BAUSTJ
BAUST Journal

Comparative Analysis of Emoji Prediction from Tweets using Machine Learning Techniques

Shamira Shamsee[1], Md. Al-Hasan[2*], Serajus Saleheen[3], Mst. Rubina Aktar[4], Md. Mahbubur Rahman Tusher[5], and Susmita Roy Rinky[6]

[1,2*,3,5,6]Department of CSE, Bangladesh Army University of Science and Technology, Saidpur, Bangladesh
[4]Department of EEE, Bangladesh Army University of Science and Technology, Saidpur, Bangladesh

emails: *[1]shamira170201032@gmail.com; [*2]al-hasan@baust.edu.bd; [3]serajussaleheen74@gmail.com; [4]rubina@baust.edu.bd; [5]mdmahbuburrahmantusher@gmail.com; and [6]susmita.roy.rinky@gmail.com ('*' - corresponding author)*

ARTICLE INFO

Article History:

Received: 07 August 2022
Revised: 28 October 2022
Accepted: 12 November 2022
Published online: 31 December 2022

Keywords:

Emoji prediction
Recurrent neural network
Natural language processing
Bi-directional memory
Logistic regression

ABSTRACT

Emojis are little images that are frequently utilized by social media in text messages. In this thesis, we investigate how words and emojis interact by simulating the task of predicting the emojis from textual tweets. In the previous year's many people try to predict emoji from tweet using any specific one or two algorithms for training their model. The performance of several classifiers, machine learning techniques, and deep learning on the Twitter dataset is highlighted in this study. According to experiments, bi-directional LSTM is the most reliable sentiment predictor LSTM is the most reliable sentiment predictor, according to experiments, with a 90% accuracy rate. Additionally, we have included 27 new emojis.

© 2022 BAUSTJ. All rights reserved.

1. INTRODUCTION

With the advent of social media, a new method of communication emerged in which the meaning is made up of a combination of brief texts and visual additions, or emojis. [Fig 1]. Not only when using Twitter, but also when utilizing other significant online social media platforms like Facebook, WhatsApp, or Instagram, this, the visual portion of the message, is now the new norm. [1]. Emojis are widely utilized in social media as a language for the graphic depiction of emotions. They transmit additional sentimental information beyond textual content and improve the effectiveness of interaction mediated by an electronic device. [3]. Recent advancements in AI provide a chance to use this to automatically write text messages with context-relevant emojis. [6]. Emoji evaluation and prediction are extremely difficult due to the complexity and sensitivity of emoji usage. The main goal of this thesis is to determine whether a trained machine learning model, deep learning model, or classifier can predict emojis, could aid in a better understanding of natural language and thus, to multiple tasks involving natural language processing, along with providing emoji-enriched content on social media, enhancing emotional/sentiment analysis systems, and improving the extraction of the materials from social networks. [5]. By exploring the difficulty of predicting emoji that provide more meaning to a given input of text and have a strong relationship to the meaning transmitted by language, we hope to reduce the gap between emojis and textual matter. The models we explore here,

Multinomial NB Classifier [5],
Bernoulli NB Classifier [6],
Logistic Regression Classifier [7],
SVC (Support Vector Classifier) [8],
Linear SVC [9], [10].
Decision Tree Classifier [11],
Random Forest Classifier [12].
Long Short-Term Memory (LSTM) [13],
Gradient Boosting [14],
Bi-Directional Long Short-Term Memory (Bi-LSTM) [15].

This thesis has been implemented using NLTK [12] and Python 3 [11]. It uses the Python 3.7.6 [8] version. Python doesn't require compilation, thus testing and troubleshooting proceed swiftly. A sizable number of open-source libraries are available for Python. Python's Natural Language Toolkit

(NLTK) library facilitates program development and data classification.

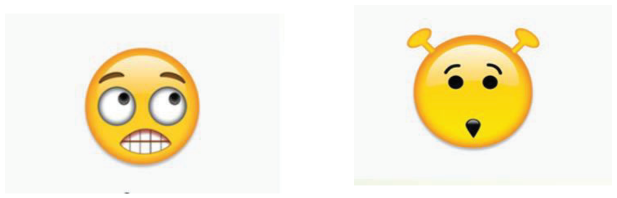


Figure 1: Popular emojis

We train a sizable dataset of words labelled with emoticons taken from tweets. In the Inference stage, the trained classifier considers the best emoji to employ in the sentence after receiving the sentence as input. The main goal is to develop a model that can predict an emoji that is appropriate for the given language as accurately as possible. And to help individuals convey their feelings more effectively, we've created 26 new emojis. Section 2 provides a summary of the related research that has been done in this field. The evolution of the system and its architecture are discussed in Section 3. The system's outcome is presented in Section 4. In Section 5, this essay is finished.

2. LITERATURE REVIEW

Text-based symbols like:-) and:-(could be used to replace language, according to computer scientist Scott Fahlman, who initially proposed this idea in 1982, predating the emoji [19]. Throughout the 2000s, new editions of Unicode expanded its list of written characters several times. However, there was little interest in including the Japanese cellular emoji sets (which were deemed outside of scope) [20], even though symbol characters that would later be categorized as emoji were constantly being added. For instance, the Unicode 4.0 release introduced 16 new emoji, such as eject buttons, caution triangles, and direction arrows [21]. Nicolas Loufrani, the CEO of The Smiley Company, led the campaign for digital smileys [22]. He developed a smiley toolbar that was used in the early 2000s at smileydictionary.com and could be emailed just like emoji are now [23].
The Smiley Dictionary was created by The Smiley Company and released in 2001 [24]. A proposal to include the ARIB extended characters used in Japanese broadcasting in Unicode was made in 2008. There were various pictograms included in this. In 2009, Unicode 5.2 included them. The eggplant (aubergine) emoji (U+1F346), which is almost exclusively [better source needed] used in North America to depict a penis, has also caused controversy. From December 2014 forward. A sentiment analysis of emoji was released in December 2015, and a 2016 Emoji pedia investigation found that just 7% of English language tweets using the peach emoji refer to the actual fruit, indicating that the peach emoji (U+1F351) has also been employed as a euphemism emblem for buttocks [25]. A musical about emojis debuted in Los Angeles in 2016 [26] [27]. In January 2017, in what is believed to be the first large-scale study of emoji usage, researchers at the University of Michigan analyzed over 1.2 billion messages input via the Kika Emoji Keyboard [28] and announced that the Face with Tears of Joy was the most popular emoji.
A drop of blood emoji was launched on March 5, 201, [29] with the goal of eradicating the stigma associated with menstruation [30]. In addition to normalizing times, it will also be important to highlight healing events like blood donation and other blood-related exercises.

3. DATASET

We got Tools and Datasets from the competition of SemEval-2018 Task 2-Detection [31]. The Twitter Emoji dataset included 50,000 tweets with the corresponding emoji designation. A matching integer label that identified a particular emoji was assigned to each tweet in the collection. The 20 most popular emojis are included in this list. Initially, the dataset is written to a CSV file in raw text format. First, we open an excel file and create a new spreadsheet. Then we select the first row of the first column and wrote the tweet. After then we selected the second row of the first column and then select data. We select from the text and go to the folder where the text file has existed. Selecting that text file and then clicking on open. Then selecting delimitated and next. Then select tab and click on next, then click on general, then finish. Then click on existing worksheet and click ok. In this way, we are bringing all the data of the first text file. Then select the first row of the second column and write the label. Then selecting second of row second column. In this way inserting all the other labels from the label file in the same procedure. Then go to save as and select file named other formats then click on csv and saved it.

4. METHODOLOGY

For machine learning and classifier, first we collect raw data. Then we clean and compress the data. Then we split the data into train and test. Then we train our models using processed dataset and find out confusion matrix and accuracy. In Figure 2 depicted the overall methodology which consists of pre-processing, modelling, training which are described below.

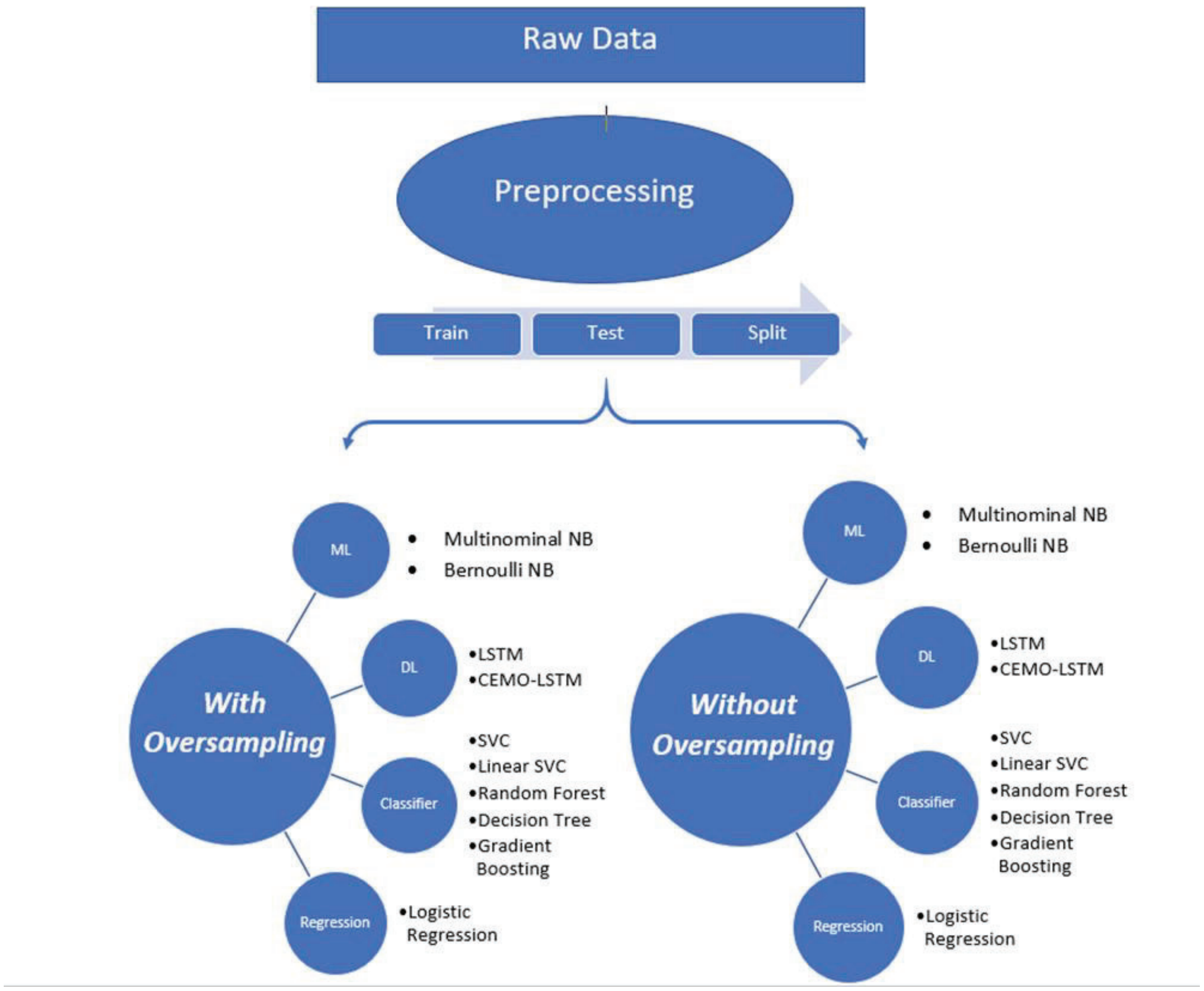


Figure 2: Overall Methodology

A. *Pre-processing*

Pre-processing for ML and Classifier:
We are describing here the approach we use for ML and classifier.

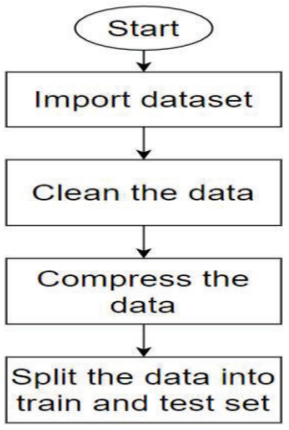


Figure 3: Pre-processing for Classifier

i. *Cleaning the Data*

We used a function called tweet_clean (tweet) which we remove@ from the sentence. RT, Hyperlinks, Numbers, Punctuation.

ii. *Spacy*

We used the free open-source software spacy for natural language processing (NLP) to expand the tokenization patterns and enhance the linguistic data already available.

iii. *NLTK*

We used NLTK stem package to remove morphological affixes from words, leaving only the word stem and to import wordNetLemmatizer. With the help of this sizable, openly accessible lexical database for the English language, we can lemmatize words and create organized semantic linkages between them.

iv. en_core_web_sm

We used en_core_web_sm as English pipeline for optimized the CPU and for trained the written tweet and as a token to vector.

v. Train Test Split

We used the train-test split to take dataset and dividing it into two subsets. We used 85% of data for training our model and 15% of data to testing the model with random state 101.

vi. Pre-processing for Deep Learning:

We are describing here the approach what we used for deep learning.

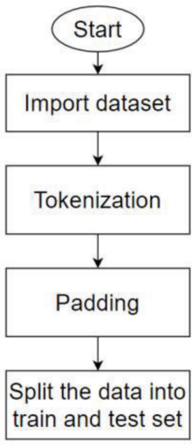


Figure 4: Pre-processing for Deep Learning

- **Tokenization**: Tokenization is essentially used to break up a longer text passage into smaller lines and words.
- **Padding:** We use padding to identify the end of a sentence because the decoder is trained sentence-by-sentence.
- **Train Test Split:** We use train-test split procedure for taking a dataset and dividing it into two subsets. We used 70% of data for training and 30% of data for testing our model.

B. Models

In this section, we described about the methods what we used.

i. Multinomial Naïve Bias

We imported naive_bayes module from sklearn package. Then import Multinomial NB model from naive_bayes module. We only use the fit technique to fit the model to the training set of data without taking any parameters.

ii. Bernoulli Naïve Bias

We imported naive_bayes module from sklearn package. Then import Bernoulli NB model from naïve_bayes module. We take one parameter which is alpha=2 which define there will be feature vectors are binary (i.e., zeros and ones) and fit the model according to the training data using fit method.

iii. SVC

We imported svm module from sklearn package. Then import SVC model from svm module. We don't take any parameter just fit the model according to the training data using fit method. We import svm module from sklearn package. Then import linear SVC model from svm module. We don't take any parameter just fit the model according to the training data using the fit method. We take this model for large dataset.

iv. Random Forest Classifier

We imported ensemble module from sklearn package. Then import Random Forest Classifier model from ensemble module. We take one parameter which is the random _state for initializing the inside arbitrary number generator, which can choose to part the information into prepare and test that set the random _state a settled esteem which means each time same grouping of irregular numbers is produced each time I run the code. Then just fit the model according to the training data using fit method.

v. Decision tree Classifier

We imported tree module from sklearn package. Then import DecisionTreeClassifier model from tree module. We don't take any parameter just fit the model according to the training data using fit method.

vii. Gradient Boosting Classifier

We imported ensemble module from sklearn package. Then import GradientBoostingClassifier model from ensemble module. We take five parameters. The learning rate hyper parameter limits the rate or speed at which the model learns. It could be a hyper parameter that controls how much to alter the model in reaction to the evaluated mistake each time the demonstrate weights are upgraded. The number of boosting steps is indicated by the value n estimators, which we set to 0.5. Gradient boosting is moderately resistant to over-fitting; therefore, a large number often results in much improved performance. We set it 20. Max depth is generally integer. Maximum depth of the individual regression estimators. Adjust this parameter for optimal performance; the ideal value depends on how the input components interact.

We set it 2. In case of max_features for the best split we choose integer and consider max_features feature at two splits. The look for a part does not halt until at slightest one substantial segment of the hub tests is found, indeed if it requires to successfully review more than max_ features highlights. In our setting random state to a fixed value would ensure that the same set of random numbers is created every time I execute the code. We chose random state = 0 for initializing the internal irregular number generator, which may choose the portion of information into train and test records. Simply use the fit technique to fit the model to the training set of data.

C. Models for Deep Learning

i. LSTM

TensorFlow is a Python library. Keras is an API. sequence is the bland term for a requested set. We import sequential, Dropout, Dense, LSTM, Bidirectional, Embedding, GlobalMaxPool1D from TensorFlow, kerasmodels Use vocabulary_size = vocab_size, seq_len=40 and embed_len=20. Use SoftMax activation function. Use early stopping for better training the model. Here Batch_size=128 and epochs=20 for training.

ii. Bi-Directional LSTM

TensorFlow is a Python library. Keras is an API sequence is the generic term for an ordered set. We import sequential, Dropout, Dense, LSTM, Bidirectional, Embedding, GlobalMaxPool1D from TensorFlow. kerasmodels. Use vocabulary_size = vocab_size, seq_len = 40 and embed_len=128. Use SoftMax and relufunction. Use early stopping for better training the model. Here Batch_size=64 and epochs=15 for training.

iii. Logistic Regression Classifier

We import linear_model module from sklearn package. Then import logistic regression classifier model from linear_model module. We use parameter c, max_iter and n_jobs. I choose (inverse of regularization) = 2 to overcome over fitting problem, use max_iter=1000 to avoid converge problem and use n_jobs=-1 defines jobs so that all processors will be used in parallel. To fit the model to the provided training data, we employ the fit approach. We employ the predict () function, which accepts the x test parameter as the test to be performed. It gives back the labels for the training data the model provided. As a result, the predict () method utilizes the trained model in addition to mapping the learnt label and predicting the labels for the test data.

iv. Target Emoji

Since bi-directional lstm meets the necessary requirements, we employ it. Bidirectional LSTMs are an evolution of conventional LSTMs that can advance display execution on problem categorization for arrangement. Bidirectional LSTMs prepare two LSTMs on the input grouping rather than one in problems where all time steps of the input arrangement are accessible.

5. RESULTs AND DISCUSSION

Table 1

Emoji prediction accuracy with oversampling

Algorithm	Accuracy
Multinomial Naïve Bias	0.27
LSTM	0.27
Logistic Regression	0.28
SVC	0.30
Bernoulli Naïve Bias	0.21
Gradient Boosting	0.21
Bi-Directional LSTM	0.27
Linear SVC	0.25
Random Forest	0.25
Decision Tree	0.21

Table 2

Emoji prediction accuracy without over sampling

Algorithm	Accuracy
Bernoulli Naïve Bias	0.21
LSTM	0.88
Logistic Regression	0.22
SVC	0.30
Multinomial Naïve bias	0.20
Gradient Boosting	0.05
Bi-Directional LSTM	0.90
Linear SVC	0.19
Random Forest	0.28
Decision Tree	0.20

In the above two tables, we have got some different accuracies using different kinds of algorithm. At first, we got the accuracies with oversampling after then again; we implemented those algorithms without oversampling. We can see some algorithms are slide difference from each other in both scenarios. Some are huge differ to others. But we can LSTM and Bi-Directional LSTM two algorithms are given much better accuracy from among all. Using oversampling we got 88% of accuracy for LSTM and 90% of Bi-Directional LSTM. on the other hand, without oversampling, we can see far differ as before. Only 27% 0f accuracy for both LSTM and Bi-Directional LSTM. Now we can mention the other accuracies Like BNB and SVC both have same accuracies as 21% and 30% in two scenarios. And LR 22%-28%, MNB

20%-27%, GBC 5%-21%, Linear SVC 19%-25%, RFC 28%-25% and DTC 20%-21% of accuracies in with oversampling and without oversampling.

Real World data are not organized and the recurrent neural network dataset converts the real-world data into it. Random Inspecting is the foremost common pattern that's utilized for comparing against dynamic learning methodologies.

The RND approach randomly selects events from the unlabelled pool without considering whether those events provide the classifier with any additional information. Although it is straightforward and does not consider any classifier-specific value, it implicitly chooses representative instances that are evenly and independently distributed, thus it frequently acts as a solid baseline that is challenging to overcome. All significant traits may not be captured in the sample by simple random sampling. Regularly, we stratify the test (isolate the test into groups depending on certain characteristics) and then choose test units from those groups or strata if the features alluded to by the distinct colors are anything considered to be significant while organizing the study. Usually carried out to ensure that all features are properly reflected.

A. Multinomial naïve Bias

Multinomial Naïve Bias cares about counts for multiple features that do occur. So, after random sampling when the count is difference, the accuracy getting low.

B. Bernoulli Naïve Bias

Bernoulli Naive Bias is use to predict the class variable took up only the values yes or no. So, after using random sampling there is no effect on accuracy.

C. Support Vector Classifier

Support Vector Classifier cannot fit the data as well. It probably could not find any ways to predict the result other than choosing majority classes. So, after random sampling there is no changes in SVC.

D. Linear SVC

Linear SVC is use liblinear. Therefore, it has additional options for selecting penalty and loss functions. So, after random sampling the accuracy become low.

E. Random forest Classifier

A meta estimator that fits a variety of classification trees on several subsamples of the dataset and uses averaging to improve predictive accuracy and reduce overfitting is an example of an irregular timberland classifier. That is why accuracy improves following random sampling.

F. Decision Tree

Decision Tree are greedy and deterministic if they don't normally give their best result if we add one row more or take one out the result can be different, also that they tend to over fit. That is why it give low accuracy after random sampling.

G. Gradient Boosting Classifier

Gradient Boosting Classifier is a weak learner that classifies the data but doesn't poorly, perhaps no better that random guessing. So, after random sampling the accuracy decreases.

H. Logistic Regression

All significant traits may not be captured in the sample by simple random sampling. The sample will typically be divided into groups based on these characteristics (a process known as stratifying the test), and then test units from those groups or strata, if the characteristics represented by the different colors are something that was thought to be significant when designing the study. This is frequently done to make sure that everyone is truly addressed.

I. Long Short-Term Memory

LSTM are broadly utilized for arrangement expectation issues and have demonstrated to be greatly compelling. The reason they work so well is that LSTM can store past vital data and disregard the data that's not. So, the accuracy increases after random sampling.

J. Bi-Directional LSTM

Bi-directional lstm uses for multiple lstm. Here the input flows in two directions. The key feature is that gated recurrent neural networks can store information that can be store for future call. It entails duplicating the primary repeated layer in the arrangement such that there are now two layers next to one another, supplying the input grouping as input to the primary layer on the off chance that, and delivering a reversed copy of the input grouping to the moment. Using all the input data in the past and future of a certain time frame, the bi-directional recurrent neural network may be trained. A regular RNN's state neurons should be divided into portions for the forward state (positive time course) and the reverse (negative time course) (in reverse state). This is because we get high accuracy after random sampling.

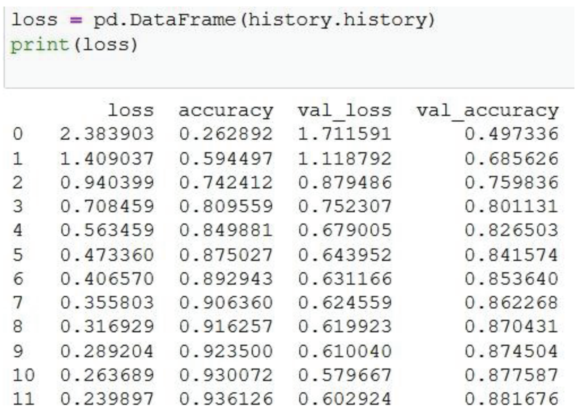
6. ERROR ANALYSIS

In this section, we have discussed the issue of error.

A. LSTM

After oversampling the two losses (both loss and vaue loss) are decrease and the two accuracies (accuracy and value_accuracy) are increasing. So, this shows the modeling is prepared in a great way. So oversampling is good for LSTM.

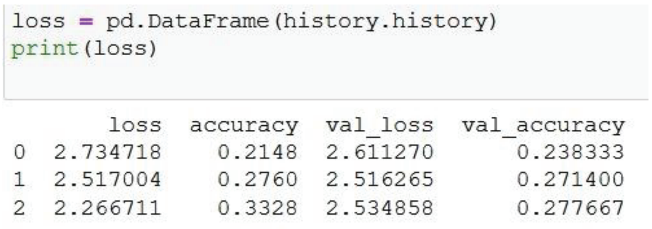
i. With Oversampling



```
loss = pd.DataFrame(history.history)
print(loss)

        loss  accuracy  val_loss  val_accuracy
0   2.383903  0.262892  1.711591      0.497336
1   1.409037  0.594497  1.118792      0.685626
2   0.940399  0.742412  0.879486      0.759836
3   0.708459  0.809559  0.752307      0.801131
4   0.563459  0.849881  0.679005      0.826503
5   0.473360  0.875027  0.643952      0.841574
6   0.406570  0.892943  0.631166      0.853640
7   0.355803  0.906360  0.624559      0.862268
8   0.316929  0.916257  0.619923      0.870431
9   0.289204  0.923500  0.610040      0.874504
10  0.263689  0.930072  0.579667      0.877587
11  0.239897  0.936126  0.602924      0.881676
```

Figure 5: Loss and accuracy with oversampling

ii. Without Oversampling



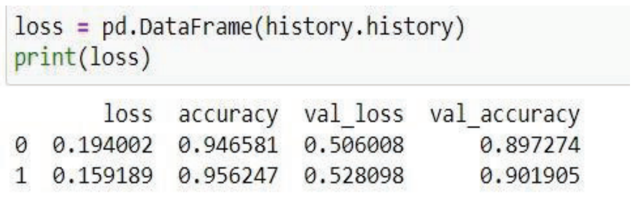
```
loss = pd.DataFrame(history.history)
print(loss)

       loss  accuracy  val_loss  val_accuracy
0  2.734718    0.2148  2.611270      0.238333
1  2.517004    0.2760  2.516265      0.271400
2  2.266711    0.3328  2.534858      0.277667
```

Figure 6: Loss and accuracy without oversampling

B. Bi-Directional LSTM

After oversampling loss is decreasing but val_loss is increasing, so we use early stopping. And both accuracy and val_accuracy is increasing. So, the model training is good. So, for bi-directional LSTM oversampling is good if we use early stopping.

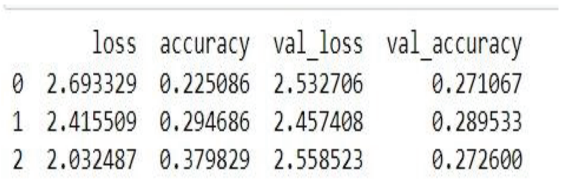
i. With Oversampling



```
loss = pd.DataFrame(history.history)
print(loss)

       loss  accuracy  val_loss  val_accuracy
0  0.194002  0.946581  0.506008      0.897274
1  0.159189  0.956247  0.528098      0.901905
```

Figure 7: Loss and accuracy with oversampling

ii. Without Oversampling



```
       loss  accuracy  val_loss  val_accuracy
0  2.693329  0.225086  2.532706      0.271067
1  2.415509  0.294686  2.457408      0.289533
2  2.032487  0.379829  2.558523      0.272600
```

Figure 8: Loss and accuracy without oversampling

7. PROPOSED MODEL AND INVENTED EMOJIS

Our proposed model is Bi-Directional LSTM with 90% accuracy.

We have created the below emojis. These emojis could be used to express different situations.

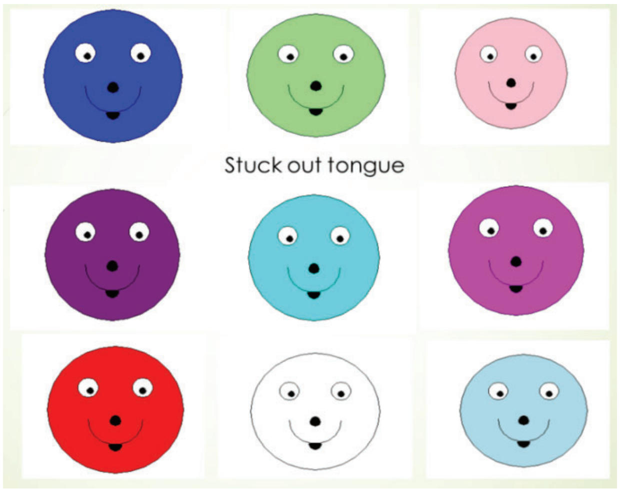


Figure 9: Stuck out tongue emojis express in different situations

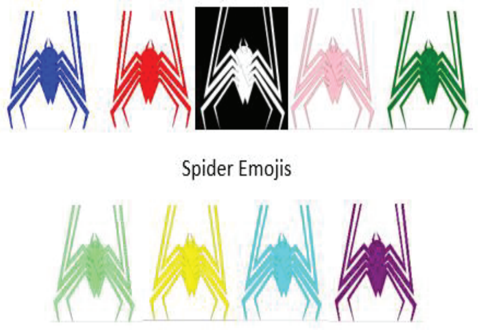


Figure 10: Spider Emojis express in different situations

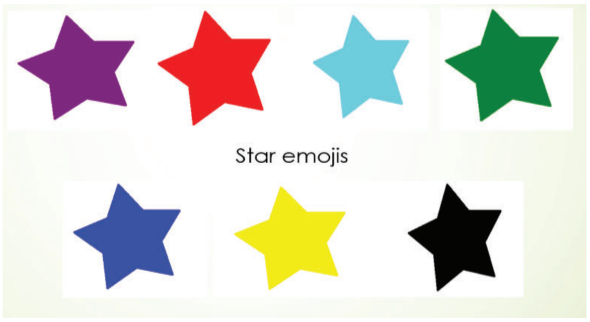


Figure 11: Star Emojis express in different situations

In future people who do this emoji prediction thesis can understand that which algorithm is best. And can create many emoji like us.

8. CONCLUSION AND FUTURE WORK

In this paper we trained multiple machine learning classifier, deep learning, and regression model. And our proposed model is bi-directional LSTM model which give higher accuracy. And we have also created 27 emojis to express different situtations. In future, we will develop our algorithm and we also work on our proposed emojis.

REFERENCES

[1] Felbo, Bjarke, et al. "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing,* 2017.

[2] Zhao, Peijun, et al. "Analyzing and predicting emoji usages in social media." *Companion Proceedings of the The Web Conference 2018*, 2018.

[3] Barbieri, Francesco, Miguel Ballesteros, and Horacio Saggion. "Are emojis predictable?." *Computation and Language (cs.CL),* 2017.

[4] Cappallo, S., Svetlichnaya, S., Garrigues, P., Mensink, T. and Snoek, C.G., 2018. New modality: Emoji challenges in prediction, anticipation, and retrieval. *IEEE Transactions on Multimedia*, *21*(2), pp.402-415.

[5] C.D. Manning, P. Raghavan and H. Schütze, "Introduction to Information Retrieval", *Cambridge University Press*, pp. 234-265, 2008.

[6] A. McCallum and K. Nigam, "A comparison of event models forNaive Bayes text classification", *Proc. AAAI/ICML-98 Workshop on Learning for Text Categorization*, pp. 41-48, 1998.

[7] Schmidt, Mark, Nicolas Le Roux, and Francis Bach. "Minimizing finite sums with the stochastic average gradient." *Mathematical Programming* 162 2017: 83-112.

[8] Cristianini, Nello, and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[9] T. Wu, C. Lin and R. Weng, "Probality estimates for multi-class classification by pairwise coupling", *Proc. JMLR-5*, pp. 975-1005, 2004.

[10] http://scikitlearn.org/stable/modules/svm.html#svm-classification. Accessed 30 Mar 2023

[11] Lam, Savio LY, and Dik Lun Lee. "Feature reduction for neural network based text categorization." In *Proceedings. 6th international conference on advanced systems for advanced applications*, pp. 195-202. 1999.

[12] Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.

[13] Sundermeyer, Martin, Ralf Schlüter, and Hermann Ney. "LSTM neural networks for language modeling." *Thirteenth annual conference of the international speech communication association*. 2012.

[14] Liu, Man. "Emonlp at semeval-2018 task 2: English emoji prediction with gradient boosting regression tree method and bidirectional lstm." In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pp. 390-394. 2018.

[15] Jang B, Kim M, Harerimana G, Kang SU, Kim JW. Bi-LSTM model to increase accuracy in text classification: Combining Word2vec CNN and attention mechanism. Applied Sciences. 2020 Aug 24.

[16] Welcome to Python.org.
In: Python.org. https://www.python.org./. Accessed 30 Mar 2022

[17] NLTK. https://www.nltk.org/. Accessed 30 Mar 2022.

[18] "Happy 30th Birthday Emoticon!"
Independent. September 8, 2012. Retrieved November 30, 2017.

[19] turns 25, Associated Press, September 20, 2007, archived from the original on October 12, 2007, retrieved September 20, 2007.

[20] Whistler, Ken (February 1, 2021). "Re: Origins of ⌚ U+231A WATCH and ⌛ U+231B HOURGLASS". Unicode Mail List Archives.

[21] Jump up to: a b "Emoji Encoding Principles". Unicode Consortium.

[22] "Host of New Characters and Emoji Introduced in Unicode 7.0". Hexus. June 17, 2014. Retrieved Nov 30, 2017.

[23] "Nokia 3310 User guide" (PDF). Nokia.

[24] Scherer, Markus; Davis, Mark; Momoi, Kat; Tong, Darick; Kida, Yasuo; Edberg, Peter. "Emoji Symbols: Background Data—Background data for Proposal for Encoding Emoji Symbols", UTC L2/10-132.

[25] Kircher, Madison (December 16, 2016). "Very Official Study Finds Peach Emoji Most Often Paired with Eggplant". Emojipedia. Retrieved July 7, 2018.

[26] "Emoji Sentiment Ranking". Retrieved Dec 8, 2015.

[27] Jump up to: a b Gans, Andrew (April 12, 2016). "New Musical About Emojis Will Premiere in Los Angeles". Playbill. Retrieved December 23, 2016.

[28] Lawrence, Derek (July 27, 2017). "The Emoji Movie: Here's what the critics are saying". Entertainment Weekly. Retrieved August 13, 2017.

[29] "Snapchat Emoji Meanings". Archived from the original on August 15, 2018. Retrieved February 28, 2017.

[30] "Emoji Recently Added, v12.0". www.unicode.org. Retrieved November 16, 2020.

[31] https://competitions.codalab.org/competitions/17344. Accessed 30 Mar 2022.