HinglishNLP: Fine-tuned Language Models for Hinglish Sentiment Detection

Meghana Bhange
Verloop.io
Bengaluru
meghana@verloop.io

Nirant Kasliwal
Verloop.io
Bengaluru
hi@nirantk.com

Abstract

Sentiment analysis for code-mixed social media text continues to be an under-explored area. This work adds two common approaches: fine-tuning large transformer models and sample efficient methods like ULMFiT (Howard and Ruder, 2018). Prior work demonstrates the efficacy of classical ML methods for polarity detection. Fine-tuned general-purpose language representation models, such as those of the BERT family are benchmarked along with classical machine learning and ensemble methods. We show that NB-SVM beats RoBERTa by 6.2% (relative) F1. The best performing model is a majority-vote ensemble which achieves an F1 of 0.707. The leaderboard submission was made under the codalab username `nirantk`, with F1 of 0.689.

1 Introduction

Code-mixing or code-switching refers to the use of two or more languages or speech variants together (Contini-Morava, 1995). This is commonly observed in informal conversations, especially those on social media, e.g. Twitter (Rudra et al., 2016) (Rijhwani et al., 2017). While a small body of work does exist on code-mixing detection, this task focuses on polarity detection (Sentiment Analysis). We demonstrate the impressive performance of transfer learning for the task of sentiment detection in code-mixed context. We discuss limitations of existing deep learning pre-trained models which are trained on ”monolingual” text – which has sentences in only one language.

In this work, we demonstrate how it is beneficial to fine-tune the language model (LM) for a code-mixed setting like Hinglish, when Hindi is written in the Roman script. The labelled data for the sentiment classifier is from the task paper — (Patwa et al., 2020). It consists of a total of 17000 tweets. The sentiment labels are positive, negative, or neutral, and the code-mixed languages are English-Hindi. The train data after the split has 14k tweets. The validation and test data contain 3k tweets each.

2 Data

We used two primary datasets. The data used for fine-tuning LM is from Twitter stream. The tweet stream data was used which contains ∼1.9 million Hinglish tweets. We manually sampled 3k tweets to verify what fraction of them are Hinglish. This dataset consists of ∼86.5% Hinglish tweets. The 17k tweet Sentimix data (Patwa et al., 2020) was used to further fine-tune the sentiment classifier. The Sentimix data contains 14,000 tagged tweets marked as positive, negative and neutral for training, and 3,000 for development and testing each.

2.1 Mining and Filtering Twitter Corpus

Pre-training LM require large datasets. In order to enable this, we gathered ∼1.9 Million tweets from the Twitter 1% sample stream for the entire year of 2018. We curated a seed dictionary of Hinglish words and their spelling variants. Next, we calculate the Jaccard Index (Jaccard, 1912) between our seed dictionary

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 934–939
Barcelona, Spain (Online), December 12, 2020.

and every tweet. For values above 0.6, we mark the tweet as "Hinglish tweet". This threshold value 0.6 was selected empirically, by evaluating Jaccard values for 200 tweets.

Next, every token which is present in the tweet, but missing in our dictionary is marked as a Candidate token C_t. We remove duplicates to get our set of unique candidate tokens C. For every unique C_t in C, we manually review and add to our dictionary. A known limitation of this iterative-expansion dictionary based approach is that it starts out with a bias for smaller tweets with fewer total tokens. Hence, we repeated this exercise in batches of 10,000 tweets each – till we saw two batches of full 280 character tweets. We marked these tweets as "highly likely" to be Hinglish. These were roughly 160,000 tweets. A secondary split of roughly 380,000 tweets was marked as "possibly" Hinglish. Both of these were primarily used for training or fine-tuning the LM backbone.

The ~1.9M tweets are composed of 3 different splits, with differing Hinglish percentage. The first split of 162K tweets is enriched to 86.5% Hinglish, with 13% tweets being empty, or non-Hinglish in other ways. The second split of 384K tweets is enriched to 89.6%. The remaining $\tilde{1}$.4M tweets are expected to have 83% Hinglish tweets. We randomly pulled 1K tweets from each of these splits to get these estimates. The estimate is hence, prone to sampling biases/errors. To re-iterate, this added dump is neither tagged with polarity nor pure Hinglish.

2.2 Release

The dataset is released on Github.[1] Since Twitter discourages releasing the text directly, tweet_ids are shared. This leaves the user to pull the specific tweets using Twitter's Developer API.

2.3 Cleaning and Pre-processing

We de-duplicated the 1.9M tweets corpus using a string equality check. We also de-duplicated tweets using the meta-information in the JSON when a "retweeted" text is included twice. The text was pre-processed before data were introduced to the model. The pre-processing included removal of both external links and shortened twitter links. The "@" was replaced with with "mention". Similarly, "#" was replaced with the word "hashtag". Emojis were converted to text equivalent using the emoji package (Taehoon Kim and Kevin Wurster, 2019). During this stage, both the datasets (SemEval and Twitter Large Supervised Dataset) are pre-processed with identical code, both during training/fine-tuning and inference.

3 Experiments

3.1 Training and Fine-tuning

The Language Models (LMs) were fine-tuned on the entire Twitter Stream Sample. We used held out about 10% for measuring LM perplexity. Classifiers were trained using 14000 tweets from the 17000 tweets in the SemEval training corpus. The linear layers were fine-tuned on the SemEval training corpus for 3 epochs for all experiments. The fine-tuning parameters for the BERT-family sentiment classifiers are referenced in Table 1. For Attention dropout and hidden dropout, the parameters were empirically chosen using random grid-search with a range of 0.1 to 0.9. The range considered for Adam Epsilon was 1e-8 to 9e-8 with 1e-8 granularity. Learning rates varied from 1e-7 to 1e-4. These parameters were combined with two learning rate schedulers, a linear learning rate scheduler and a cosine learning rate scheduler. The training for models which took place in two steps: First, the pre-trained language model was fine-tuned using the 1.9M tweets. Second, this fine-tuned deep LM was used as an encoder for training the polarity classifier using the 14K tagged tweets from Sentimix.

3.2 Evaluation

During the competition, we used multiple methods of evaluation and different train-test splits. In this work, the **test dataset** refers to the officially released test set of 3,000 tweets. The performance numbers have been updated to reflect the same. We chose to ignore the validation set from SemEval for evaluation because most of our LMs had consistently very high performance of 0.95 F1 or more on the set. The F1

[1] https://github.com/NirantK/Hinglish

Parameter	BERT Multilingual	Hinglish Fine-tuned BERT	RoBERTa	DistilBERT
Attention Dropout Probability	0.4	0.4	0.1	0.6
Hidden Dropout Probability	0.3	0.3	0.1	0.6
Adam Epsilon	3e-8	1e-8	5e-8	1e-8
Warmup Steps	100	100	0	100
Maximum Learning Rate	5e-7	5e-7	4e-5	3e-5
Learning Rate Scheduler	linear	linear	linear	cosine

Table 1: Fine-tuning Parameters for BERT Family Classifiers

score used for internal evaluation is macro-F1 while the leaderboard submission uses weighted-F1. The F1 scores that are shown in the result table are from internal evaluation and thus are macro-f1.

4 Modeling Approaches

4.1 NB-SVM

NBSVM is the approach proposed by Wang and Manning (2012), which performs well on text classification in tasks like sentiment classification. It takes a linear model such as SVM (or logistic regression) and incorporates the possibilities of Bayesian by replacing terms with Naive Bayes log-count ratios. The NBSVM implementation was borrowed as is from zaxcie (2018). The motivation for using NBSVM is that they are comparatively faster to train as opposed to deep learning models. We chose $C = 4$, the inverse regularization parameter.

4.2 ULMFiT: Universal Language Model Fine-tuning for Text Classification

We used AWD-QRNN (Bradbury et al., 2016) instead of AWD-LSTM (Howard and Ruder, 2018) for pre-training and fine-tuning. We used Sentence Piece(Kudo and Richardson, 2018) for text tokenisation. The intent was to capture sub-word level features. Vocabulary Size of the sentencepiece tokenizer was 8000 and was trained on 540k tweets to save compute time. For the ULMFiT-QRNN model batch size of 1024 was used while training both the LM and classifier (linear) layers. AWD-LSTM gives an F1 0.48 on the test set where as AWD-QRNN performs with an F1 of 0.650. The hypothesis which could explain this is that tweet length, which is typically less than 140 characters, is too short for LSTM is learn a meaningful pattern.

4.3 BERT Multilingual

BERT-base-multilingual-cased (Devlin et al., 2018), without any fine-tuning of LM on Hinglish data, was used to train the sentiment classifier. It is trained on cased text in the top 104 languages with the largest Wikipedia corpora. The linear layers were trained/fine-tuned without updating the frozen backbone for 3 epochs.

4.4 Hinglish Fine-tuned BERT

The base model for fine-tuning BERT LM on Hinglish data was *BERT-base-multilingual-cased* (Devlin et al., 2018). Both the backbone and linear layers of the LM were fine-tuned. This was on a pre-processed Twitter Stream Sample (described in the previous section) over 26,000 iterations. It was trained for a total of 4 epochs. Training batch size was four and vocab size 119,547. The perplexity of the fine-tuned LM was 8.2. The trained BERT tokenizer and model were utilized for fine-tuning classifier.

4.5 RoBERTa

RoBERTa (Liu et al., 2019) is a robustly optimized BERT pre-training approach. It is trained over longer sequences and removes the next sentence prediction task from BERT pre-training. The base model for fine-tuning the LM-backbone for RoBERTa on Hinglish data was *RoBERTa-base*. The LM was fine-tuned

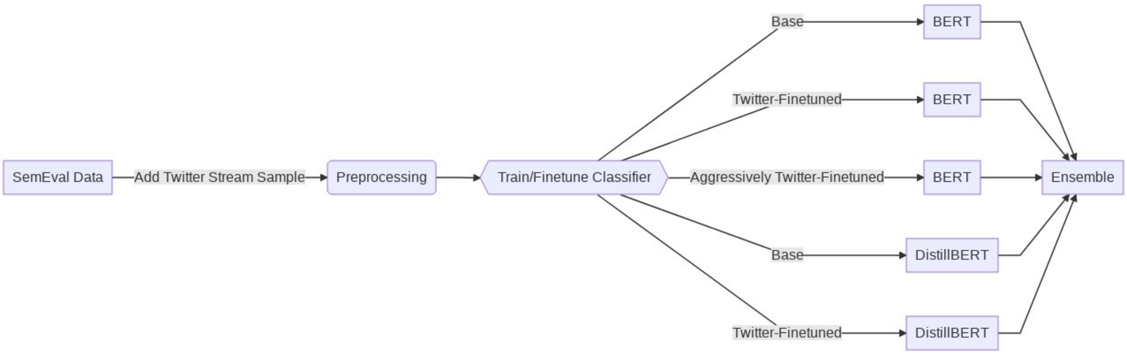

Figure 1: Once pre-processed, the data is used for predicting results which are then passed to the ensemble described in section 4.7.

Model	Accuracy	Precision	Recall	F1	LM-Perplexity
Majority Vote (Ensemble)	**0.704**	**0.709**	**0.707**	**0.707**	–
DistilBERT-Base-cased	0.685	0.690	0.685	0.687	**6.51**
Logistic Regression Funnel (Ensemble)	0.678	0.678	0.696	0.684	–
BERT-Base-Multilingual-Cased	0.680	0.692	0.678	0.677	8.2
NB-SVM (Ensemble)	0.667	0.666	0.685	0.673	–
ULMFit AWD-QRNN	0.645	0.646	0.660	0.650	21.0
RoBERTa-base	0.630	0.629	0.644	0.635	7.54

Table 2: Results on sentiment classification where the F1 is performances of the model on test-data provided by Sentimix. The models with LM-backbones are provided with the perplexity of the fine-tuned LM where as the ones without are denoted by NA.

on a pre-processed unsupervised twitter dataset over 25,000 iterations. It was trained for a total of 3 epochs. Training batch size was four and vocab size 50265. The perplexity of this model was 7.54.

4.6 DistilBERT

DistilBERT (Sanh et al., 2019) uses the technique of knowledge distillation to improve the performance of BERT and create a smaller distilled version of the model. The LM-backbone was fine-tuned on a pre-processed unsupervised twitter dataset over 49,000 iterations. It was trained for a total of 6 epochs. Training batch size was four and vocab size 28996. The perplexity of the fine-tuned LM-backbone for distilBERT was 6.51 and the base model used for fine-tuning the LM was *distilbert-base-cased*.

4.7 Ensemble

For the final submissions, three variations of BERTs and two variations of DistilBERT were used. These were the top 5 selected based on their validation accuracy. For the ensemble, Weighted Majority Voting, by using the prediction confidence (0 to 1 scale) as the weight; The ensemble methodology and its usage in our case is described in Figure 1.

5 Results

The result for the experiments are summarized in Table 2. Out of all the techniques used on test-data, Weighted majority vote ensemble with LR funneling gained a significant edge when it comes to F1 score. Traditional machine learning models like NB-SVM show a comparative performance.

6 Future Work

There are three main incremental directions of improvements: data, methods, adopting techniques from text classification. For instance, initial tweet data had a lot of truncated tweets, using tweet_ids to get an entire tweet would enrich our inputs. The training data can also be augmented in a wide variety of ways such as using vector similarity (Ma, 2019).

We can investigate other methods which might help in understanding missed case. Sentence embeddings for Hinglish, similar to InferSent (Conneau et al., 2017) or Universal Sentence Encoding (Cer et al., 2018) may be promising, in addition to Skip Thought or other sentence vectorisation methods, as well as exploring the performance of models which do not focus on transfer learning like R-CNN, and LSTMs.

Lastly, a wide variety of deep learning tricks and methods could be used, such as label smoothing (Müller et al., 2019), which can help in generalising better beyond the small training sample.

7 Conclusion

We demonstrate that ensembles of classical Machine Learning models, even NB-SVM exhibit competitive performance and can in fact be better than some Transformer baselines. It is still worthwhile to implement simple classical baselines. Additionally, we hope that the released dataset and models [2] will encourage readers to investigate this further.

References

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2016. Quasi-recurrent neural networks. *CoRR*, abs/1611.01576.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal sentence encoder. *CoRR*, abs/1803.11175.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *ArXiv*, abs/1705.02364.

Ellen Contini-Morava. 1995. Duelling languages: Grammatical structure in codeswitching. *Journal of Linguistic Anthropology*, 5(2):246–247.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jeremy Howard and Sebastian Ruder. 2018. Fine-tuned language models for text classification. *CoRR*, abs/1801.06146.

Paul Jaccard. 1912. The distribution of the flora in the alpine zone.1. *New Phytologist*, 11(2):37–50.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *CoRR*, abs/1808.06226.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Edward Ma. 2019. Nlp augmentation. `https://github.com/makcedward/nlpaug`.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? *CoRR*, abs/1906.02629.

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.

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.

[2] https://github.com/NirantK/Hinglish

Koustav Rudra, Shruti Rijhwani, Rafiya Begum, Kalika Bali, Monojit Choudhury, and Niloy Ganguly. 2016. Understanding language preference for expression of opinion and sentiment: What do Hindi-English speakers do on twitter? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1131–1141, Austin, Texas, November. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Taehoon Kim and Kevin Wurster. 2019. emoji.

Sida Wang and Christopher D. Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, page 90–94, USA. Association for Computational Linguistics.

zaxcie. 2018. Nb-svm: Strong linear baseline.