TueMix at SemEval-2020 Task 9: Logistic Regression with Linguistic Feature Set for Sentiment Analysis of Code-Mixed Social Media Text

Elizabeth Bear **Diana Constantina Höfels** **Mihai Manolescu**
{elizabeth.bear, diana-constantina.hoefels, mihai.manolescu}
@student.uni-tuebingen.de
Eberhard-Karls University of Tübingen

Abstract

Commonly occurring in settings such as social media platforms, code-mixed content makes the task of identifying sentiment notably more challenging and complex due to the lack of structure and noise present in the data. SemEval-2020 Task 9, SentiMix, was organized with the purpose of detecting the sentiment of a given code-mixed tweet comprising Hindi and English. We tackled this task by comparing the performance of a system, TueMix - a logistic regression algorithm trained with three feature components: TF-IDF n-grams, monolingual sentiment lexicons, and surface features - with a neural network approach. Our results showed that TueMix outperformed the neural network systems and the addition of the linguistic features beyond TF-IDF n-grams enhanced our performance, yielding a weighted F1-score of 0.685.

1 Introduction

The task of sentiment analysis with code-mixing is gaining more and more attention, given that on social media platforms, bilingual or multilingual individuals create content by mixing two or more different languages within a single sentence. Code-mixing occurs when phrases, words, and morphemes of one language are inserted into a sentence of another language (Sridhar and Sridhar, 1980). People may code-mix to make themselves appear noteworthy or imply a certain social status (Shbat, 2007), to show solidarity, or simply to express their feelings (Holmes, 2008).

A substantial number of individuals combine English and Hindi on social media platforms, and as a result, the need to process code-mixed content has become essential if we want to build robust and competitive systems for sentiment detection. Sentiment analysis of code-mixed text faces multiple challenges due to the lack of a formal grammar for the hybrid code-mixed generated language and the existing characteristics of multilingual content where there are atypical variations in spelling, ungrammatical structures, and transliteration (Bali et al., 2014).

The International Workshop on Semantic Evaluation, or SemEval, addressed this challenge by proposing the shared task SentiMix: Sentiment Analysis for Code-Mixed Social Media Text (Patwa et al., 2020). The training data provided contained 14000 Hindi-English tweets annotated with the corresponding sentiment label (positive, neutral, or negative), and with additional language labels at the word-level (English (Eng), Hindi (Hin), emoji (EMT), or universal (0) for other tokens). The class distribution was fairly balanced with 33.1% positive, 37.6% neutral, and 29.29% negative tweets.

Our strategy consisted of a machine learning and a neural network-based approach. Our main system, TueMix, used a logistic regression algorithm trained with three categories of features: TF-IDF n-grams, monolingual sentiment lexicons, and a number of surface features. In our findings, TueMix achieved better results than the neural network systems, placing in the top third of 62 teams (listed as "guzimanis" in the official CodaLab rankings). The addition of the linguistic features beyond the TF-IDF n-grams were also shown to enhance our model. In this paper, we detail our approach.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1316–1321
Barcelona, Spain (Online), December 12, 2020.

2 Related Work

A growing body of work has addressed some of the aforementioned challenges of sentiment analysis of code-mixed content. The SAIL Code-Mixed Shared Task (Patra et al., 2018) focused on the detection of emotion in Indian languages with code-mixed text. Most of the participants attempted to solve the task by using machine learning algorithms such as support vector machines (SVMs) or naive Bayes trained with n-grams or sentiment lexicon-based features. Another approach used in the same competition employed the use of emoji dictionaries annotated with the corresponding sentiment. The participants who submitted solutions based on deep learning systems did not report better results than those who submitted solutions based on machine learning.

Another approach to sentiment analysis of code-mixed text involved the construction of a system composed of a monolingual and a multilingual model (Vilares et al., 2015). High accuracy and F1-scores were reported when pursuing a hybrid approach, specifically a system that initially creates sub-word level representations for words employing a convolutional neural network (CNN), which are subsequently fed together with orthographic features and word embeddings into a dual encoder network (Lal et al., 2019). Similarly high results were reported in an approach consisting of two bidirectional long short-term memory networks (LSTMs) combined with orthographic features and monolingually trained word embeddings (Kumar and Dhar, 2018). Another attempt included SVM, naive Bayes, logistic regression, and neural networks trained with TF-IDF vectors and two monolingual dictionaries, one for English and one for Hindi (Kaur et al., 2017). An additional analysis of 840 experiments revealed the best combination for emotion detection of Hinglish text to be a combination of TF-IDF feature vectors, feature selection based on gain ratio, and a radial basis function neural network (Ravi and Ravi, 2016).

3 Model

In this section we present the architecture of our systems. In line with previous work on the effectiveness of using TF-IDF vectors for sentiment analysis, we first created a sparse feature matrix of word and character n-grams using sklearn's TF-IDF vectorizer (Pedregosa et al., 2011). Baseline measures were calculated for seven different sklearn classifiers, namely logistic regression, linear SVM, random forest, multinomial naive Bayes, Gaussian naive Bayes, k-nearest neighbors, and extreme gradient boosting. A preliminary analysis of the scores revealed logistic regression as the highest-performing model. As a point of comparison, we also developed neural network models in parallel using Keras (Chollet and others, 2015). The following additional steps were undertaken to improve the performance of our models.

3.1 Pre-Processing

The data set contained all elements of the tweets, including URLs, hashtags, user mentions, and the retweet marker 'RT'. Previous work has shown that cleaning tweets of potential noise elements can improve model performance (Shoukry and Rafea, 2012; Ifrim et al., 2014, e.g.). Through experimentation, we discovered that our model performed the best when only URLs were excluded. The other elements thus appeared to carry relevant information toward sentiment classification. Following this finding, we limited our pre-processing to the removal of URLs.

3.2 Neural Networks

For comparative purposes, we experimented with different neural networks. We started with the implementation of a simple LSTM model with one embedding and one LSTM layer which had 64 units and a recurrent dropout of 0.1. A bidirectional LSTM model was explored in comparison to the simple LSTM model. We also implemented a CNN which consisted of one embedding layer, a one-dimensional convolutional layer with 250 filters and a kernel size of three, and a global max pooling layer. The CNN also had two dropout layers and two dense layers with the dimensionality of the output space set to 250 for the first and three for the second.

Each of the models were compiled with a categorical cross-entropy loss function, and the optimizer was set to adam (Chollet and others, 2015). The models were trained on 30 epochs and with the batch size set to 128. After an analysis, we discovered that our neural network approach performed worse than the

baseline logistic regression classifier. Neural networks in general are difficult to train and tune, especially when building such complex systems. Since these models did not show the expected results, we shifted our attention back to the machine learning approach.

3.3 TueMix

As logistic regression performed the best in our preliminary analysis, we focused on improving its performance through tuning and feature selection. To avoid overfitting the data, we used 5-fold cross validation and tuned the hyperparameters using grid search. The best performing model after tuning had an optimal C value of 1, an L2 penalization norm, and the algorithm used in the optimization problem was Newton-CG. Since the class distributions were not entirely equal, we also set the class weights to balanced.

3.3.1 Feature Engineering

In addition to word and character n-grams vectors, we created several other features which could be relevant for sentiment analysis of code-mixed tweets. The following list describes each feature, and a summary of selected feature combinations used for feature selection is illustrated in Table 1.

1. **TF-IDF n-grams:**
    - **Word (W):** We experimented with various lengths of the n-grams and ultimately determined uni-grams, bi-grams, and tri-grams to be the optimal lengths.
    - **Character (CH):** Similar to word n-grams, different n-grams lengths between one and four were tested, and n-grams of length two to four ultimately performed the best in combination with other features.
2. **Monolingual Sentiment Lexicons (SL)**:
    - We obtained an English sentiment lexicon (Cambria et al., 2016) and a Hindi sentiment lexicon (Chen and Skiena, 2014). Both contained a list of words and multiword expressions classified as positive or negative, and the Hindi lexicon was written in Hindi script. Since the tweets contained both Hindi and transliterated characters, we transliterated the list using an online software tool.[1] Keeping the matches to the transliterated and untransliterated lists as separate features improved our performance. This thus created a set of six features: two for the number of matches to the positive and negative English lists and four features for the matches to the transliterated and untransliterated positive and negative Hindi lists. Applying min-max scaling to the six features additionally enhanced our results.
3. **Surface Features:**
    - **The predominant language in a tweet (PL):** Using the language labels, we determined the predominant language in a tweet and converted it to a categorical feature.
    - **The number of uppercase words in a tweet (UW):** The raw number of uppercase words in a tweet were calculated, and then the feature was scaled using min-max scaling. This was found to perform better than other methods of calculation, such as the percentage of uppercase words.
    - **The percentage of Hindi words (HW):** This feature represented the number of Hindi words in a tweet divided by the number of English words in a tweet.
    - **The percentage of universal tokens (UT):** This feature calculated the number of tokens marked as universal, such as symbols and punctuation, divided by the total number of tokens in a tweet.

As can be seen in Table 1, each additional feature, with the exceptions of the percentage of Hindi and the percentage of universal tokens, improved the model's performance when added to the word and character n-gram vectors. However, the number of universal tokens did enhance the model in combination

[1] http://www.easyhindityping.com/hindi-to-english-translation

Feature set	F1-Score
W	0.5745
CH	0.6047
W + CH	0.6147
W + CH + SL	0.6180
W + CH + PL	0.6160
W + CH + UW	0.6150
W + CH + HW	0.6133
W + CH + UT	0.6127
W + CH + SL + PL + UW	0.6230
W + CH + SL + PL + UW + HW	0.6221
W + CH + SL + PL + UW + UT	**0.6243**
W + CH + SL + PL + UW + HW + UT	0.6218

Table 1: Macro-averaged development set F1-Scores for testing of feature combinations

with other features, while the addition of the percentage of Hindi did not. This may have been due to the percentage of Hindi feature conveying similar information to the categorical predominant language feature. The combination of features generating the highest score is highlighted in bold and represents the combination subsequently used for TueMix.

4 Evaluation

4.1 Model Performance

The performance of TueMix was subsequently evaluated on the test data. For our training data, we concatenated the development set to the original training data, giving us a total of 17000 tweets. As comparisons, we also ran the test data on the CNN and a logistic regression model with the same hyperparameters using only word and character n-grams. As demonstrated in Table 2, TueMix outperformed the CNN and performed marginally better than a logistic regression model trained on only word and character n-grams. Our weighted F1-score of 0.685 placed 17th out of 62 teams.

Model	Accuracy	Precision	Recall	F1-Score
CNN	0.534	0.601	0.534	0.553
LR w/ W + CH only	0.686	0.683	0.686	0.683
TueMix	0.687	0.684	0.687	**0.685**

Table 2: Results on test data

We further evaluated the F1-scores for each class. We achieved an F1-Score of 0.738 for the prediction of the positive class and 0.736 for the prediction of the negative class. This shows that TueMix was much better at predicting these classes compared to the neutral class, for which we achieved an F1-Score of 0.594. To break this down further, we constructed a confusion matrix. As illustrated in Figure 1, for the positive and negative classes, most of the false predictions were for the adjacent neutral class. For the neutral class, the false predictions were fairly evenly split between the positive and negative class.

4.2 Model Coefficients

Since our additional linguistic features improved TueMix's performance, we wanted to gain a better understanding of how they contributed to the model. We ran each surface feature separately as a single feature in a logistic regression model with the same hyperparameters to examine the strength and direction of the coefficients. As shown in Table 3, the strongest single feature was the percentage of Hindi in the tweet, despite the categorical predominant language feature performing better in combination with the other features. It is also interesting to observe that the percentage of Hindi was positively associated

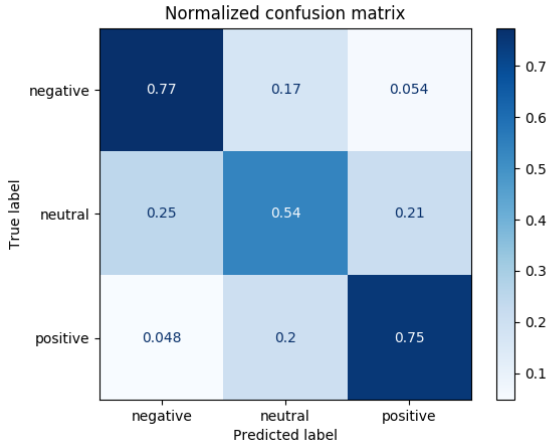

Figure 1: Normalized confusion matrix for test data

Feature	Coefficients			F1-Score
	Positive	Neutral	Negative	
PL	-0.56	0.03	0.52	0.325
UW	0.87	-0.13	-0.74	0.289
HW	-1.39	-0.39	1.78	0.387
UT	0.72	1.09	-1.81	0.303

Table 3: Coefficients and F1-Scores of single language features using logistic regression

with the negative class, moderately yet negatively associated with the neutral class, and more strongly negatively associated with the positive class. In other words, the more Hindi was in the tweet compared to English, the more likely the tweet was to be negative.

In addition, the number of uppercase words and the percentage of universal tokens were positively correlated with the positive class and negatively associated with the negative class. This suggests that in our data, uppercase case words and tokens such as symbols and punctuation were more likely to express enthusiasm or positivity rather than anger or negativity.

4.3 Error Analysis

While inspecting the data, we noticed some errors and inconsistencies in the dataset annotation, specifically in the word-level language tags. Let us consider one example from the dataset, *@ (O) marrjao (Hin) @ (O) Shining (Eng)__ (O) star77 (Eng) Ohho (Eng) Very (Hin) beautiful (Eng) eyes (Hin) yarrrrr (Hin)*, where the tokens *Very* and *eyes* are mislabelled as Hindi. These errors influenced our calculations of the surface features, namely the predominant language in a tweet and the percentage of Hindi words. We also noticed that some of the emojis were labelled as universal instead of emojis. This impacted the accuracy of the percentage of universal tokens surface feature. Numerous characters and words from languages other than English or Hindi were also observed.

5 Conclusion & Future Work

In our approach to this task, we compared a neural network approach with a logistic regression classifier, TueMix, consisting of three categories of linguistic features. The linguistic features beyond TF-IDF n-grams were shown to enhance the model's performance and resulted in a final F1-Score of 0.685. TueMix was also much more effective at predicting the positive and negative classes than the neutral class. Further work could seek to improve predictions for the neutral class and investigate which additional linguistic features impact sentiment analysis and in which direction. Our error analysis also revealed some of the difficulties of collecting and analyzing code-mixed data.

Our results are in line with other tasks which have shown n-gram-based machine learning models to outperform neural networks (Çöltekin and Rama, 2018a; Çöltekin and Rama, 2018b). However, because we devoted more attention to tuning and feature engineering TueMix, we cannot make an assertion that a machine learning approach is strictly superior to a neural network approach for this task. Overall, due to the aforementioned challenges, much work remains to be done in the field of sentiment analysis of code-mixed social media text. Several avenues of future research include combining rule-based systems with machine learning systems, adding differing weights to different features and parts of the text, and the generation of more multilingual or code-mixed lexical resources for sentiment analysis.

References

Kalika Bali, Jatin Sharma, Monojit Choudhury, and Yogarshi Vyas. 2014. "I am borrowing ya mixing ?" an analysis of English-Hindi code mixing in Facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 116–126, Doha, Qatar, October. Association for Computational Linguistics.

Erik Cambria, Soujanya Poria, Rajiv Bajpai, and Bjoern Schuller. 2016. SenticNet 4: A semantic resource for sentiment analysis based on conceptual primitives. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2666–2677, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Yanqing Chen and Steven Skiena. 2014. Building sentiment lexicons for all major languages. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 383–389, Baltimore, Maryland, June. Association for Computational Linguistics.

François Chollet et al. 2015. Keras. `https://keras.io`.

Çağrı Çöltekin and Taraka Rama. 2018a. Tübingen-oslo at SemEval-2018 task 2: SVMs perform better than RNNs in emoji prediction. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 34–38, New Orleans, Louisiana, June. Association for Computational Linguistics.

Çağrı Çöltekin and Taraka Rama. 2018b. Tübingen-Oslo system: Linear regression works the best at predicting current and future psychological health from childhood essays in the CLPsych 2018 shared task. *ArXiv e-prints*.

Janet Holmes. 2008. An introduction to sociolinguistics. England.

Georgiana Ifrim, Bichen Shi, and Igor Brigadir. 2014. Event detection in Twitter using aggressive filtering and hierarchical tweet clustering. *CEUR Workshop Proceedings*, 1150:33–40, 01.

Harpreet Kaur, Veenu Mangat, and Nidhi Krail. 2017. Dictionary based sentiment analysis of Hinglish text. *International Journal of Advanced Research in Computer Science*, 8(5).

Vaibhav Kumar and Mrinal Dhar. 2018. Looking Beyond the Obvious: Code-Mixed Sentiment Analysis (CMSA).

Yash Kumar Lal, Vaibhav Kumar, Mrinal Dhar, Manish Shrivastava, and Philipp Koehn. 2019. De-mixing sentiment from code-mixed text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 371–377, Florence, Italy, July. Association for Computational Linguistics.

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. *CoRR*, abs/1803.06745.

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.

Kumar Ravi and Vadlamani Ravi. 2016. In *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*, pages 641–645. IEEE.

Pamela Charbel Shbat. 2007. *The relationship between gender and types of code-switching among Lebanese youth-by Pamela Charbel Shbat*. Ph.D. thesis.

Amira Shoukry and Ahmed Rafea. 2012. Preprocessing Egyptian dialect tweets for sentiment mining. In *The Fourth Workshop on Computational Approaches to Arabic Script-based Languages*, page 47.

Shikaripur Sridhar and Kamal Sridhar. 1980. The syntax of psycholinguistics of bilingual code-mixing. *Canadian Journal of Psychology/Revue canadienne de psychologie*, 34:407–416, 12.

David Vilares, Miguel Alonso Pardo, and Carlos Gómez-Rodríguez. 2015. Sentiment analysis on monolingual, multilingual and code-switching Twitter corpora. pages 2–8, 01.