MSR India at SemEval-2020 Task 9: Multilingual Models can do Code-Mixing too

Anirudh Srinivasan
Microsoft Research, India
t-ansrin@microsoft.com*

Abstract

In this paper, we present our system for the SemEval 2020 task on code-mixed sentiment analysis. Our system makes use of large transformer based multilingual embeddings like mBERT. Recent work has shown that these models posses the ability to solve code-mixed tasks in addition to their originally demonstrated cross-lingual abilities. We evaluate the stock versions of these models for the sentiment analysis task and also show that their performance can be improved by using unlabelled code-mixed data. Our submission (username Genius1237) achieved the second rank on the English-Hindi subtask with an F1 score of 0.726.

1 Introduction

The task of identifying sentiment from text is extremely important in this age where large volumes of text content are being consumed via social media. The task becomes even more interesting when it comes to bilingual communities as these communities exhibit the phenomenon of code-mixing online (Rijhwani et al., 2017).

Existing approaches to tackling this problem have mainly been based on statistical methods (Vilares et al., 2016; Patra et al., 2018). These methods have used features like n-gram counts and TF-IDF vectors along with a linear classifier. There have been very few approaches to this problem using deep learning as the amount of labelled code-mixed data available has always been very less. Methods like the one in Pratapa et al. (2018b) train word embeddings using unlabelled code-mixed data, the availability of which is not as problematic as labelled data, and use these embeddings along with a recurrent neural network based model.

Recent advancements in natural language processing have shown that large transformer based models like BERT (Devlin et al., 2019), when pre-trained on large corpora, are easily adaptable for downstream tasks with small datasets. These models even perform well in a cross-lingual manner (Conneau et al., 2018) when pre-trained on corpora spanning multiple languages. Our experiments show that these multilingual models perform well even on code-mixing tasks, having had no exposure to any code-mixing during pre-training. We use such a system to solve the code-mixed sentiment analysis problem. We also show that it's performance can be improved by using a combination of generated and real code-mixed text.

The rest of the paper is organized as follows. Section 2 tasks about the dataset for the task and the pre-processing done to it. Section 3 talks about the different systems we evaluated, with Section 3.3 in particular going into how we improved the multilingual models using code-mixed data. Section 4 describes the performance of the different models and Section 5 concludes our discussion.

2 Dataset and Preprocessing

The details about the datasets (Patwa et al., 2020) for both the English-Hindi (En-Hi) and English-Spanish (En-Es) tasks are described in Table 1. The datasets comprise entirely of tweets. The English-Hindi

*Author can be contacted at anirudhsriniv@gmail.com

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 951–956
Barcelona, Spain (Online), December 12, 2020.

Language	Train	Dev	Test
En-Es	12002	2998	3789
En-Hi	14000	3000	3000

Table 1: Dataset details

Feature	mBERT	XLM-R
Tokenization	WordPiece	SPM
Languages	104	100
Vocab	30k	250k
Num. Layers	12	12
Params	110M	270M

Table 2: Model Differences

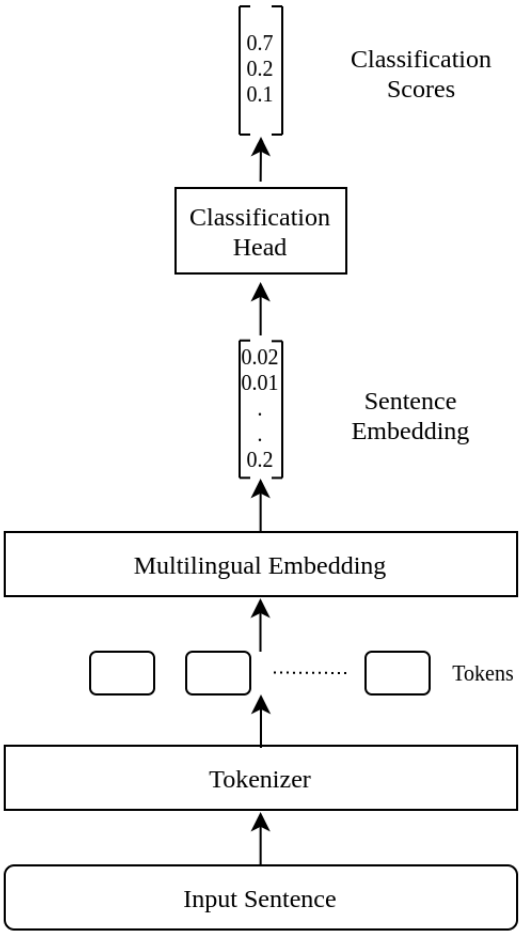

Figure 1: Model

dataset consists of tweets where the Hindi is written in the Roman script. We make use of the language identification tool by Gella et al. (2014) to identify the Romanized Hindi sections and transliterate them to Devanagari using the Bing Translator API [1]. The language tags provided along with the data is not used. No other pre-processing is done to the data.

3 System Description

Figure 1 describes the model used for sentiment analysis. The model is a classification model that comprises of a pretrained transformer-based multilingual embedding (like BERT) and a linear layer acting as a classification head. The embedding takes in a tokenized sentence and outputs a single embedding for that sentence. This embedding is then run through the linear layer that outputs scores for each of the 3 classes. The entire system was implemented using the Huggingface Transformers library (Wolf et al., 2019). We experimented with different models for the embedding. We also experimented with different pooling techniques that are used to obtain the sentence embedding and these are detailed below. Finally, as a baseline, we report the results from the method in Pratapa et al. (2018b), using Word2vec embeddings trained on code-mixed data along with a BiLSTM.

[1] https://aka.ms/translatordevdoc

3.1 Multilingual Embeddings

Multilingual BERT (mBERT) (Devlin et al., 2019) is a transformer based model that is pre-trained on a corpora comprising 104 languages. This performs well on cross-lingual tasks like XNLI and this was taken as our baseline model. A more recent model is XLM-Roberta (XLM-R) (Conneau et al., 2019) and this has been shown to outperform BERT on many cross-lingual tasks. This differs from BERT in the type of tokenization it uses and the amount of data it is pre-trained on. Table 2 contains a list of differences between the two models. We use the `bert-base-multilingual-cased` model for BERT and the `xlm-roberta-base` model for XLM-R.

3.2 Sentence Embedding Technique

The aforementioned multilingual models output one embedding per input token. These need to be pooled together to obtain a sentence embedding to use for the sequence classification task. There have been multiple works proposing different methods to obtain a sentence embedding from BERT (Reimers and Gurevych, 2019; Wang and Kuo, 2020). The two most popular (and simplest) methods are performing average pooling over the embeddings of every token or using the embedding of the first token (`[CLS]` token in case of BERT, `<s>` in case of XLM-R). We evaluate both these methods and report the performance of both.

3.3 Finetuning Multilingual Embeddings on Code-Mixed Data

There have been multiple works proposing techniques to create domain specific versions of models like BERT (Sun et al., 2019; Lee et al., 2019; Alsentzer et al., 2019). Khanuja et al. (2020) showed that when models like mBERT are finetuned on synthetic and non-synthetic code-mixed data, they perform much better on downstream code-mixed tasks. Along these lines, we finetune both mBERT and XLM-R with code-mixed data on the masked language modeling task. We follow a 2 stage curriculum, first finetuning on a large corpus of 2 million generated (synthetic) code-mixed sentences and then with a smaller corpus of 90,000 real (non-synthetic) code-mixed sentences. The curriculum followed and synthetic sentences generated are based on the technique in Pratapa et al. (2018a). We create one model each for En-Es and En-Hi, finetuned on code-mixed data from that pair. We call these Modified mBERT and Modified XLM-R.

4 Results and Analysis

The results are presented in Tables 3 and 4. Each table contains F1 scores averaged over 5 different seeds. For all the runs, a batch size of 64 was used along with the Adam optimizer with a learning rate of 5e-5. Each batch was made to have equal number of samples from all 3 classes. Training was performed for 10 epochs. Right away, we are able to observe that the stock versions of mBERT and XLM-R, which are not exposed to any form of code-mixing during their pre-training show impressive F1 scores. This is talked about more in Section 4.2. We present an analysis of the sentence embedding techniques first.

4.1 Sentence Embedding Methods

Both the sentence embedding methods experimented with are shown as separate columns in Tables 3 and 4. Using average pooling does bring in improvements in some cases, mainly on the Dev sets, but the corresponding Test set numbers are not better.

The embedding of the first token (`[CLS]`/`<s>`) in the final layer is computed as a weighted sum over the embeddings of the all the tokens of the $n - 1^{st}$ layer. Given such a mechanism, the embedding of the first token may be able to capture enough information over all the tokens of the sentence and is able to perform as well as the average pooling method for a simple sequence classification task. Our results are in line with the results in Wang and Kuo (2020), where most simple downstream tasks do not see big differences in performances of the 2 embedding methods, with only more complex sentence similarity or probing tasks showing the average pooling method to perform better.

Model/Sent. Embedding	First Token		Avg. Pooling	
	Dev	Test	Dev	Test
Word2vec + BiLSTM	67.83	59.73	-	-
Stock mBERT	72.29	65.80	**81.38**	66.21
Modified mBERT	**78.49**	67.32	79.31	66.60
Stock XLM-R	69.62	**70.40**	73.79	**69.69**
Modified XLM-R	72.74	70.03	72.58	69.50

Table 3: F1 scores on En-Hi Dataset

Model/Sent. Embedding	First Token		Avg. Pooling	
	Dev	Test	Dev	Test
Word2vec + BiLSTM	54.50	-	-	-
Stock mBERT	60.06	-	60.31	-
Modified mBERT	60.66	63.73	**61.94**	-
Stock XLM-R	57.45	**68.44**	61.23	-
Modified XLM-R	**62.00**	-	56.84	-

Table 4: F1 scores on En-Es Dataset[a]

[a]Since test labels were not available, we only have numbers from the models that were submitted to the online contest

4.2 Finetuning on Code-Mixed Data

Both mBERT and XLM-R performing well on these tasks is pretty impressive. Finetuning[2] these models with code-mixed data improves the performance of the stock models. We observe an improvement in almost all the cases, ranging from 1-5%. Our results resonate with the ones in Khanuja et al. (2020), suggesting that most code-mixed tasks can be solved by simply using multilingual embeddings like mBERT, finetuning them on any available code-mixed data if better performance is needed.

4.3 Class-Wise Performance Analysis

We take the best performing model (on the test set this is Stock XLM-R) for both tasks and analyse the class-wise precision, recall and F1-scores. These are depicted in Tables 5 and 6. Given that training was with data balanced across the 3 classes, similar performance across them is expected. This is observed in the En-Hi task, with all 3 classes having precision and recall within a small range. Similar numbers are observed between the dev and test sets too. However when it comes to the En-Es test set, there is a big gap between the classes. The precision values for the neutral class is extremely low and this is affecting the overall F1 scores. Interestingly, this gap in scores isn't present on the dev set, suggesting that there is some aspect of the test set that the model is unable to learn from the train set during the training process.

5 Conclusion

In this paper, we present our system for the SemEval 2020 task on code-mixed sentiment analysis. We make use of multilingual models like mBERT and show that they work well for code-mixing tasks. The best performance is extracted from these models by finetuning them on code-mixed data and using this version instead of their stock versions. We also find that for simple sequence classification tasks, the choice of sentence embedding technique does not have a significant impact on the result.

There are multiple paths for further exploration of this work. While finetuning mBERT on code-mixed data, we've created one model per language and used a relatively small amount of data (compared to the amount of data BERT is pretrained on). Both these could be looked into, creating a single model for

[2]MLM finetuning

Class/Measure	Precision	Recall	F1
Dev			
Positive	0.72	0.78	0.75
Neutral	0.62	0.61	0.62
Negative	0.71	0.65	0.68
Test			
Positive	0.76	0.79	0.78
Neutral	0.62	0.66	0.64
Negative	0.76	0.66	0.70

Table 5: En-Hi Task: Class-wise performance with Stock XLM-R

Class/Measure	Precision	Recall	F1
Dev			
Positive	0.67	0.61	0.64
Neutral	0.46	0.40	0.43
Negative	0.43	0.66	0.52
Test			
Positive	0.92	0.64	0.76
Neutral	0.09	0.61	0.16
Negative	0.53	0.35	0.42

Table 6: En-Es Task: Class-wise performance with Stock XLM-R

multiple language pairs, and using much more data for this purpose. In this process, one may be able to obtain a universal model that works for a large number of code-mixed pairs in addition to the large number of languages that mBERT already supports.

Acknowledgements

We would like to thank Monojit Choudhury and Sebastin Santy for their feedback during the model evaluation process and Simran Khanuja for setting up the model training process.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium, October-November. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Spandana Gella, Kalika Bali, and Monojit Choudhury. 2014. "ye word kis lang ka hai bhai?" testing the limits of word level language identification. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 368–377, Goa, India, December. NLP Association of India.

Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. 2020. GLUECoS: An evaluation benchmark for code-switched NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3575–3585, Online, July. Association for Computational Linguistics.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, Sep.

Braja Gopal Patra, Dipankar Das, and Amitava Das. 2018. Sentiment analysis of code-mixed indian languages: An overview of sail_code-mixed shared task @icon-2017.

Parth Patwa, Gustavo Aguilar, Sudipta Kar, Suraj Pandey, Srinivas PYKL, Björn Gambäck, Tanmoy Chakraborty, Thamar Solorio, and Amitava Das. 2020. Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. In *Proceedings of the 14th International Workshop on Semantic Evaluation (SemEval-2020)*, Barcelona, Spain, December. Association for Computational Linguistics.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018a. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553, Melbourne, Australia, July. Association for Computational Linguistics.

Adithya Pratapa, Monojit Choudhury, and Sunayana Sitaram. 2018b. Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3067–3072, Brussels, Belgium, October-November. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November. Association for Computational Linguistics.

Shruti Rijhwani, Royal Sequiera, Monojit Choudhury, Kalika Bali, and Chandra Shekhar Maddila. 2017. Estimating code-switching on twitter with a novel generalized word-level language detection technique. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1971–1982, Vancouver, Canada, July. Association for Computational Linguistics.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification?

David Vilares, Miguel A. Alonso, and Carlos Gómez-Rodríguez. 2016. EN-ES-CS: An English-Spanish code-switching twitter corpus for multilingual sentiment analysis. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4149–4153, Portorož, Slovenia, May. European Language Resources Association (ELRA).

Bin Wang and C. C. Jay Kuo. 2020. Sbert-wk: A sentence embedding method by dissecting bert-based word models.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing.