Semi-Supervised and Unsupervised detection of Humour in Code-Mixed Hindi-English Tweets

Chakita Muttaraju, Aakansha Singh, Anusha Kabber, Mamatha H R
Dept. of CSE.
PES University
Bangalore, India
{chakitapesu, sudhasingh538, anushakabber}@gmail.com,mamathahr@pes.edu

Abstract

A significant portion of social media consists of code-mixed data, as the number of users from India continues to grow rapidly. The phenomenon of mixing words belonging to different languages in conversations is referred to as code-mixing. As we continue to advance our social networks, it becomes imperative to accurately classify posts so they may be seen by a wider, more appropriate audience. Classification becomes harder in light of a substantial lack of labeled Hindi-English ground-truth data - owing to the inconvenience of human annotation and the relative difficulty of scraping. Supervised methods tend to suffer from such a deficiency of labeled data, especially SOTA models which require a considerable number of labeled examples to give good results. Hence, this paper outlines a novel semi-supervised method that can be used for binary classification of humorous Hindi-English code-mixed data. The GAN-BERT architecture (Croce et al., 2020) is modified to optimize results for code-mixed data. The paper also contrasts this method with various unsupervised techniques. We look into different embedding techniques such as LASER, FastText, and BERT for unsupervised classification. Fine-tuned Hinglish BERT integrated into the GAN-BERT architecture surpassed all other methods on the test set with an accuracy of 87.5%.

1 Introduction

In a multilingual society, the usage of code-mixed languages is a common occurrence. A significant part of the content available on social media is code mixed. This code-mixed data is a challenge in the field of natural language processing because its characteristics completely vary from the traditional structures of standard languages. This makes the processing of such content significantly harder. Humor Detection has been one of the most intriguing problems in Natural Language Processing as it requires a deep semantic understanding of the text. Most past research has been focused on detecting humor in unmixed languages but owing to the tremendous amount of code-mixed data available online there is a need to develop ways to detect humor in code- mixed data as well. We are also aware that obtaining labeled data for any task is a costly and time-consuming process. A viable solution to this problem is adopting a semi-supervised approach to identify the patterns even in a small dataset. One such semi-supervised method is implemented within the Semi-Supervised Generative Adversarial Network BERT (SS-GAN-BERT) .(Croce et al., 2020) The model takes a combination of labeled and unlabelled data as input where the proportion of labeled data is significantly smaller than the unlabelled samples. Here, a generator produces "fake" examples resembling the data distribution, while BERT is used as a discriminator.

In this work, we explore semi-supervised and unsupervised approaches for detecting humor in code-mixed languages. The semi-supervised method deals with modifying the current GAN-BERT architecture by replacing the BERT model with multiple specialized, regional language-based BERT models. This also adds a novel aspect to our study since this kind of approach has not been used before to the best of our knowledge. Additionally, for the semi-supervised methods, this paper also experiments with semi-supervised SGD. Unsupervised methods include obtaining sentence embeddings of Hindi-English code mixed data and clustering to classify them into humorous and non-humorous sections.

2 Related Work

Many researchers and practitioners from industry and academia have been attracted to the problem of text classification of code-mixed languages and humor detection. (Arora, 2020)proposed pretraining ULMFiT on synthetically generated code-mixed data, generated by modeling code-mixed data gen-

eration as a Markov process using Markov chains. (Gautam et al., 2021) used pre-trained models, fine-tuned on English-only tasks, and fine-tuned these models on translated code mixed datasets and achieved state-of-the-art results. To translate English-Hindi code mixed data to English, mBART was used. Here, words were transliterated informally without any standard rules and no formal data sources were used. (Yadav and Chakraborty, 2020) provides an experimental analysis of logistic regression, naive Bayes, decision tree, random forest, and SVM on our code-mixed data for classification tasks so as to create a benchmark for further research. They have also created a corpus for Dravidian languages in the context of sentiment analysis and offensive language detection tasks. (Conneau et al., 2019) has shown that cross-lingual embeddings can be made in a totally unsupervised way, i.e. they only require monolingual embeddings of the respective languages

3 Dataset

This dataset is populated with Hindi-English code mixed tweets scraped from Twitter. It is a subset of the dataset mentioned in the paper (Khandelwal et al., 2018). The creators of the original dataset used Python's twitterscraper to build the corpus. The tweets were annotated manually. Facts were automatically considered non-humorous and insults, irony, jokes, and anecdotes were labeled as humorous. An agreement of 0.821 Fleiss' and Kappa score for inter-annotator measure was achieved while annotating this dataset. The makers made sure to keep a good mix of topics in the dataset as they did not want the tweets to be domain dependent and the classification to be based upon semantic differences.

The original dataset, as available today, has a collection of tweet IDs without the tweet text. A number of tweets from this dataset have been removed from the platform. With the use of Twitter API v2, we were able to retrieve the text of the remaining tweets for training.

For the semi-supervised approach, the data was divided into a ratio of 1:100 for labeled vs unlabelled tweets as suggested by the authors of the original GAN-BERT paper(Croce et al., 2020). We used 46 labeled tweets and 4616 unlabelled tweets from the dataset. For testing, we used another 296 tweets. We attempted to keep an even distribution of humorous vs non-humorous tweets in each of the three sections: labeled, unlabelled, and test data. For unlabelled tweet data, the labels of tweets available in the corpus were removed and replaced with a placeholder 'UNK'. For the unsupervised approach, all 4662 tweets were stripped of their labels and clustered. Once more, 296 tweets were used as the test set.

4 Proposed Method

We are well aware that code mixed languages are under-resourced and obtaining annotated data for them is a costly process. In this work, we explore techniques that rely more on unlabeled data, which is easier to procure, and hence we utilize various semi-supervised and unsupervised models for the task of humor detection in code-mixed tweets. For semi-supervised models, we ran experiments by varying the BERT models, epochs, and dropout rate whereas for unsupervised models we try combinations for different word embeddings with various clustering algorithms. '

4.1 Semi-Supervised Method

We utilize the GAN-BERT architecture (Croce et al., 2020). As specified in their paper, we use a ratio of 1:100 as the ratio between labeled and unlabelled examples. The Generator component generates a set of fake samples from a given noisy distribution. The unlabelled and labeled samples are vectorized using the BERT model. The fake samples along with vectors for the unlabelled and labeled samples are fed in as input to the discriminator component which then learns to classify the data as fake or as belonging to one of the labels. As in SS-GANs(Khanuja et al., 2021), the labeled material is initially used to train the discriminator, i.e., the BERT model, and it is trained to differentiate between generated and real samples. Additionally, the discriminator is used to label or classify the real samples. In our case, these classes would be humorous vs. non-humorous. For the purpose of this work, we substitute the standard BERT model (Devlin et al., 2018) with multilingual BERT models such as IndicBERT (Kakwani et al., 2020) and MuRIL (Khanuja et al., 2021). Additionally, we use BERT models pre-trained and fine-tuned on Hinglish data such as HinglishBERT (verloop, 2021)(ketan rmcf, 2021) and HingBERT (l3cube pune, 2021). We trained the modified GAN-BERT on a code-mixed dataset with a number of different optimizations to improve performance, including

testing multiple different BERT models to replace the original BERT in the GAN-BERT architecture. Number of epochs, batch size, and dropout rate were also experimented with to achieve the best performance.

IndicBERT IndicBERT is a version of ALBERT (Lan et al., 2019) for Indian languages. It is a multilingual language model trained on a huge corpus of some of the most popular Indian languages - Hindi, Kannada, Bangali, Tamil, Telugu, and many more. IndicBERT is said to give state-of-the-art performances for multiple language tasks in regional languages. It also uses much fewer parameters compared to models like XLM-R and mBERT. We use IndicBERT under the assumption that IndicBERT will perform better than BERT in the case of code-mixed Hindi-English data, the latter having been purely trained in English.

We found that IndicBERT does perform better than BERT with an approximate accuracy improvement of 10%. While this is true, for our specific dataset, IndicBERT does not outperform every other model chosen in this paper. This can be attributed to the fact that IndicBERT was trained on large corpora of Indian languages in their original scripts and none of the data was code-mixed. These corpora were not transliterated.

We have also seen IndicBERT achieve the highest accuracy at around 10 training epochs and a dropout rate of 0.2.

MuRIL MuRIL was assumed to be an improvement from the IndicBERT model as it is trained on transliterated Indian languages. MuRIL is a BERT model that is trained on 17 different Indian languages and their transliterated counterparts. The model is trained similarly to multilingual BERT only that it uses an exponent value of 0.3 and not 0.7 for upsampling which is shown to enhance low-resource performance.

It was observed that MuRIL did marginally better than IndicBERT at its highest point. An interesting factor for MuRIL was that the accuracy did not drop as the number of epochs increased and it maintained a respectable, fairly high accuracy when quite a few of the other models started to overfit and perform worse.

HinglishBERT These models are BERT models specifically trained on Hindi-English code-mixed data. Furthermore, this data was also scraped from code-mixed data from Twitter. The authors train and then fine-tune these models on hundreds of thousands of code-mixed tweets from a Twitter stream.

The fine-tuned HinglishBERT was by far our best model for classifying tweets when trained in a generative adversarial setting. It outperformed our second-best model by approximately 7%. We also observed that this model starts to overfit and perform worse after training for around 25 epochs or more. We attribute it to the dataset HinglishBERT was trained on. This also reinforces the authors' statement that declares fine-tuning on code-mixed data improves the model's performance as HinglishBERT did not perform quite as well as fine-tuned HinglishBERT did by quite a large margin.

HingBERT HingBERT is a BERT model that is pre-trained on a corpus that is the first of its kind with 52.93M Hindi-English code-mixed sentences. As expected, HingBERT performed similarly to HinglishBERT and better than IndicBERT with a lot of similar data sources compared to our dataset.

4.1.1 Semi-supervised SGD

In this work, we have explored another semi-supervised classification technique implemented by the self-training algorithm. In this method, we first train a classifier on the available labeled observations. After this, we use the obtained classifier to predict the classes of unlabeled samples. From these, we pick the observations that satisfy particular criteria like prediction probability and use these as 'pseudo-labels' along with the labeled data for training a new supervised model. We repeat the process for a certain number of iterations or till we run out of labeled data. We have used a Semi-supervised SGD classifier in combination with different embeddings like TF-IDF and FastText. Of these embedding methods, TF-IDF gave the maximum accuracy of 78.5% whereas FastText gave 72.9%.

The results of the experiments are detailed in the Results section of this paper.

4.2 Unsupervised Method

This paper addresses the problem of the unavailability of labeled data by choosing to use semi-supervised and unsupervised methods. Sentence and word embeddings are commonly used to obtain clusters of different classes in unsupervised language classification tasks. Here, we converted our unlabeled dataset of 4600 tweets to sentence

embeddings to then cluster them according to di ferent clustering algorithms. We explore three di ferent models that normally provide multilingua embeddings, but in this case are used to obtain em beddings for Hindi-English code-mixed data: 1 LASER, 2) FastText, 3) sentencetransformer-ber hinglish.

LASER stands for Language-Agnostic Sentenc Representations (Artetxe and Schwenk, 2019). A the name suggests these embeddings use a singl model for a variety of languages. Since code-mixe languages come under the category of the so-calleu low resource languages, LASER embeddings can be used for obtaining a vector representation for them.

FastText is a subword level embedding based on the skipgram model of word2vec (Bojanowski et al., 2016). Since code-mixed languages are riddled with spelling variations and inconsistencies and also there can code-mixing at the subword level, FastText seems to be a good choice for the sentiment analysis task.

The sentencetransformer-bert-hinglish generates a Transformer based representation for a low-resource language using existing representations in another high-resource language (Reimers and Gurevych, 2019).

We also compare the performance of different clustering algorithms, namely, K-Means clustering, Spectral Clustering, and Agglomerative Clustering for the sentiment analysis task.

5 Results

Multiple BERT models were used in the GAN-BERT architecture to construct semi-supervised methods. Additionally, semi-supervised SGD was also experimented with. Results from trials of unsupervised methods dealing with the clustering of sentence embeddings are also included. As expected, the semi-supervised methods performed better with Fine-tined HinglishBERT in the GAN-BERT architecture performing the best with 87.5% accuracy.

Semi-supervised methods performed better for a number of reasons - the models used to generate sentence embeddings were not specifically trained on Hindi-English code mixed data. Also, simple clustering based on humor, a fairly abstract concept, would not perform as well as the more customised, sophisticated semi-supervised methods used in this paper. Experiments based on unsupervised methods were still utilised for comparison to provide the reader an idea of how well the semi-supervised methods perform.

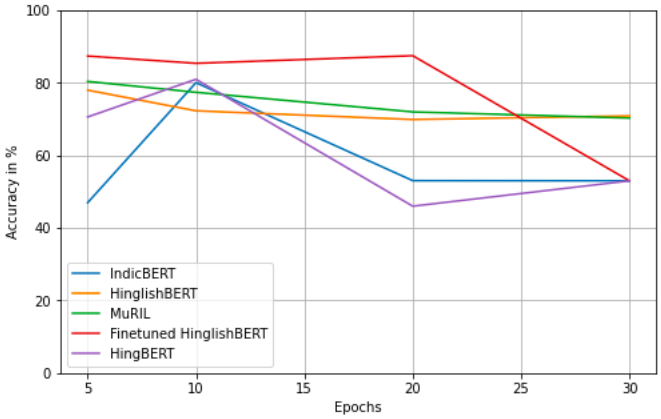

Figure 1: Accuracy improvement w.r.t increasing epochs.

5.1 Semi-supervised Methods

As mentioned before, we primarily ran experiments on the training of the GAN-BERT model on our dataset by the following processes: 1) Varying BERT models more suited to our specific language task., 2) Epochs, 3) Dropout rate.

Throughout these experiments, the training size was 4616 unlabeled tweets, and 46 labeled tweets and the test set was 296 tweets long.

The models were trained on a number of epochs and as seen in Figure 1, increasing the number of epochs did not improve performance in most cases. Fine-tuned HinglishBERT performed the best with an accuracy of 0.875 at 20 epochs.

In Table 1, models and their highest accuracy across epochs are given.

Table 1: Models and Accuracy

Model	Accuracy
BERT	0.689
Fine-tuned HinglishBERT	**0.875**
HinglishBERT	0.780
HingBERT	0.810
MuRIL	0.804
IndicBERT	0.800

We experimented with various dropout rates and found that these dropouts work best with the corresponding models as shown in Table 2.

Semi-supervised SGD resulted in an accuracy of 78.5% with TF-IDF as shown in Table 3.

Table 2: Models, Dropout Rate and Accuracy

Model	Dropout Rate	Accuracy
BERT	0.2	0.689
Fine-tuned HinglishBERT	**0.09**	**0.875**
HinglishBERT	0.7	0.780
HingBERT	0.1	0.810
MuRIL	0.09	0.804
IndicBERT	0.2	0.800

Table 3: Semi-supervised SGD and embeddings

Embedding	Accuracy
TF-IDF	**0.785**
FastText	0.729

5.2 Unsupervised Methods

For unsupervised methods, we obtained sentence embeddings from three different models: 1) LASER, 2) FastText, 3) sentencetransformer-bert-hinglish

and clustered these embeddings with three different clustering algorithms: 1) K-Means, 2) Spectral Clustering, and 3) Agglomerative Clustering.

The results of K-Means on the three different embeddings are shown in Figure 2.

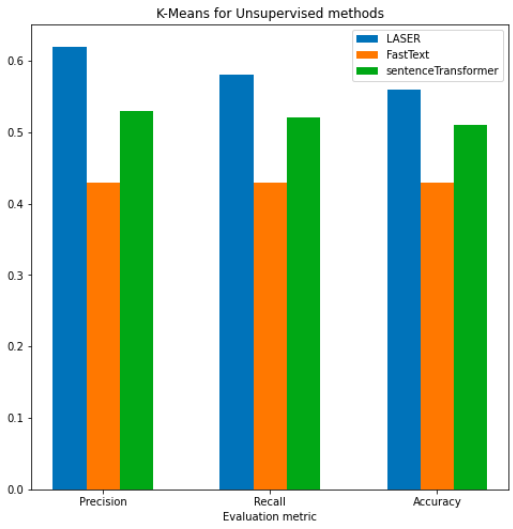

Figure 2: K-Means performed on the three different sentence embeddings obtained from three different models.

As can be observed, LASER had the highest average precision, recall, and accuracy when K-Means clustering was used. FastText easily performed the worst, getting accuracy scores of less than 0.5.

Spectral Clustering and its results are shown in Figure 3.

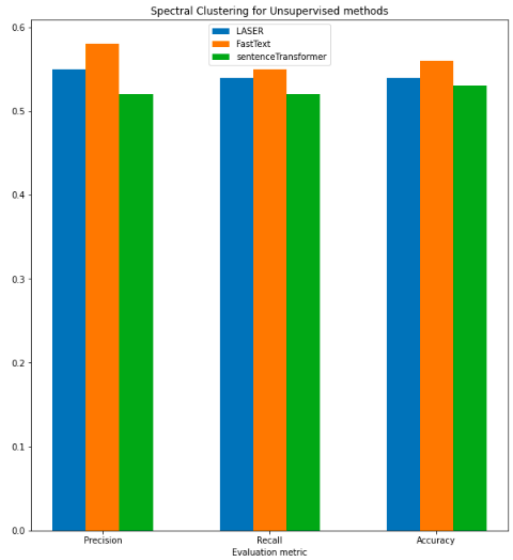

Figure 3: Spectral clustering performed on the three different sentence embeddings obtained from three different models.

With Spectral clustering, FastText had the upper hand but only slightly. The sentence transformer accuracy was barely above 0.5 for spectral clustering.

Agglomerative clustering showed a different story as shown in Figure 4.

LASER performed abysmally while sentence transformer and FastText stayed with a similar range of accuracy. No real improvement was shown.

Unsupervised methods, on a whole, did not achieve very high performance with the highest accuracy being LASER embeddings clustered with K-Means with an accuracy of 0.62.

Conclusion

We explored semi-supervised and unsupervised methods with the intent of classifying tweets as humorous or non-humorous. For semi-supervised methods, we chose to train different models in a generative adversarial setting similar to SS-GANs. We also experimented with a number of different parameters to get the highest accuracy possible. With unsupervised methods, we chose some of the most popular models to obtain multilingual sentence embeddings and clustered the embeddings with three different clustering algorithms. We found that semi-supervised methods outperform largely with Fine-tuned HinglishBERT leading the race with an accuracy of 0.875.

This research could be extended in a number of

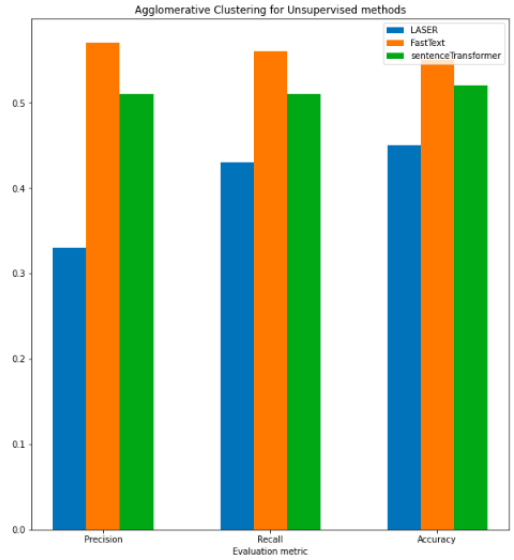

Figure 4: Agglomerative clustering performed on the three different sentence embeddings obtained from three different models.

ways. The length of the labeled and unlabeled sets is kept constant throughout the experiments. The number of labeled and unlabeled tweets could be increased to possibly achieve better accuracy. The ratio of the test set to train could be varied for a more rounded analysis.

References

Gaurav Arora. 2020. Gauravarora@ hasoc-dravidian-codemix-fire2020: pre-training ulmfit on synthetically generated code-mixed data for hate speech detection. *arXiv preprint arXiv:2010.02094*.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information.

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. *arXiv preprint arXiv:1911.02116*.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Devansh Gautam, Kshitij Gupta, and Manish Shrivastava. 2021. Translate and classify: Improving sequence level classification for english-hindi code-mixed data. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 15–25.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online. Association for Computational Linguistics.

ketan rmcf. 2021. Fine-tuned HinglishBERT model on Huggingface.

Ankush Khandelwal, Sahil Swami, Syed S Akhtar, and Manish Shrivastava. 2018. Humor detection in english-hindi code-mixed social media content: Corpus and baseline system. *arXiv preprint arXiv:1806.05513*.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages.

l3cube pune. 2021. HingBERT model on Huggingface.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

verloop. 2021. HinglishBERT model on Huggingface.

Siddharth Yadav and Tanmoy Chakraborty. 2020. Unsupervised sentiment analysis for code-mixed data. *arXiv preprint arXiv:2001.11384*.