C1 at SemEval-2020 Task 9: SentiMix: Sentiment Analysis for Code-Mixed Social Media Text using Feature Engineering

Laksh Advani and **Clement Lu** and **Suraj Maharjan**
Capital One
1600 Capital One Drive, McLean, VA
firstname.lastname@capitalone.com

Abstract

In today's interconnected and multilingual world, code-mixing of languages on social media is a common occurrence. While many Natural Language Processing (NLP) tasks like sentiment analysis are mature and well designed for monolingual text, techniques to apply these tasks to code-mixed text still warrant exploration. This paper describes our feature engineering approach to sentiment analysis in code-mixed social media text for SemEval-2020 Task 9: SentiMix. We tackle this problem by leveraging a set of hand-engineered lexical, sentiment, and metadata features to design a classifier that can disambiguate between "positive", "negative" and "neutral" sentiment. With this model we are able to obtain a weighted F1 score of 0.65 for the "Hinglish" task and 0.63 for the "Spanglish" tasks.

1 Introduction

Social media has grown exponentially in the last decade and has given rise to multilingual online communication and the consumption of media. Code Mixing is the phenomenon of embedding linguistic units such as phrases, words, or morphemes of one language into an utterance of another. Code mixing is prevalent in online discourse, where bilingual speakers mix English with their native language. For instance, bilingual speakers of Hindi and Spanish languages, which are the 3rd and 4th most spoken languages in the world respectively, frequently mix English with Hindi and Spanish in spoken language as well as online social media. The growing presence of "Hinglish" and "Spanglish" in social media is evidence that this is a growing area of research as traditional NLP tasks like sentiment analysis heavily rely on monolingual resources.

While there are many cutting edge techniques to apply common NLP tasks to monolingual text, the task of sentiment analysis, in particular, has not been explored for multilingual code-mixed texts. Conventional NLP tools make use of monolingual resources to operate on code-mixed text, which limits them to properly handle issues like word-level code-mixing. Additionally, code-mixed texts typically exist on social media which has a conjunction of other features like incorrect spelling, use of slang, and abbreviations to name a few.

While code-mixing is an active area of research, correctly annotated data is still scarce. It is particularly difficult to mine a small subset of tweets that may or may not be code-mixed and then apply a sentiment score to them. The organizers have solved this by releasing around 20,000 tweets (Patwa et al., 2020) with unique sentiment scores and word-level tagging of the language. For evaluation and ranking, the organizers used a weighted F1 score across "positive", "negative" and "neutral" classes.

While deep learning techniques are state of the art for this task, the contribution of this paper is to demonstrate how content-rich features, when applied to a classifier, can give us strong and comparable results. Some of the features we explore are lexical features like n-grams, metadata features like word frequency, and sentiment-based features like profanity count.

*Our codalab username is *lakshadvani*.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1227–1232
Barcelona, Spain (Online), December 12, 2020.

2 Related Work

Code-Mixed Sentiment Analysis is an active field of research. For monolingual sentiment analysis, Recurrent Neural Networks (RNN) and other more complex deep learning models have been successful. Socher et al. (2013) gave us a significant breakthrough by using compositional vector representations. With regards to Hindi-English Sentiment Analysis, the shared task SAIL-2015 (Patra et al., 2015) reported accuracies of different systems submitted on sentiment analysis for tweets in three major Indian languages: Hindi, Bengali, and Tamil. Recently Joshi et al. (2016) released a dataset sourced from Facebook and proposed a deep learning approach to Hindi-English Sentiment extraction. A variety of traditional learning algorithms like Naive Bayes, Support Vector Machines and Decision Trees were applied to this dataset. The winning algorithm from the SAIL-2015 shared task was a Naive Bayes based system. More recent work by Hashimoto et al. (2017) involves a hierarchical multitask neural architecture with the lower layers performing syntactic tasks, and the higher layers performing the more involved semantic tasks while using the lower layer predictions.

3 Dataset

Word	Word Tag	Word	Word Tag
@	O	Nobody	Eng
nsfw	Hin	can	Eng
_	O	make	Eng
hs	Hin	gabby	NE
Maybe	Hin	laugh	Eng
I'll	Eng	like	Eng
faing	Eng	I	Eng
and	Eng	do	Eng
I	Eng	☺	O
won't	Eng	asta	Es
feel	Eng	parece	Es
a	Eng	mongolita	Es
thing	Eng	lmao	Eng
.	O		

Table 1: Sample Annotation Examples for Hinglish and Spanglish Dataset

For the Hinglish dataset, the organizers have released a total of 20,000 manually labeled tweets with 14,000 being in the training dataset, 3,000 being part of the validation dataset and 3,000 being part of a holdout test dataset. Additionally, for the Spanglish dataset, they released 12,002 tweets for the training dataset, 2,998 tweets for the validation dataset and 3,788 tweets for the test dataset.

To ensure the accuracy of the dataset labels, the organizers employ a semi-automatic annotation methodology. They initially use word-level identifiers and then baseline sentiment analysis to obtain the initial labels. Subsequently, manual evaluation is done by at least two annotators, tweets with low confidence are discarded. While this seems to be a robust system in some cases the word tags are incorrect with words like 'Maybe' being tagged as a Hindi word, when we can safely assume that it is from the English language. Additionally, the dataset contains URLs, hashtags, usernames and emoticons. The average length for tweets in the "Hinglish" dataset is 26 tokens and 15 tokens for the "Spanglish" dataset.

Table 1 shows us two examples from the "Hinglish" and "Spanglish" datasets made available for this task. For both subtasks the organizers provided labeled trial, train and validation datasets for developing candidate models.

The data was provided in the CONLL format with each tweet having a sentiment of "positive", "negative" or "neutral". Figure 1 gives us an overview of the class distribution of these datasets. As we can see the "Spanglish" dataset is unbalanced with the majority of the samples being in the "positive"

and "neutral" datasets. From the examples in Table 1 we can see that the organizers have also provided word level information, with the classes *Eng (English)*, *Es (Spanish)*, *Hin (Hindi)*, *mixed*, and *univ* (*e.g.*, universal symbols, @ mentions, hashtags).

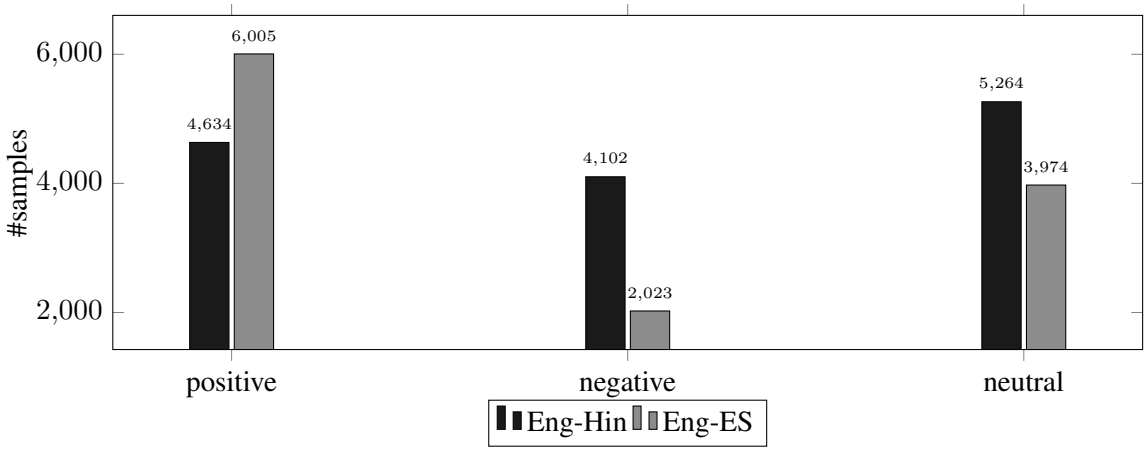

Figure 1: Categorical Label Frequency in Hinglish and Spanglish Training Dataset.

4 Methodology

We extracted different hand engineered features like word n-grams, sentiment polarity, punctuation frequency, word frequency, emoji frequency and profanity level from tweets. We weighed n-gram features using their term frequency - inverse document frequency (TF-IDF) scores. We then built a Logistic Regression classification model using Scikit-learn (Pedregosa et al., 2011) library.

4.1 Pre-processing

Prior to using the datasets we explored a variety of pre-processing techniques to reduce the noise in the data as they are obtained from online social media and include additional textual features. We used the following pre-processing techniques.

- **Abbreviation Expansion**: In many cases users on online social media tend to use informal slang and abbreviations. For instance, we expanded common abbreviations like "DM" to "direct message" using the Python regular expression library.
- **Repetition**: As communication on social media websites is informal, there are cases where letters in words are repeated such as "heelloo". We normalize these words by using the Python regular expressions library.
- **Usernames and Handles**: We use the Python regular expression library to develop a manually defined function to remove user handles for example (@Matt), URLs (`https:twitter.com`) and similar links to external images.

4.2 Hand-crafted Features

We explored and used different hand-crafted features to build our machine learning model. Our hand-crafted features are described below:

- **Lexical Features**: We extracted word n-grams ($n = 1, 2, 3$) from the tweets as they are strong lexical representations (Cavnar et al., 1994; McNamee and Mayfield, 2004; Sureka and Jalote, 2010). A word n-gram is simply a sequence of n words, for example "hello again" is a bi-gram or 2-gram.
- **Sentiment Lexicon Features**: We extracted the sentiment polarity for English language tokens using Valence Aware Dictionary and sEntiment Reasoner (VADER) (Hutto and Gilbert, 2015).

VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. This feature provides strong clues about the sentiment of the tweets by explicitly discriminating "positive" or "negative" words in the tweet.

- **Emoji Features**: Emoji are frequently used in online discourse and give us a hint about the tone of the tweets. We used a rule-based dictionary to formalize their polarity as "happy", "sad", and "neutral". We constructed this lexicon using the Emoji Sentiment Ranking (Kralj Novak et al., 2015).
- **Profanity Features**: Profane words are correlated with negative connotations. We used the presence or absence of profane words as a feature to discriminate between the three classes.
- **Tweet Metadata Features**: Finally we looked into using metadata about the tweet. We extracted repetition, punctuation count, and length and used them as features.

4.3 Experimental Setup

We used the training and validation splits for Spanglish and Hinglish datasets to build our final classifier. We experimented with standard machine learning classifiers like Logistic Regression, Support Vector Machines, Random Forests with our hand-crafted features defined in Section 4.2. We tuned the C hyper-parameter of the Logistic Regression and Support Vector Machine using an extensive grid search over the range of values $\{10e^{-2}, \ldots, 10\}$ on the validation dataset. For our final model, we used the best hyperparameter value on the validation dataset. After a number of trials by using a step size of 0.01 we found that 0.9 was the best value to use as it gave us the best results. Additionally, based on the size and type of the data we selected the "Liblinear" solver, which works in a one vs. rest fashion for multi-class tasks.

Moreover, we ran experiments using deep learning models. For deep learning models, we used BERT (Devlin et al., 2019), ELMo (Peters et al., 2018), and GloVe (Pennington et al., 2014) embeddings to initialize the embedding layer, Bidirectional Gated Recurrent Units (Cho et al., 2014) for sequence to sequence encoding, and used self-attention (Lin et al., 2017) to reduce the sequence of encoded vectors to a single vector for representing the tweet. The tweet representations were then used for classification. Additionally, we used the Adam optimizer to train our deep neural network. To select our top candidate we evaluated these models based on the F1 score.

5 Results and Analysis

Model	Hinglish F1 Score	Spanglish F1 score
Logistic Regression	**0.58**	**0.55**
Support Vector Machines	0.49	0.50
Feed-forward Networks	0.56	0.53
Random Forest	0.48	0.46
BERT Language Model	0.56	0.48
GloVe + ELMo Language Model	0.56	0.54
Organizer Baseline	0.58	0.49

Table 2: Model Results for Hinglish and Spanglish Validation dataset

Table 2 shows us the results of a variety of classification models applied to the Hinglish and Spanglish validation datasets. As we can see the "Logistic Regression" model performed the best giving us a F1 score of 0.58 for the Hinglish validation dataset and 0.55 for the Spanglish validation dataset. It is interesting to see that the use of traditional hand-engineered features performed better than the state of the art deep learning approaches. We suspect that the limited amount of data negatively affects deep learning approaches and makes it conducive to use linear classification models. As a result, we chose to use the "Logistic Regression" classifier on the final dataset.

After the release of the test dataset, we merged the training dataset with the validation dataset and trained our model with the best hyperparameters. For the Hinglish test dataset, we got a significantly higher score of 0.65, which matches the organizer baseline of 0.65. Additionally, for the Spanglish dataset, we got a score of 0.63 which is a lower than the organizer baseline of 0.65.

5.1 Error Analysis

Text	Gold	Prediction
Kameeny loogo ko justice system sy wessy he nikal deena chah-eye khud dafa hona bhi theek hy .	Negative	Negative
Dear all of every one so I am help you me Vishnu Sharma gwalior madhyapradesh se hoo My mere teen better he Jo.	Positive	Positive
'bht moody ho aajkal . hope all is well	Negative	Neutral
'receptionist sit wherever you ' d like me thank you I ' ll be in my car'	Neutral	Neutral
I'm poor poor-ever happy kase God is with 😇	Negative	Positive

Table 3: Sample Predictions from the Hinglish Validation Dataset

Table 3 shows us some of the label predictions from the dataset. In many cases, words associated with profanity and emoticons help the model decide between classes like positive and negative but these do not help the model distinguish between "neutral" and "positive" or "negative" tweets as we do not have a list of words specific to a "neutral" class. As we can see in the third row in Table 3 our model gives us an insight into incorrectly classified results. The model gives us a prediction of "Neutral" where the true label is "Negative". As seen in the last row of Table 3 when the tweet is ambiguous the emoji contributes significantly to the prediction. In this case, our prediction is "Positive" but the gold label is "Negative". We suspect the presence of emoji with positive polarity signaled the model to predict "Positive" label. While this is an incorrect prediction we can perhaps infer that the dataset has incorrectly labeled samples.

6 Conclusions and Future Work

In this paper, we demonstrated our system for performing sentiment analysis on code-mixed tweets. We used a variety of lexical, metadata, and sentiment-based features. We used a Logistic Regression Classifier with these features to classify the tweets as "positive", "negative" and "neutral". We demonstrate how a lightweight and memory-efficient model with prior extracted features can be competitive with state of the art Language Models. In the future, we would look into combining hand-engineered lexical, sentiment, and metadata features with the representations learned from Convolutional Neural Networks (CNN) and Bidirectional Gated Recurrent Unit (Bi-GRU) having attention model applied on top (Kar et al., 2017). Recent developments in NLP research have pointed to the combination of deep learning representations as a strong approach to gain better results.

References

William B Cavnar, John M Trenkle, et al. 1994. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval.*

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

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.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, Copenhagen, Denmark, September. Association for Computational Linguistics.

C.J. Hutto and Eric Gilbert. 2015. Vader: A parsimonious rule-based model for sentiment analysis of social media text. 01.

Aditya Joshi, Ameya Prabhu, Manish Shrivastava, and Vasudeva Varma. 2016. Towards sub-word level compositions for sentiment analysis of Hindi-English code mixed text. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2482–2491, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Sudipta Kar, Suraj Maharjan, and Thamar Solorio. 2017. RiTUAL-UH at SemEval-2017 task 5: Sentiment analysis on financial data using neural networks. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 877–882, Vancouver, Canada, August. Association for Computational Linguistics.

Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. 2015. Sentiment of emojis. *PLOS ONE*, 10(12):1–22, 12.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *ICLR (Poster)*. OpenReview.net.

Paul McNamee and James Mayfield. 2004. Character n-gram tokenization for european language text retrieval. *Information Retrieval*, 7(1-2):73–97.

Braja Gopal Patra, Dipankar Das, Amitava Das, and Rajendra Prasath. 2015. Shared task on sentiment analysis in indian languages (sail) tweets - an overview. In Rajendra Prasath, Anil Kumar Vuppala, and T. Kathirvalavakumar, editors, *Mining Intelligence and Knowledge Exploration*, pages 650–655, Cham. Springer International Publishing.

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.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October. Association for Computational Linguistics.

Ashish Sureka and Pankaj Jalote. 2010. Detecting duplicate bug report using character n-gram-based features. pages 366–374, 11.