Symbol Predictor from Sentence using RNN, LSTM

Abhishek Chintalapati
Graduate Student, Computer Science
University of Maryland, Baltimore County
Baltimore, USA
abhishc1@umbc.edu

Sai Sragvi Vibhushan Nidamarti
Graduate Student, Computer Science
University of Maryland, Baltimore County
Baltimore, USA
gt73213@umbc.edu

***Abstract*- Emojis are an awesome way to express. So, a machine which would suggest an appropriate emoji that can be used at the right time will do a great job! In this project, we investigate the relation between words and emojis, studying the novel task of predicting which emojis are evoked by text-based messages. The model will be predicting emoji's according to a given sentence using Deep Learning Models like RNN and LTSM. The goal of emoji prediction models is to be able to extract meaning and emotion for a short text and then predict an emoji that contains the same meaning and emotion accordingly.**

Key Words - Word embeddings, Natural Language Processing, LSTM, RNN.

I.INTRODUCTION

With the advent of social media, a new method of communication has emerged, one in which the meaning is conveyed through a combination of short texts and visual improvements, or symbols. This, Visual component of the message, is now the new default not only on Twitter, but also on other major online social networking sites such as Facebook, WhatsApp, and Instagram.

For indexing multimedia information, retrieval, or content extraction systems, understanding the meaning of emojis in relation to their context of use is critical. Furthermore, emoji can be used to enhance the meaning of a message, i.e., an emoji can be used to determine the emotion of a text; nevertheless, such emotive figures may become fragile when used in an ironic or mocking context.

Symbols can be thought of as a visual language for expressing emotions that is widely utilized on social media. They transmit more sentimental information and boost the efficacy of engagement mediated through an electronic device, in addition to textual write text messages that include context-relevant emojis. Nonetheless, emoji usage can be perplexing and sensitive, making emoji evaluation and prediction a substantial difficulty.

Emoji modeling and prediction is thus a critical issue in achieving the ultimate objective of accurately expressing the intended meaning of a social media communication. Emoji prediction, i.e., predicting the most likely associated emoji(s) for a given (usually short) message, may help to improve a variety of NLP tasks, such as information retrieval, generation of symbols from social media content, or emoji suggestion when writing text messages or sharing pictures online. It's also been useful for sentiment analysis, emotion recognition, and irony detection. Despite its newness, the problem of emoji prediction has already seen significant progress. for example, offer an LSTM model that, in some cases, beats a

logistic regression baseline based on word vector averaging and even human judgment.

In addition to emoji similarity datasets and emoji sentiment lexicons, the efforts listed above have opened the path for a greater understanding of emoji semantics. However, we only have a rudimentary knowledge of what the neural networks for emoji prediction are capturing right now. What factors does a model consider when linking a message with happy (), negative (), or sad () intentions, for example? Implementing an attention mechanism over the hidden states of LSTM layers would be a natural method to test this. In fact, attentive architectures in NLP have recently gotten a lot of attention, mostly for sequence to-sequence models (which are useful for machine translation, summarization, and language modeling), and a lot of different modifications have been proposed, such as additive, multiplicative, and self-attention mechanisms.

II. IMPLEMENTATION

In this section, I go over the Implementation of the project and the flow is shown in fig-1. Specifics of my neural model that is based on deep learning. The model's architecture is shown in fig-2.

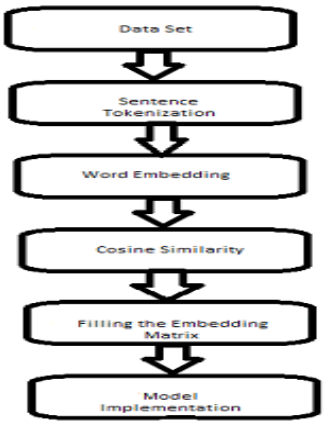

Fig -1

A. Dataset

We need a dataset with words, sentences, and other textual information representing human moods because the project revolves around words and emoji. The train dataset comprises 157 rows, whereas the test dataset has 68 rows, with the labels denoting the emotion being custom-classified numbers. We have manually added data in test and train in-order to achieve better performance and validation of the model. Below is the snippet of our labels -

B. Global Vector Embedding Layer

GloVe is a program that provides relevant information about the meaning of individual words and may be used to compute the correlation between them. Each word in a semantic vector formatted representation of a language is evaluated using a real-valued vector. These vectors are from a special weighted least squares model that uses statistics to train on a data set with word-by-word co-occurrence and frequency counts. The model generates a word vector space with various dimensional substructures such as 50, 100, and 150 dimensions. For example, cosine similarity (France-Paris) = 0.802 and cosine similarity (India - France) = 0.383.

The words France-Paris has a higher resemblance than the word India-France. The

proposed model can perform well on unseen or new words because it has this co-relational vector representation of all English dictionary words. The input statement will be converted to a GloVe formatted vector representation, and the average of the embedding layer will be supplied to the LSTM layer as a single encoded vector.

C. Long-short term memory layer (LSTM)

To evaluate the context of textual information, the LSTM layer has a fantastic feature of remembering the pattern of the statement for lengthy periods of time. The fact that LSTM is made up of different memory blocks called cells is a significant advantage. Cells are in charge of memory. The two states that are passed along to the next cell are cell state and concealed state. Manipulation of memory blocks is performed through three gates. To delete information from the cell state, the forget gate is utilized. To add information to the cell state, an input gate is utilized. The output get function is used to select meaningful data from the current cell and send it to the next cell as an output. To process the information and determine the true context of the input statement, all three gates act in tandem.

A type of RNN network is the LSTM. It is suitable for analyzing words because it allows prior outputs to be utilized as inputs while maintaining hidden states. The vanishing gradient problem is a problem with RNN that LSTM can solve. Finally, we employ a SoftMax layer, which generates a vector representing the probability distribution of a set of potential emoji.

The first layer of the *RNN* model is the embedding layer, which represents each word as a matrix. The embedding layer is loaded from a GLoVe model that has already been trained, and the weight is set during training. The Glove will be used in a high-dimensional feature space, map comparable words to close vectors. Two 1-D convolution layers follow, followed by a bidirectional LSTM layer.

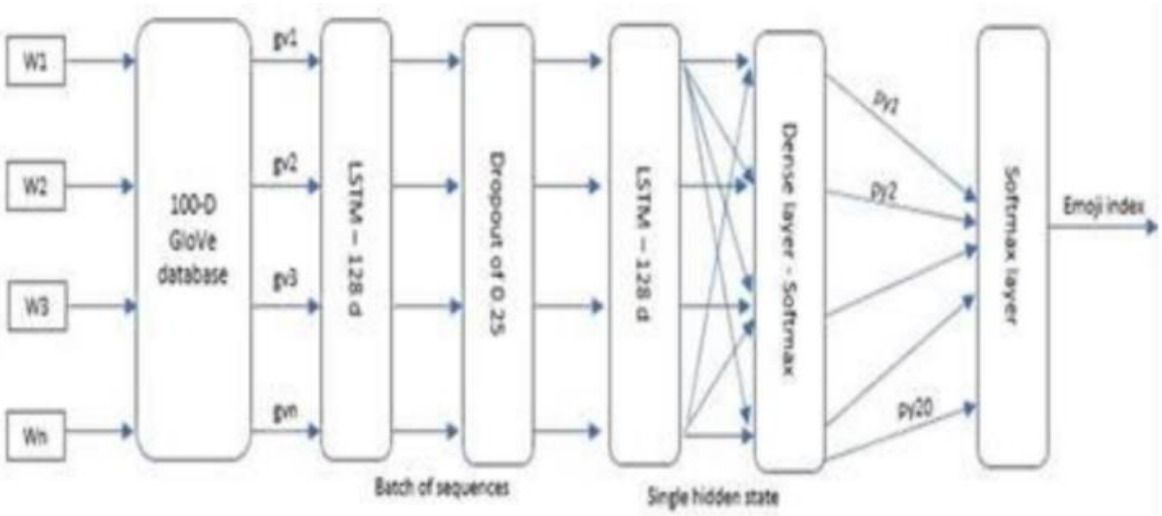

Fig-2

III. EXPERIMENTS & RESULTS

Now, we can go through the results of the experiment done in this section. Let us start by defining some abbreviations which will help us in understanding the data from the tables below.

SM – SoftMax Activation Function

SIG – Sigmoid Activation Function

DO – Dropout

LF – Loss Function

FL – Final Loss

Acc – Accuracy

CCE – Categorical_cross_entropy

MSE – Mean Squared Error

Table 1 and Table 2 showcase the results obtained from LTSM Model. We have come up with a couple of models which vary in the activation function, dropout and loss function used.

SIG	DO	LF	FL	Acc
	0.2	CCE	0.0035	55.8%
	0.2	MSE	0.0088	60.2%
	0.5	CCE	0.0803	58.82%
	0.5	MSE	0.0091	57.35%

Table 1

SM	DO	LF	FL	Acc
	0.2	CCE	0.0025	61.76%
	0.2	MSE	0.0054	61.76%
	0.5	CCE	0.0051	60.2%
	0.5	MSE	0.0225	63.2%

Table 2

Now we repeat the process for the RNN model as well and we can find the results in the Table 3 and Table 4.

SM	DO	LF	FL	Acc
	0.2	CCE	0.0031	63.23%
	0.2	MSE	0.0062	67.6%
	0.5	CCE	0.2129	55.88%
	0.5	MSE	0.0073	59%

Table 3

SIG	DO	LF	FL	Acc
	0.2	CCE	0.0157	58.8%
	0.2	MSE	0.0070	57.35%
	0.5	CCE	0.0424	52.9%
	0.5	MSE	0.0166	55.88%

Table 4

The precision and recall were almost the same as accuracy for every respective model. From these results, we can state that we were able to achieve our best accuracy with RNN and SoftMax activation function with a dropout of 0.2 (we are randomly avoiding 20% of the inputs from the activation function and formulation of weights to avoid overfitting).

Now, let us have a look at the confusion matrix for one model each from the RNN and LTSM slots. Fig 3 and Fig 4 show us these matrices plotted against true labels vs predicted labels. Each number in these labels

stand for an emoji (which we have described earlier).

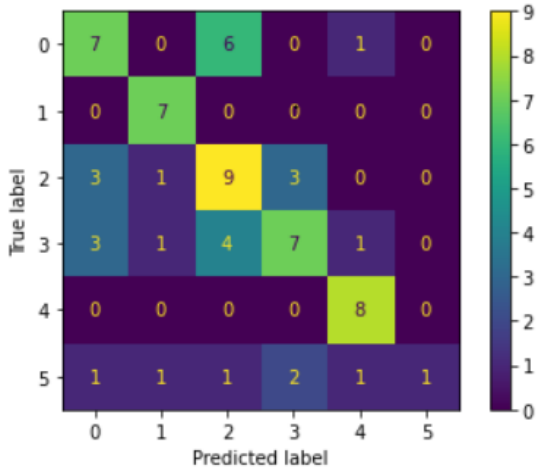

Fig 3

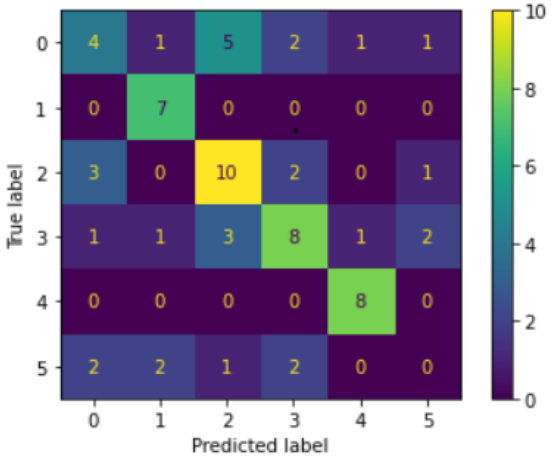

Fig 4

These are the results that we have obtained by running the model through the test data.
We have also tested against giving the model a random sentence as an input and see what emoji it predicts. You can see the results below in Fig 5. The sentence that was given as input is 'I am mad at you'. From the emoji's that we have taken into consideration for both building the models and the train and test datasets, the apt one would be the 'angry' face. We were able to predict this right by one of our models.

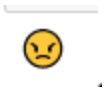

Fig 5

Below is a snippet of the misclassifications made by the highest accuracy model from our experiment. In Figure 6, we can see that the prediction would be right in sense of emotion. May be, if we get more train data then we could get the right emotion into picture by our model.

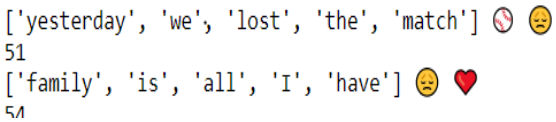

Fig 6

The left one is the predicted one and the right one is the true class label. Model was able to pick up keywords like 'match' and ['family', 'I'] but got the prediction wrong when mapped with the test data.

IV. Result Analysis

From the data presented, we can see that for few instances where we are only modifying the dropout, there was not much of an impact on the accuracy. This can be happening because of the size of the training data. May be, given a huge training data set, we can see a noticeable difference in the performance with a change in the dropout.

From these observations, we can say that for a multilabel data, a SoftMax activation function with Mean Squared Error is giving us the best results. This can be because that when compared with Sigmoid function, SoftMax is an extension to it. This works in favor of the multilabel data whereas sigmoid is a better performer when have binary labels.

V. CONCLUSION

In this study, we present a couple of recurrent neural network models using GloVe embeddings to predict the emoji symbol while comprehending the whole context of the statement and to develop the model in a simple and efficient manner. Tweaking the hyper parameters has become much faster with the use of GloVe embeddings, both for evaluating model performance and for adjusting hyper parameters. Of the models, we can say that RNN with SoftMax activation function yielded the highest accuracy.

Instead of an absolute single emoji, we should use an output of several predictions with decreasing confidence for future improvements. There isn't always a 1:1 correspondence between an emoji and a phrase or expression. When we ask a user to choose which emojis to use for a comparable statement, the results are frequently surprising. Furthermore, while calculating accuracy, it may be preferable to use weighted penalties to determine accuracy. With some decent thinking, the correlation matrix depicts a lot of the emoji overlap. Our accuracy is now predicated solely on absolute correctness, which is a difficult task for a model to meet given the problem's weak semantic character.

REFERENCES

Barbieri, Francesco, Miguel Ballesteros, and Horacio Saggion. ”Are emojis predictable?.” arXiv preprint arXiv:1702.07285 (2017).

Yoonjung Choi, Youngho Kim, Sung-Hyon Myaeng. Domain-specific sentiment analysis using contextual feature generation. ACM New York (2009).

Thomas Dimson. 2015. Emojineering Part 1: Machine Learning for Emo ji Trends (2015).

Mikalai Tsytsarau, Sihem Amer-Yahia, Themis Palpanas. Efficient sentiment correlation for largescale demographics. ACM New York (2013).

Spencer Cappallo, Thomas Mensink, and Cees GM Snoek. 2015. Image2emoji: Zero-shot emoji prediction for visual media. In Proceedings of the 23rd ACM international conference on Multimedia, pages 1311–1314. ACM.

Mayu Kimura and Marie Katsurai. 2017. Automatic construction of an emoji sentiment lexicon. In Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017, pages 1033–1036. ACM.