that our systems were not able to handle intra-word level code-mixing, where a single word contains characters from both the languages. This issue was especially prominent in Telugu-English code-mixed sentences, where there is a high degree of intra-word code-mixing due to the structure and morphology of the Telugu language.

   - The voice clear లేదు.
     - Translation: The voice is not clear. The reference code-mixed sentence in Dialogue dataset is : voice clearగా లేదు. The model could not generate clearగా, which has intra-word level code-mixing.

6 Conclusion

In this work, we present a novel approach to create high-quality silver parallel data for code-mixed data. The primary focus of our approach is to select natural code-mix sentences from a larger synthetically generated code-mixed corpus. Leveraging human knowledge, we train filters to select high-quality code-mixed sentences. Using the filtered sentences, we fine-tune MLLMs for machine translation task. Our filtering-based neural machine translation approach for code-mixed sentence generation shows promising results across various datasets, and different language pairs - Hindi-English and Telugu-English. The fine-tuning of pre-trained models such as mT5 and mBART has enabled us to generate high-quality code-mixed sentences with minimal gold-standard corpus. We also experimented with the probabilistic filter method, which does not need human annotations for quality but relies on human generated code-mixed sentences. The probabilistic filter is effective and language-independent, as probabilistic filter either matches or outperforms other filters proposed in the study. It can easily be extended to other languages, unlike other mechanisms that require human effort. Our study has implications for the generation of natural code-mixed sentences at scale - which can improve downstream task performances.

6.1 Limitations and Future Work

Training supervised filters for the quality of code-mixed text is dependent on the availability of human-annotated corpus. The availability of such resource limits the extension of our methods to other language pairs. It would also be worthwhile to investigate the effectiveness of filtering techniques creating high-quality code-mixed data, particularly low-resource languages, for advancing the research for resource-constrained code-mixing language pairs. Additionally, One-shot and Zero-shot learning techniques could also be explored to determine whether the models are trained to generate code-mixed sentences in general or if it is specific to the languages they are trained upon.

One potential future research direction is to explore the performance of models when trained on a combination of various filtering mechanisms for generating code-mixed text. BERT and polynomial filters are created based on GCM, which generates code-mixing using some techniques only. A further analysis by humans on the code-mixed sentences generated using these as training data could give us valuable insights into these approaches.

In this study, we have relied on n-gram overlap measures (BLEU, ROUGE) for evaluating the models. In the context of code-mixing, such measures are limited because there could be multiple ways of writing the same code-mixed sentence. Even if the model output is valid and semantically coherent code-mixed translation, measures like BLEU/ROUGE could mischaracterize the quality of translations. Exploring semantic evaluation methods (like BERTScore) for code-mixed text could be another avenue for future work.

References

Irshad Bhat, Riyaz A Bhat, Manish Shrivastava, and Dipti Sharma. 2017. Joining hands: Exploiting monolingual treebanks for parsing of code-mixing data. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 324--330.

Eyamba G Bokamba. 1989. Are there syntactic constraints on code□mixing? *World Englishes*, 8(3):277--292.

Shuguang Chen, Gustavo Aguilar, Anirudh Srinivasan, Mona Diab, and Thamar Solorio. 2022. Calcs 2021 shared task: Machine translation for code-switched data.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Mrinal Dhar, Vaibhav Kumar, and Manish Shrivastava. 2018. Enabling code-mixed translation: Parallel corpus creation and MT augmentation approach. In *Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing*, pages 131--140, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Suman Dowlagar and Radhika Mamidi. 2023. A code-mixed task-oriented dialog dataset for medical domain. *Computer Speech Language*, 78:101449.

Devansh Gautam, Prashant Kodali, Kshitij Gupta, Anmol Goel, Manish Shrivastava, and Ponnurangam Kumaraguru. 2021. CoMeT: Towards code-mixed translation using parallel monolingual sentences. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 47--55, Online. Association for Computational Linguistics.

Sunil Gundapu and Radhika Mamidi. 2018. Word level language identification in English Telugu code mixed data. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong. Association for Computational Linguistics.

Akshat Gupta, Sargam Menghani, Sai Krishna Rallabandi, and Alan W Black. 2021. Unsupervised self-training for sentiment analysis of code-switched data. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 103--112, Online. Association for Computational Linguistics.

Gualberto A Guzmán, Joseph Ricard, Jacqueline Serigos, Barbara E Bullock, and Almeida Jacqueline Toribio. 2017. Metrics for modeling code-switching across corpora. In *INTERSPEECH*, pages 67--71.

Ganesh Jawahar, El Moatez Billah Nagoudi, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2021. Exploring text-to-text transformers for English to Hinglish machine translation with synthetic code-mixing. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 36--46, Online. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Siva Subrahamanyam Varma Kusampudi, Preetham Sathineni, and Radhika Mamidi. 2021. Sentiment analysis in code-mixed Telugu-English text with unsupervised data normalization. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 753--760, Held Online. INCOMA Ltd.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020a. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726--742.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020b. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726--742.

C. Myers-Scotton. 1997. *Duelling Languages: Grammatical Structure in Codeswitching*. Clarendon Press.

Shana Poplack. 1980. Sometimes i'll start a sentence in spanish y termino en espaÑol: toward a typology of code-switching1. 18(7-8):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. 2022. Samanantar: The largest publicly available parallel corpora collection for 11 indic languages. *Transactions of the Association for Computational Linguistics*, 10:145--162.

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.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of ACL*.

Shashi Shekhar, Dilip Kumar Sharma, and MM Sufyan Beg. 2020. Language identification framework in code-mixed social media text based on quantum lstm—the word belongs to which language? *Modern Physics Letters B*, 34(06):2050086.

Vivek Srivastava and Mayank Kumar Singh. 2021. Hinge: A dataset for generation and evaluation of code-mixed hinglish text. *ArXiv*, abs/2107.03760.

Ishan Tarunesh, Syamantak Kumar, and Preethi Jyothi. 2021. From machine translation to code-switching: Generating high-quality code-switched text. *ArXiv*, abs/2107.06483.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483--498, Online. Association for Computational Linguistics.