

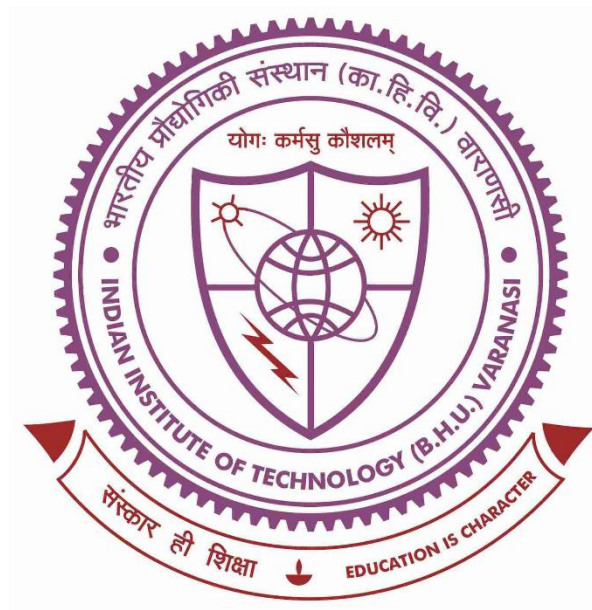
# **Emoji Prediction for Tweets using Word2Vec Representations**

By

**Mayank Dwivedi**  
**15SCSE108001**  
**Galgotias University, Greater Noida**

Under the Guidance of

**Dr. A. K. Singh**



**Department of Computer Science & Engineering**  
**INDIAN INSTITUTE OF TECHNOLOGY (BANARAS HINDU UNIVERSITY)**  
**VARANASI – 221005**

# CONTENTS


1. Abstract
2. Overview
3. Dataset and Pre-processing
4. Models and Algorithms
  - 4.1 Word2Vec Model
  - 4.2 KNN Classifier
  - 4.3 Logistic Regression Classifier
  - 4.4 SVM Classifier
5. Development
6. Analysis and Evaluation
7. Conclusion
8. Future Work
9. Acknowledgement
10. References

## ABSTRACT

Emojis are pictograms and smileys used for complementing a message visually in lesser words. They are becoming a way of expressing emotions better in textual interactions. Social media platforms like Twitter has made the use of emojis even more frequent due to their limitations on the number of characters. Prediction of emojis in sentences is thus bound to become a more and more consistent part of Natural Language Processing with their growing popularity. Inclusion of emojis in newer Unicode encoding versions is only a further confirmation to this fact. In this project, my prime objective was to try and develop an emoji prediction and classification model for text-based tweet messages. This model predicts a single emoji from a limited set of 20 emojis based on the words frequently used in the tweet. The tweets were represented as vector of words using the Word2Vec model and then classified into respective emoji labels using different classifier algorithms.

## OVERVIEW

The evolution of textual interactions over web and the need to improve expressions in digital conversations has led to the usage of emojis. This evolution resulted in the emojis becoming a part of encoding standards so that they can be used over any platform universally. From 2010 onwards, some emoji character sets have been incorporated into Unicode, a standard system for indexing characters, which has allowed them to be used and to be standardized across different operating systems.

Further promotion of emoji usage occurred with the advent of social media platforms like Facebook, Twitter, WhatsApp etc. The trend of keeping messages short along with making them more visually appealing and expressive increased due to these platforms. Microblogging sites like Twitter introduced character limitations, focusing on sharing short messages. The 'Face with Tears of Joy' () emoji was even selected as the Oxford Dictionaries Word of the Year in 2015.

Due to their increasing popularity, processing and studying emojis in Natural Language Processing and Machine Learning is becoming important. With Twitter being an ideal source of diverse topics and high frequency of emoji usage, it becomes an ideal candidate for such studies and model development. In this project, I have tried to study the relationship between the textual contents of tweet message and the emoji used in them and also predict the emojis for tweets based on their representation.











I have envisioned this model as a classification problem (with multiple classes) of machine learning domain. During this task I developed a classification model which generates an emoji as output that goes most probably/frequently with a given twitter message. The machine relies primarily upon the Word2Vec model to generate word vectors which are then used as features for the classifier models (KNN Classifier, SVM, Logistic Regression Classifier etc.) to predict the most probable emoji for the given input.

## **DATASET AND PRE-PROCESSING**











**Dataset:** The dataset of the project consisted of 500,000 tweet messages which were extracted from Twitter using a REST framework crawler utilizing Twitter APIs. The dataset was based on pre-determined Tweet IDs which were already geo-localized to the region of United States of America. Moreover, the tweets contained one and only one emoji in the message which was a subset of a pre-defined frequent emoji set. Out of the 500,000 tweets, of them were not retrieved. This could be due various reasons such as the tweet being deleted, the content not being available as a public tweet etc. This reduced my final dataset to 486,807 tweets.

For the sake of simplification, I used a limited set of 20 emojis to predict from. These emojis were the ones that were used most frequently in the tweet messages. Each emoji was mapped to a corresponding label number for classification. The emojis used and their corresponding labels are provided in Table 1.

**Pre-processing:** The input and labels were constructed by removing the emoji from the raw tweet and using it as the label for the same. Before feeding the input to the machine for further processing, I performed a few pre-processing steps in order to increase the efficiency and reduce noise and sparsity in data. The steps included removal of usernames (@user) and hashtags (#hashtag). The tweets were also stripped of URLs, special characters and retweet symbols. All the text was then converted to lowercase.

									
0	1	2	3	4	5	6	7	8	9

									
10	11	12	13	14	15	16	17	18	19

*Table 1: Emojis and their respective label numbers used in the experiment.*

## MODELS & ALGORITHMS

### 1. Word2vec Model

Word2Vec is set of various related models which are used to generate word embeddings for a large corpus of text. It produces a dense vector representation of each unique word in the corpus consisting of several dimensions of fixed length. Each word is represented by a point in the embedding space and these points are learned and moved around based on the words that surround the target word [2]. The word embeddings output by Word2vec model provide a better substitute over other word representation models such as latent semantic analysis. This model primarily incorporates either of the two frameworks to produce vector representations: The continuous bag-of-words (CBOW) model or skip-grams. For this project, I will be using the *Word2Vec* class provided in *Gensim*, an open-source Python library for Natural Language Processing with a focus on topic modelling. This class incorporates neural

network iterations to reconstruct and train linguistic context of words to the machine. The rare words below the word count of 5 were ignored in order to reduce noise. I kept the vector dimension size limited to 100 to minimize space requirements and training time.

## **2. KNN Classifier**

The K-Nearest Neighbours algorithm is a basic classification and regression algorithm in Machine Learning which is based on instance based learning. It outputs a class label determined by a majority vote of its  $k$  nearest neighbours on a graphical feature space. I used the KNN Classifier as my primary classifier with the word embeddings of 100 dimensions as feature space. The nearest neighbours taken in consideration were 5. This was due to the fact that even similar words at times could be used with different emojis which would have resulted in the decision boundaries being inaccurate. Although the KNN Classifier is an algorithm which is robust to noisy training data and its effective when the training data is large, the computation cost and time complexity of this algorithm gives it a slight disadvantage. Yet, the algorithm proves to be of decent accuracy in the provided classification task.

## **3. Logistic Regression Classifier**

Logistic Regression is a type of classification algorithm where the dependent variable is categorical. This algorithm applies a logistic function to a linear combination of features to predict the outcome of a categorical dependent variable based on predictor variables. I used Multinomial Logistic Regression classifier for this task due to the problem being a multi-class problem. The logistics regression classifier is easier to inspect and less complex. It is particularly beneficial in this task as it does not assume a linear relationship between the dependent and independent variables and hence can also handle non-linear effects. But since we are using a high dimensional vector space as features for the classifier, the logistic regression model may cause overfitting of the training data. It is probable that because of this, the resulting accuracy was not very good. Also, the logistic regression classifier requires a large training data to get more accurate results which might be the issue in this case.

#### 4. SVM Classifier

The Support Vector Machine (SVM) are supervised learning models. They are a versatile classification model which can perform linear classification as well as non-linear classification using kernel trick. Kernel trick involves mapping inputs implicitly into high- dimensional feature spaces. The kernel used by me in the task was *Radial Basis Function (RBF)* Kernel. SVM works by classifying the data into different classes by finding a line (hyperplane) which separates the training data set into classes. As there a multitude of such linear hyperplanes, the SVM algorithm tries to maximize the distance between the various classes that are involved. This is also referred as margin maximization. If the line that maximizes the distance between the classes is identified, the probability to generalize well to unseen data is increased. The SVM classifier goes well with the vector feature space I used and produces the highest accuracy results among the 3 classifier models I used. However, the time required to train the SVC is also high due to the complexity involved.

## DEVELOPMENT

In order to study the relation between words and emojis, I performed the modelling in two phases. In the first phase, I trained my Word2Vec neural network model against the pre-processed tweet dataset. The second phase involved training and prediction of different classifier models based on the sentence vector embeddings fed into them as feature space. The different models trained were then compared to the originals for the evaluation using the metrics accuracy score.

**Phase 1:** The tweets were pre-processed and then tokenized using the NLTK tokenizer library. NLTK provides a tokenizer class namely *TweetTokenizer* which is a Twitter-aware casual tokenizer, designed to be flexible and easy to adapt to new domains and tasks. The tweets

were hence converted into a list of tokens. This list of tokenized tweets was then used to train the Word2Vec model. Rare words below the word count of five were removed to reduce noise. Additionally, the model was pre-trained on a similar list of 50k trial tweets in order to increase the Twitter semantic vocabulary and accuracy.

The word embeddings from the model were then used to create a single vector representation for each tweet message in the dataset. Formally, each tweet message  $m$  was represented by vector  $V_t$  :

$$V_t = \sum_{t \in T_m} S_t$$

Where  $T_m$  are the set of tokens included in the message  $m$ ,  $S_t$  is the vector of token  $t$  in the Word2vec model, and  $|T_m|$  is the number of tokens in  $m$ .

After obtaining a representation of each tweet, they were fed into various classifier models for classification against given labels.




**Phase 2:** In the beginning of second phase we assigned the sentence representations list to the variable  $X$  and the label list for the corresponding tweets were assigned to variable  $y$ . I divided the dataset into 2 parts, training (90%) and testing (10%). The training dataset was used to train the three classifiers. The 100 dimensions of the sentence representations were used as features for the classifier models. Table 2 reports the results for all three algorithms on the complete dataset.

## ANALYSIS & EVALUATION

The accuracy score for all models was calculated using the *metric* module of *scikit-learn* toolkit. The metric accuracy score provided a decent measure of accuracy since the data does not require sophisticated accuracy and precision evaluation.

As we can observe from table 2, the SVM classifier outperforms both KNN and the Logistic Regression Classifier. However, KNN and Logistic Regression are quite competitive in their results.



	<b>Total Correct Predictions</b>	<b>Most Frequently Predicted Emoji</b>	<b>Accuracy Score</b>
<b>KNN Classifier</b>	182,961		<b>.37583</b>
<b>Log. Reg. Classifier</b>	165,826		<b>.34064</b>
<b>SVC</b>	347,930		<b>0.71472</b>

*Table 2:* Results of the KNN Classifier, Logistic Regression Classifier and SVC along with the most frequent emoji predicted by them.

## CONCLUSION

We can see that emojis are used extensively in social media, however little has been discovered about their use and semantics, especially because emojis are used differently and with varying contexts over different communities. Along with that, people tend to use out-of-context emojis as well at times which makes their semantics even more unpredictable.

In this project, I tried to provide a neural architecture prototype for modelling and prediction of emojis, exploring their relation and usage with different words. It is realized from the results that using vector word embeddings to generate sentence vectors and using them as feature space for classification could be an efficient and quite accurate for emoji prediction in tweets. It is also observed that in the classification task, the SVM classifier outperforms the KNN and Logistic Regression classifier models by a great margin. However, this comes at the cost of increased training time.

One major part of the difficulty in this task was working with a noisy dataset. However, the decent accuracy of the Word2Vec represented classification suggests that there still is an important and unique relation between the words and the emojis used in the tweet.

## FUTURE WORK

Although the initial development of the project looks promising, the investigated models still have a significant room to improve. Cleaning up the dataset to remove nonsensical tweets may help reduce noise and improve the classification performance. Incorporating deep learning models such as Long-Short Term Memory (LSTM) hidden layers and Convolutional Neural Network (CNN) would allow a deeper analysis of the words used resulting in a much better accuracy. Increasing efficiency of the model and extending it predict more than one emoji for a given tweet along with inclusion of more languages apart from English could be done.

## **ACKNOWLEDGEMENT**

I express my profound and sincere gratitude to my mentor Dr. Anil Kumar Singh for providing me with all the facilities and support during my winter internship period.

I would like to thank my guides Dr. Sanket Pathak and Mr. Rajesh Mundotiya for their valuable guidance, constructive criticism, encouragement and also for making the requisite guidelines enabling me to complete my work with utmost dedication and efficiency.

At last, I would like to acknowledge my family and friends for the motivation, inspiration and support in boosting my moral without which my efforts would have been in vain.

## REFERENCES

- [1] <http://time.com/4114886/oxford-word-of-the-year-2015-emoji/>
- [2] *"How to Develop Word Embeddings in Python with Gensim"*  
<https://machinelearningmastery.com/develop-word-embeddings-python-gensim/>
- [3] *"Embedding vector for word, query, sentence and paragraph"*  
<https://groups.google.com/forum/#!topic/gensim/bEskAT45fXQ>
- [4] Francesco Barbieri, Miguel Ballesteros and Horacio Saggion. *"Are Emojis Predictable?"* Large Scale Text Understanding Systems Lab, TALN Group Universitat Pompeu Fabra, Barcelona, Spain, 24 Feb 2017
- [5] Luda Zhao and Connie Zeng. *"Using Neural Networks to Predict Emoji Usage from Twitter Data."* Stanford University, 2017.
- [6] Barbieri, Francesco, Francesco Ronzano, and Horacio Saggion. *"What does this Emoji Mean? A Vector Space Skip-Gram Model for Twitter Emojis."* Language Resources and Evaluation conference, LREC, Portoroz, Slovenia. 2016.
- [7] P. Selvaperumal, A. Suruliandi. *"A short message classification algorithm for tweet classification."* Recent Trends in Information Technology (ICRTIT) Conference, 2014.
- [8] *"Top 10 Machine Learning Algorithms"*  
<https://www.dezyre.com/article/top-10-machine-learning-algorithms/202>
- [9] *"Semeval 2018 Task 2 Multilingual Emoji Prediction"*  
<https://groups.google.com/d/forum/semeval-2018-task-2-multilingual-emoji-prediction>
- [10] *Speech and Language Processing (3<sup>rd</sup> Edition)*. Book by Daniel Jurafsky and James H. Martin