Proceedings of the WILDRE-6 Workshop @LREC2020, pages 35–41
Marseille, 20 June 2022

Identifying Emotions in Code Mixed Hindi—English Tweets

Sanket Sonu, Rejwanul Haque, Mohammed Hasanuzzaman, Paul Stynes, and Pramod Pathak
School of Computing
National College of Ireland
Mayor Street Lower, IFSC, Dublin 1, Ireland
x19206071@student.ncirl.ie, firstname.lastname@ncirl.ie

Abstract

Emotion detection (ED) in tweets is a text classification problem that is of interest to Natural Language Processing (NLP) researchers. Code-mixing (CM) is a process of mixing linguistic units such as words of two different languages. The CM languages are characteristically different from the languages whose linguistic units are used for mixing. Whilst NLP has been shown to be successful for low-resource languages, it becomes challenging to perform NLP tasks on CM languages. As for ED, it has been rarely investigated on CM languages such as Hindi—English due to the lack of training data that is required for today's data-driven classification algorithms. This research proposes a gold standard dataset for detecting emotions in CM Hindi–English tweets. This paper also presents our results about the investigation of the usefulness of our gold-standard dataset while testing a number of state-of-the-art classification algorithms. We found that the ED classifier built using SVM provided us the highest accuracy (75.17%) on the hold-out test set. This research would benefit the NLP community in detecting emotions from social media platforms in multilingual societies.

Keywords: Emotion Detection, BERT, Code-mixing

1. Introduction

People nowadays use social media more often; even they follow news channels on micro-blogging websites such as Twitter and Facebook. In fact, many users prefer to use such platforms and avoid traditional media platforms (e.g. television) for news. India is a country with a one and a half billion population. A significant portion of the population makes use of micro-blogs on daily basis for a variety of reasons, e.g. entertainment, learning, and motivation. Short comments posted by social media users on the micro-blogging websites may show certain kinds of emotions. This can be used by multinational companies and SMEs to check the opinions of the users for a variety of reasons, e.g. reviewing products (Onan, 2021), identifying buying intentions (Haque et al., 2019), recognising complaints about products or services (Preoţiuc-Pietro et al., 2019; Singh et al., 2020a; Singh et al., 2020b). Such hidden insights could be crucial for improving the quality of their products or services. Over and above, identification of emotions or sentiments could be used in improving the quality of output of other NLP models, e.g. machine translation (MT) (Kumari et al., 2021), text summarization (Abdi et al., 2018). In fact, ED is not a new area of NLP and being heavily investigated over a decade and on a variety of contents, e.g. tweets (Liew and Turtle, 2016), technology review (Garcia-Garcia et al., 2017), suicide notes (Desmet and Hoste, 2013). The bidirectional encoders using masked language models, e.g. bidirectional encoder representations from Transformers (BERT) (Devlin et al., 2018), multilingual BERT (mBERT) (Pires et al., 2019), produce state-of-the-art results in numerous NLP tasks. This transfer learning strategy is very effective when labelled data is not abundantly available especially in low-resource scenarios. To the best of our knowledge, none of the existing large-scale language models have been trained exclusively with code-mixed data. Therefore, the problem is that they do not perform well in the NLP tasks involving CM languages. Despite this problem, we considered a popular CM language for investigating emotion analysis, Hindi–English. Hindi is the most popular language in India, and it is spoken as a first language by nearly half a billion people worldwide and as a second language by some 120 million more. A large number of people use Hindi with English for speaking and writing. In social networking platforms, this has even become a normal trend that users write Hindi posts in scripts that are a mixture of Hindi and English linguistic units (e.g. words, syllables). Such code-mixed posts are hard to parse for a variety of reasons, e.g. the NLP preprocessors were traditionally trained on monolingual datasets, not on CM datasets. Moreover, the expression of interest of a social media user may be associated with the user's state of mind, emotion, or other phenomena. Hence, the different users can express their thoughts of interest in numerous ways. For an example, a Hindi word "tmhara" which means 'your' in English can be written in a variety of forms, e.g. "tmharaa", "tumhara", "tumara", "tmhare". To the best of our knowledge, there are no freely-available linguistic preprocessors and tools (syntactic or semantic) that can handle CM Hindi–English texts, e.g. word sense disambiguation, lemmatization, parsing, spelling corrections. This is another challenge that researchers face while they encounter CM languages for different NLP tasks.

In this work, we focused on creating a gold standard dataset for detecting emotions in CM Hindi–English

social media texts. To the best of our knowledge, the paper that is closely related to ours is (Vijay et al., 2018), who investigated emotion detection from CM Hindi–English social media texts. More specifically, (Vijay et al., 2018) extracted tweets using Twitter's API and created a small data set that constitutes of 2,866 instances which were annotated into six emotions (Ekman, 1992): '*Happy*', '*Sad*', '*Angry*', '*Fear*', '*Disgust*', or '*Surprise*'. For building their ED classifiers they used only three emotion classes. They considered those classes that are more prevalent in the dataset, i.e., '*Happy*', '*Sad*', '*Angry*'. Although the work of Vijay et al. (2018) is closely related to ours, our work differs from them in many ways and the key contributions of this work are: (i) we manually created a gold-standard dataset for detecting emotions in CM Hindi–English social media texts. Unlike (Vijay et al., 2018), this is nearly a class-balanced data set, and (ii) we achieved an F_1 score of 75.17% on our held-out test set with our best-performing ED classifier that was built using the SVM algorithm, which surpassed the performance of the best-performing models of Vijay et al. (2018) and Wadhawan and Aggarwal (2021).

2. Related Work

As discussed above, Vijay et al. (2018) investigated ED from CM Hindi–English social media texts. They created a small data set that has a class-imbalance issue. In order to avoid the class-imbalance problem in classification, they considered instances with the three most frequent classes for system building. Vijay et al. (2018) applied a number of preprocessing strategies including replacement of URLs, user names, emoticons with placeholders such as 'URL', 'USER', 'Emoticon', respectively. For building classifiers they extracted features using character *n*-grams, *n*-grams, emoticons, punctuation, repeating characters, negations, sentiment lexicons, upper case characters, and intensifiers. They were able to achieve a maximum of 58.2% accuracy when SVM is used. They demonstrated the performance of their classifiers by integrating the aforementioned features into their classification models. Their experiments showed that features derived from URLs, user names, emoticons, negations, lexicons do not help. More recently, Wadhawan and Aggarwal (2021) cited the problem of using the data created by Vijay et al. (2018) for building data-driven classifiers in detecting emotions in CM user-generated content. Instead, they created a self-annotated class-balanced Hindi–English CM dataset using Twitter tags like #happy, #sad, #angry, #fear, #disgust, #wow. In other words, they automatically annotated tweets using a list of collected searched #tags. For example, all tweets under the *#happy* tag were labelled as *'Happy'* emotion. They used different recurrent neural networks (RNNs) for building their classifiers, in particular with Bi-directional Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997). Additionally, they made use of three pre-trained LMs for classification, BERT, RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019). They were able to achieve an accuracy of 71% on their test set when they used BERT.

We now turn our attention to the papers that looked into sentiment analysis in the social media sphere. The works of Tiwari and Sinha (2020) and Srividya and Sowjanya (2019) focused on sentiment detection in English Facebook posts. They collected Facebook posts and annotated the collected datasets into three sentiments, positive, negative, or neutral. As for training, they mainly considered the supervised learning algorithms that are commonly used for text data such as SVM and Naïve Bayes. They were able to achieve an accuracy in the range of 90% on their test data. There have been a plethora of works that studied this area of NLP considering both high-resource (e.g. English) or low-resource languages (e.g. Indic languages); we refer interested readers to some of the notable papers (Ahmad et al., 2019; Mukherjee, 2019; Singh, 2021). We also refer an excellent initiative to promote this line of NLP research through introducing a shared task (Sentiment Analysis of Code-Mixed Tweets) (Patwa et al., 2020).

3. Corpus Creation and Annotation

The data set created by Vijay et al. (2018) contains a list of 2,866 instances, each of which is manually tagged into one of the following six sentiments *'Happy', 'Sad', 'Angry', 'Fear', 'Disgust', or 'Surprise'*. They made their data set available online for the NLP community. We followed their annotation scheme for creating a robust and balanced dataset. First, we collected tweets using Twitter's official API – Tweepy.[1] Since our aim is to collect CM Hindi–English tweets, we set the language parameter of API to 'Hindi'. We used a list of #tags in order to collect tweets that are expected to be CM Hindi–English texts; some of them are *happy, smile, birthday, badhai ho, Sad, darr, mujhe darr lag rha, Surprise, amazing*. This crawling process provided us with a raw dataset that contains 102,809 CM Hindi–English tweets. In this regard, as pointed out in Section 2, Wadhawan and Aggarwal (2021) automatically created a CM Hindi–English data set using the strategy that we just described. However, the process of automatically annotating tweets by making use of #tags has an important drawback. Many tweets that were extracted with #tag *'happy'* may not be opinionated texts. In fact, some of them could belong to different emotion classes. As an example, consider a tweet "wow amazing. I can't even imagine, how can he be so happy?". This tweet was labelled as *'Happy'* since it was extracted using #tag. In reality, the tweet should belong to the *'Surprise'* emotion class. Since we wanted to create a balanced dataset for emotion detection in user-generated content and it would serve as a gold-standard dataset for this task, the dataset must

[1] https://www.tweepy.org/

not contain any noisy or anomalous examples. That is the reason why we carried out a systematic manual annotation task which is described below.

The tweets that we crawled contains a significant amount of unnecessary entries. For example, they are too short text, contain only URLs, links, #tags, images, Hindi script or English scripts. Such tweets were automatically discarded from the initial list of tweets. Then, we applied the following cleaning procedure in order to clean the tweets: (i) removing URL: links, email IDs, and URLs that were part of tweets were removed, (ii) removing #tags and @User_Names: #tags and @User_Names were removed. (iii) removing emoticons: all emoticons that were part of tweets were removed.[2] (iv) removing RT: all RT tags that were part of tweets were removed, and (v) removing noises: all the null values and special characters were removed from tweets.

As pointed out above, posts in micro-blogging websites usually are short and a word can take one of the numerous lexical variations of the word (e.g. "tmhara" as "tmharaa", "tumhara", "tumara", "tmhare"; cf. Section 1). This can easily be captured by continuous distributed vector representation of words using the Skip-gram model (also known as Word2Vec) (Mikolov et al., 2013) or similar algorithms. However, for this, we need to have a word-embedding model trained on CM Hindi-English corpus. To the best of our knowledge, there is no such freely available corpus to be used for this purpose. Alternatively, we created a list of the most frequently used (Hindi or CM Hindi–English) words on Twitter, which are written in multiple ways. We manually created a list of 365 words, and for each word all alternative variations were turned into their most widely used variation. We will see in the next section that this strategy helped in improving the performance of our ED classification models. We illustrate this with an example. Consider the following Hindi tweet "bro tm pagaal hoo", its literal English translation is "bro you are crazy". The Hindi word 'tum' may be written as 'tumm', 'tm', 'tuum'. In our case, this Hindi tweet is turned into "bro tum pagal ho". Additionally, if there is a tweet that has spelling mistakes, spelling errors are corrected. As an example, consider a Hindi tweet 'bro tm pagaalhoo gaeho" whose literal English translation is "You have become crazy". This tweet is modified or corrected as "bro tum pagal ho gaye ho". This is accomplished via manual inspection since such spelling mistakes cannot be detected using regular expressions. In sum, we applied a noise cleaning method to the tweets. This cleaning process was carried out with a manual editor who has excellent English and Hindi skills and good knowledge of tweets. On completion of preprocessing and manual cleaning, we ended with a set of 7,150 tweets.

[2]In future, we aim to test emoji normalisation method (`https://pypi.org/project/emoji/`) too as they could interesting signal for detecting emotions.

3.1. Annotation

We label each of the collected clean tweets with either one of the 6 emotion categories (*'Happy', 'Sad', 'Anger', 'Fear', 'Disgust', 'Surprise'*) or the no emotion category. The manual annotation process is accomplished with a GUI that randomly displays a tweet from the set of 7,150 tweets. At the end of the annotation task, each tweet is associated with one tag. Since we have three annotators and three values are associated with each of the tweets of the set of 7,150 labeled tweets. The final class for a tweet is determined based on a majority voting strategy. There might be complete disagreement (i.e. each annotator tag it with different class) among the annotators. In that case, we do not consider that Tweet. On completion of the annotation task, we ended up with a set of 6,299 annotated Tweets. The inter-annotator agreement was computed using Fleiss' Kappa (Fleiss and Cohen, 1973) at tweet level. For each tweet, we count an agreement whenever three annotators agree with the annotation result. We found the kappa score to be high (i.e. 0.83) for the annotation task. This indicates that our tweet labeling task is to be excellent in quality.

In Table 1, we show two examples of CM Hindi–English tweets from our dataset. The first and second examples shown in Table 1 belong to *'Happy'* and *'Sad'* classes, respectively.

Table 1: CM Tweets from our gold-standard dataset.

CM Tweet	main bht khush hn Modi k decision se
Translation	I am happy with Modi's decision
CM Tweet	ipl kab se chalu hoga bhaiya so sad
Translation	When will IPL start brother so sad

4. Data Statistics

The CM Hindi-English dataset of Vijay et al. (2018) contains 2,866 tweets which are labeled with 6 emotions. From now on, we call this dataset Dataset 1. As mentioned in the above section, our CM Hindi–English dataset contains 6,299 tweets which were labeled with 7 emotion tags. From now on, this dataset is referred to as Dataset 2. Table 2 shows the data distribution, i.e., count of tweets in each emotion class. To build our classifiers, we used a combined dataset (i.e. our dataset with one created by Vijay et al. (2018)) that contains a total of 9,165 CM Hindi–English tweets. From now on, we call the combined dataset Final Dataset. As can be seen from Table 2, although numbers for "No emotion" and "Fear" are slightly high (1,892) and low (559), respectively, Final Dataset is nearly a class-balanced data set. In future, we aim to enrich this data set by increasing the number of Tweets in each class (e.g. "Fear").

5. Emotion Detection: Methodology, Results, and Discussion

In above, we discussed how we created a gold-standard dataset for detecting emotions in CM Hindi–English

Labels	Dataset 1	Dataset 2	FD
No emotion	0	1,892	1,892
Happy	595	631	1,226
Sad	878	651	1,529
Angry	667	1,096	1,763
Fear	85	474	559
Disgust	291	856	1,147
Surprise	182	867	1,049

Table 2: Data Distribution

social media texts. We tested the usefulness of this dataset by employing a number of classification algorithms on it. This section details the building of our ED classification models and presents evaluation results.

5.1. Experimental Setups

The recurrent neural network, in particular with LSTM hidden units, has been proved to be an effective model for many classification tasks in NLP. In addition to LSTM, we used SVM, Logistic Regression, Random Forest algorithms for building our classifiers. As discussed in Section 2, Wadhawan and Aggarwal (2021) fine-tuned BERT on an automatically created CM Hindi–English dataset and achieved an accuracy of 71% on their test set. We also tested how BERT would perform on our data set. In order to obtain an optimal set of hyperparameters for SVM, Logistic Regression, and Random Forest, we used a simple grid search algorithm[3] based on a 5-fold cross-validation strategy. For building our classifiers we tried with a range of hyperparameters and a number of their combinations for training our neural-network-based classifiers, LSTM and BERT. We list the optimal set of hyperparameters for each of the classification algorithms: (i) *SVM*: kernel = ['rbf ','linear','poly','sigmoid'], gamma = [1, 0.1, 0.01], and C = [0.1, 1, 10, 100], (ii) *Logistic Regression*: C = [0.1, 1, 20, 40, 60, 80, 100], solver = ['lbfgs', 'liblinear'], (iii) *Random Forest*: n_estimators = [80,85,90,95,100], max_depth = [20,30,None], criterion = ['gini','entropy'], (iv) *LSTM*: epochs = 2, batch size = 64, embedding dim = 100, SpatialDropout1D = 0.4, Memory, unit = 250, LSTM layer dropout = 0.2 and recurrent dropout = 0, Dense layer activation = 'softmax', loss='categorical crossentropy', and optimizer = 'adam', and validation split = 0.1 and (v) *BERT*: epochs = 3, dropout = 0.2, Dense layer activation ='softmax', loss='categorical crossentropy', and optimizer = 'adam'.

We applied four different types of preprocessing modules to our dataset: (i) *punctuation*: This preprocessing module removes all the punctuation's from the dataset, (ii) *stop-words*: We used a list of 1,036 Hindi–English stop-words, (Rana, accessed on 1st Oct 2021). These stop-words are removed from the corpus, (iii) *numbers*: This module removes all the numeric entities from the corpus, and (iv) *repeating characters*: the repeating characters are removed from the corpus (e.g. *'happyyyyy' is changed to 'happy'*).

Each of our classification models were tested on the following setups: (a) setup 1: Removing Punctuation: All the models are trained and tested after removing punctuations from the corpus, (b) setup 2: Removing Stop words: All the models are trained and tested after removing stop words from the corpus, (c) setup 3: Removing Numbers: All the models are trained and tested after removing numbers from the corpus, (d) setup 4: Removing Repeating Characters: All the models are trained and tested after removing repeating characters, (e) setup 5: setups 1–4 All the models are trained and tested after removing punctuation, stop-words, numbers, and repeating words, and (f) setup 6: Keeping all above features: All the models are trained and tested without eliminating any of the above special features.

5.2. Feature Extraction

We followed the standard methods to convert texts (tweets) into numerical vectors so that it is understandable by machine learning models. For this, we applied a popular bag-of-words strategy with two different weightings: word frequency counts (WFC), term-frequency inverse-document-frequency (TF-IDF). We found that classifiers trained on TF-IDF weighted data outperformed the one that was trained on WFC weighted. Therefore, we used the TF-IDF weighting in our training setup. As for measuring TF-IDF weights, we considered both 1-gram and 2-gram word occurrences. Additionally, we also experimented with converting text data into dense vectors using Doc2Vec.[4] We tested three different algorithms available in Doc2Vec: distributed bag-of-words (DBOW), distributed memory (DM), concatenation of DBOW and DM (DBOW+DM).

5.3. Results

For evaluation we split our data set into two parts, i.e. 80% instances used for training and 20% instances were used for evaluation. In order to measure classifier's performance on the test set, we used a widely-used evaluation metric: accuracy.

Table 3 presents performance of our classifiers for TF-IDF features (1- and 2-grams). As can be seen from Table 3, SVM with TF-IDF unigrams weighting provided us an accuracy of 75.17% when none of the special features were eliminated from the dataset (Setup 6). This turned out to be our best-performing ED classifier.

We picked the best-performing model (SVM; cf. Setup 6 of Table 3), and show confusion matrix and classification performance in terms of precision, recall and $F1$ in Tables 4 and 5. We can see from Tables 4 and 5 that our ED SVM model performs consistently across the

[3] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[4] https://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html

	Models for TF-IDF (1-gram)			
Setups	**SVM**	**LR**	**RF**	**LSTM**
1	75.12%	73.37%	73.32%	71.03%
2	72.83%	72.77%	74.24%	68.47%
3	75.12%	73.32%	72.94%	71.96%
4	75.06%	73.37%	73.75%	68.14%
5	72.88%	73.15%	74.74%	69.78%
6	75.17%	73.43%	73.48%	71.79%
	Models for TF-IDF (2-gram)			
Setups	**SVM**	**LR**	**RF**	**LSTM**
1	74.03%	74.30%	71.24%	71.36%
2	74.03%	74.41%	73.64%	69.34%
3	74.03%	74.79%	72.34%	73.05%
4	73.97%	74.57%	71.35%	68.19%
5	73.64%	74.41%	72.83%	69.18%
6	73.97%	74.30%	72.23%	72.23%

Table 3: Performance of our ED classifiers (accuracy).

all emotion classes and produces excellent precision, recall, and F_1 on the evaluation test set.

	Confusion Matrix						
Labels	**0**	**1**	**2**	**3**	**4**	**5**	**6**
NE - 0	323	2	5	19	3	1	8
Happy - 1	43	171	13	14	0	1	0
Sad - 2	53	11	219	24	0	1	2
Angry - 3	58	5	13	259	0	7	4
Fear - 4	14	3	6	6	70	3	2
Disgust - 5	35	3	4	21	1	175	0
Surprise - 6	45	4	4	14	2	1	161

Table 4: Confusion Matrix of SVM with TF-IDF(*1*-gram) weighting without removing any of the special features (Setup 6). NE: No Emotions.

Labels	**Precision**	**Recall**	**F1-score**	**Support**
0	0.57	0.89	0.69	361
1	0.86	0.71	0.78	242
2	0.83	0.71	0.76	310
3	0.73	0.75	0.74	346
4	0.92	0.67	0.78	104
5	0.93	0.73	0.82	239
6	0.91	0.70	0.79	231
accuracy			0.75	1833
macro avg	0.82	0.74	0.76	1833
weighted avg	0.79	0.75	0.76	1833

Table 5: Performance of SVM with TF-IDF(*1*-gram) weighting without removing any of the special features (Setup 6).

In Table 6, we present the performance of our classification models when features were extracted using the "Doc2Vec" word embedding method. As can be seen from Table 6, none of the features (DBOW, DM, DBOW+DM) is effective, the classification models performed quite poorly as they produce accuracy in the range of 28–45%. As in (Wadhawan and Aggarwal, 2021), we fine-tuned BERT on our data set in order to see how it would perform on a small-sized CM data. We found that BERT's accuracy is quite low (nearly 20%) on our held-out test set.

	Models for Doc2Vec – DBOW		
Setups	**SVM**	**LR**	**RF**
1	29.84%	28.69%	27.82%
2	36.33%	35.35%	35.51%
3	29.18%	28.75%	28.91%
4	29.89%	28.53%	28.96%
5	34.47%	34.58 %	34.58%
6	29.51%	28.09%	28.80%
	Models for Doc2Vec – DM		
Setups	**SVM**	**LR**	**RF**
1	33%	34.04%	25.25%
2	32.46%	31.96%	27.71%
3	33.06%	32.73%	23.94%
4	33.82%	34.64%	24.22%
5	31.96%	33.06 %	29.51%
6	32.18%	32.67%	24.05%
	Models for Doc2Vec – (DBOW+DM)		
Setups	**SVM**	**LR**	**RF**
1	37.97%	38.62%	30.44%
2	43.97%	44.62%	40.48%
3	35.62%	37.20%	28.15%
4	38.35%	38.62%	29.78%
5	41.89%	42.22 %	38.78%
6	38.89%	39.77%	31.42%

Table 6: Performance of classification models building using word-embeddings generated by Doc2Vec.

6. Conclusions and Future Work

In this paper, we presented our experiments on emotion detection in code-mixed Hindi–English social media text. Since there was no publicly available balanced data set for this task, we created a moderate-sized gold-standard balanced dataset. For this, we crawled CM Hindi–English tweets from the most popular micro-blogging website, Twitter. Our data preparation process included a number of steps including cleaning, transformation, and annotation. Next, we investigated usefulness of our dataset and used a number of state-of-the-art classification algorithms for building emotion detection classifiers. We carried out a wide range of experiments in order to find the setup that would work best for this dataset. We obtained an accuracy of 75.17% on our hold-out test set with our best-performing ED classifier that was built using the SVM algorithm. Our best-performing ED classifier outperformed the best-performing models presented in Wadhawan and Aggarwal (2021) and Vijay et al. (2018) with large margins.

In future, we aim to test multi-stage fine-tuning strategy on BERT (Xie et al., 2020) for this task, where in the first stage we will fine-tune BERT on automatically extracted self-annotated data as in Wadhawan and Aggarwal (2021), and in the second stage we will fine-tune the model obtained in first stage on our data set.

7. Bibliographical References

Abdi, A., Shamsuddin, S. M., Hasan, S., and Piran, J. (2018). Machine learning-based multi-documents sentiment-oriented summarization using linguistic treatment. *Expert Systems with Applications*, 109:66–85.

Ahmad, G. I., Singla, J., and Nikita, N. (2019). Review on sentiment analysis of indian languages with a special focus on code mixed indian languages. In *2019 International Conference on Automation, Computational and Technology Management (ICACTM)*, pages 352–356.

Desmet, B. and Hoste, V. (2013). Emotion detection in suicide notes. *Expert Systems with Applications*, 40(16):6351–6358.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ekman, P. (1992). Are there basic emotions?

Fleiss, J. L. and Cohen, J. (1973). The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 33(3):613–619.

Garcia-Garcia, J. M., Penichet, V. M., and Lozano, M. D. (2017). Emotion detection: a technology review. In *Proceedings of the XVIII international conference on human computer interaction*, pages 1–8.

Haque, R., Ramadurai, A., Hasanuzzaman, M., and Way, A. (2019). Mining purchase intent in twitter. In *Proceedings of CICLing 2019, the 20th International Conference on Computational Linguistics and Intelligent Text Processing*, La Rochelle, France.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Kumari, D., Ekbal, A., Haque, R., Bhattacharyya, P., and Way, A. (2021). Reinforced nmt for sentiment and content preservation in low-resource scenario. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 20(4):1–27.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Liew, J. S. Y. and Turtle, H. R. (2016). Exploring fine-grained emotion detection in tweets. In *Proceedings of the NAACL Student Research Workshop*, pages 73–80.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546.

Mukherjee, S. (2019). Deep learning technique for sentiment analysis of hindi-english code-mixed text using late fusion of character and word features. In *2019 IEEE 16th India Council International Conference (INDICON)*, pages 1–4.

Onan, A. (2021). Sentiment analysis on product reviews based on weighted word embeddings and deep neural networks. *Concurrency and Computation: Practice and Experience*, 33(23):e5909.

Patwa, P., Aguilar, G., Kar, S., Pandey, S., PYKL, S., Gambäck, B., Chakraborty, T., Solorio, T., and Das, A. (2020). Semeval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. *CoRR*, abs/2008.04277.

Pires, T., Schlinger, E., and Garrette, D. (2019). How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*.

Preoţiuc-Pietro, D., Gaman, M., and Aletras, N. (2019). Automatically identifying complaints in social media. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5008–5019, Florence, Italy.

Rana, S. (accessed on 1st Oct 2021). Hinglishnlp. `https://github.com/TrigonaMinima/HinglishNLP/blob/master/data/assets/stop_hinglish`.

Singh, R. P., Haque, R., Hasanuzzaman, M., and Way, A. (2020a). Identifying complaints from product reviews: A case study on hindi. In *Proceedings of the Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2020)*, Dublin, Ireland.

Singh, R. P., Haque, R., Hasanuzzaman, M., and Way, A. (2020b). Identifying complaints from product reviews in low-resource scenarios via neural machine translation. In *Proceedings of ICON 2020: 17th International Conference on Natural Language Processing*, Patna, India.

Singh, D. (2021). Detection of emotions in hindi-english code mixed text data. *CoRR*, abs/2105.09226.

Srividya, K. and Sowjanya, A. M. (2019). Sentiment analysis of face book statuses.

Tiwari, S. and Sinha, A. (2020). Sentiment analysis of facebook data using machine learning. *International Journal of Innovative Research in Applied Sciences and Engineering*, 4:2456–8910, 10.

Vijay, D., Bohra, A., Singh, V., Akhtar, S. S., and Shrivastava, M. (2018). Corpus creation and emotion prediction for hindi-english code-mixed social media text. pages 128–135, 01.

Wadhawan, A. and Aggarwal, A. (2021). Towards emotion recognition in hindi-english code-mixed data: A transformer based approach.

Xie, Y., Yang, W., Tan, L., Xiong, K., Yuan, N. J., Huai, B., Li, M., and Lin, J. (2020). Distant supervision for multi-stage fine-tuning in retrieval-based ques-

tion answering. In *Proceedings of The Web Conference 2020*, pages 2934–2940.