International Journal of Modern Agriculture, Volume 10, No.2, 2021
ISSN: 2305-7246

Using Natural Language Processing (Nlp) And Deep Learning To Analyze Sentiment In Memes

Roushan Kr. Giri[1], VineetChamoli[2], Prashant Singh[3], Subham Singh[4], Subhash Chandra Gupta[5]

[1,2,3,4] Department of School of Computing Science & Engineering, Galgotias University, Greater Noida, India
[5] Assistant Professor of School of Computing Science & Engineering, Galgotias University, Greater Noida, India
Email: [1]kr.beraw33@gmail.com, [2]vineetchamoli20@gmail.com, [3]prashant.singh081997@gmail.com, [4]virat7839@gmail.com, [5]subhash.chandra@galgotiasuniversity.edu.in

Abstract

Social media is the most popular interactive platform which is most common among people, today memes are a hot topic. Originally memes originated to have funny content but due to later development, it rather can be offensive sometimes, usually some hateful message or character image. Some memes may have abusive content and have the wrong impact on our society. It‘s difficult to categorize such memes with Human interaction This paper presents an approach to analyze and detect sentiment in memes using Natural Language Processing (NLP) and Deep Learning. It will detect offensive memes, in three steps. First, it will extract the text from the given image, then it will classify the given text as offensive or not offensive. If the text is found to be offensive then in the third step it will further classify offensive text in three categories namely slight offensive, very offensive, and hateful offensive. The model uses very simple architecture with a multi-layer dense network structure involving NLP with RNN and LSTM along with word embeddings such as GloVe and fast text.

Key words: NLP, CNN, LSTM, RNN, GloVe, fasttext

Introduction

The term ‘meme’ existed for a long time when Richard Dawkins, a British ethologist, used it to describe how cultural information spreads rapidly. He never thought that the word based on the context of biology would evolve into funny content. Memes are generally symbolic with some piece of text over them, generally hilarious. Recently with the rise of social media platforms, there has been a significant increase in the number of memes we find over the internet. The subject of the meme is rather very vast It can include the simplest of the topic to the most complex ones such as science, religion politics, and anything which one can think about. It can be a surprise to know how such meme pages generate revenues. Something which gets monetize must be kept in check as this could be misused. They affect the belief and view of a person in the given context.

Since the last few years, there has been a significant presence of teenagers on social media platforms (B. G.-Learning). Children nowadays are becoming obsessive about memes not just in image format but also in the video. Considering the impact these memes can have on teenagers, we need to be careful about memes and their content (S. S. Elayan and S. T. Jackson). Although we have norms to check on content which deals with harassment and hateful speeches it is simply not automated and we are aware of the risk it carries. Nevertheless, we don’t have any such norms regarding memes till now. Anyone can create a meme and circulate it all over the web. The content in the meme is solely based on the views of an individual or a community. A meme that may seem hilarious to one person can be offensive to another. It may lead to some unpredictable outcomes sometimes (N. Gal).

But due to the increasing number of memes over the internet, it is not an easy task to evaluate each meme before it spreads all around. This raises demand for a system that can automate the process of evaluating memes before they agitate a crowd or spread the humor. This paper presents a model to detect offensive memes, in three steps. First, it will extract the text from the given image, then it will classify the given text as offensive or not offensive. If the text is found to be offensive then in the third step it will further classify offensive text in three categories namely slight offensive, very offensive, and hateful offensive.

[2] Deep Learning and NLP can be used to perform the task of offense evaluation in memes spreading across the internet (T. Nasukawa and J. Yi)(J. Yi).

The whole researched paper can the briefly described as follows

- Extract text out of the given meme.
- Upsampling as data is not evenly distributed.
- Train and Test the model to classify if it's offensive or not offensive.
- For the memes that were classified as offensive, we trained another model to classify them as slightly offensive, or hateful offensive.

Literature / Related Work

There have been numerous research papers published on sentiment analysis and similar fields related to sentiment analysis. This section describes the previous research conducted on sentiment analysis, text-classification, and use of pre-trained models.[1] The most common approach in most of the works related to hate speech detection is to generate some kind of embedding, using N-gram features. More recently, in 2019, Dogu Tan Araci used pre-trained Glove word embeddings in his paper "FinBERT: Financial Sentiment Analysis with Pre-trained Language Models" for his master's thesis (Dogu Tan). [3] In 2019, A paper titled "Hate Speech in Pixels: Detection of Offensive Memes towards Automatic Moderation" was published by Anonymous authors (Sabat). Not much work has been done in classifying memes.

In 2017, Mathieu Cliche from Bloomberg published his paper, "Twitter Sentiment Analysis with CNNs and LSTMs" (Cliche), in which he used Convolutional Neural Networks(CNN) and Long Short Term Memory(LSTM) networks for his model. Also in 2019, A paper titled "Hate Speech in Pixels: Detection of Offensive Memes towards Automatic Moderation" was published by Anonymous authors (Sabat). Not much work has been done in classifying memes.

Methodology

The methodology used in this model is very typical to understanding any Deep learning model with CNN and RNN. We have tried to explain lucidly every point that we have used in developing this model to detect offensive memes. Contextually, Figure 1 gives a brief overview of the model understanding in brief.

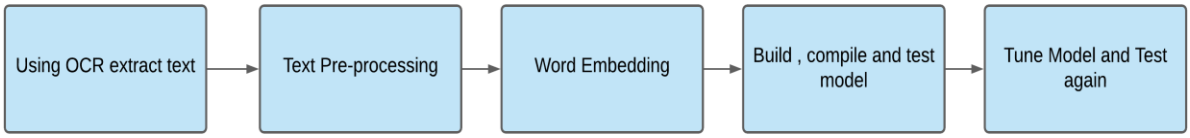


Fig.1. Shows the basic flow of the project

The very first step is to extract the text using already available OCR, which follows by initial pre-processing of text. We can use the regex tool which is very flexible in all programming languages to do so. We have used regex for data cleaning that includes removing stop-words or words that add no meaning to the result such as punctuations, emails, contact numbers, etc. After cleaning of data our next task was to split the text into an array of lists which is actually tokenizing and then furthermore assigning them with vocabulary index. For this whole process, we have used the NLTK library for python.

In figure 1.1 we have the data distribution of the data set:

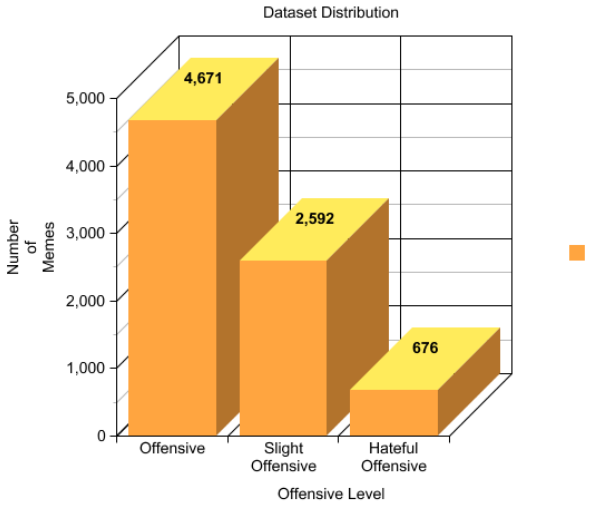


Figure 1.1 Distribution of the data

Word Embeddings:

[4] The most famous Word Embedding available to us is by Standford and Facebook. We have tried to build our model using the Word embedding

- GloVe Word Embedding by Stanford
- Fast text Word Embedding by Facebook

FastText word embedding:

[7] This method uses fast texts supervised model and fastText's 300M English Word Embeddings. It is a library for creating word embeddings and word classification. These pre-trained embeddings and models are used as per the requirement in our application. Meme classification also comes under sentiment analysis. So it was very intuitive to use this tool as we have seen fellow researchers use it in their research paper Sentiment analysis can be done in many ways i.e. by using Naive Bayes, using Neural networks (CNN, RNN with LSTM), etc. [6]

FastText's inbuilt models support mainly two models Supervised and Unsupervised.

The fast text.the supervised model takes two parameters

- The path of the training dataset file
- Hyperparameters

Data:

The train/test data that is given to the fast text model should be a .txt file and this file should contain a data object separated by commas. Each data object contains a label and the text related. And especially the label in the dataset .txt file should be in the form

_label_ACTUALLABEL

For example,

In this case it is: *_labeloffensive*, *__labelnot_offensive*, *__labelslight_offensive* etc.

FastText ,model:

#model=fasttext.train_supervised(input=path, hyper_params).

GloVe word embedding:

[3] GloVestands for global vector for word representation. It was developed by Stanford for generating word embeddings [4]. The resulting word embedding Fig1.2 shows linear structures of words in vector space along with Word2Vec comparison.

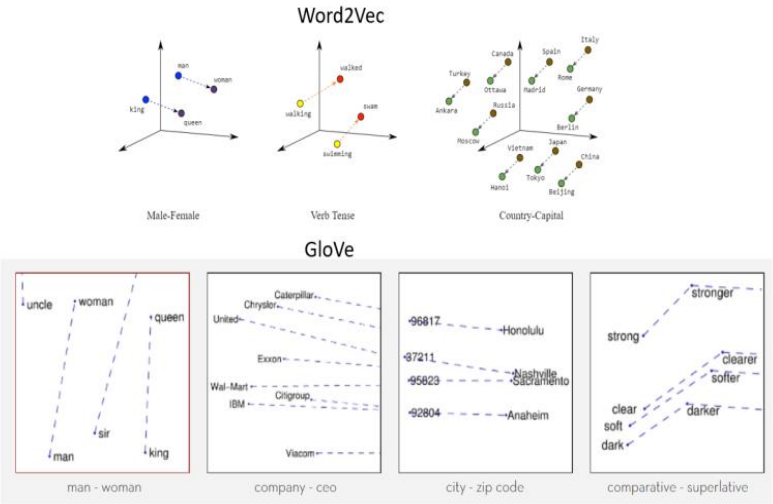


Fig 1.2 An example of linear substructures.

There are 4 pre- trained GloVe model and the one used in this project is glove.840B.300d.zip which has 840 billion tokens, 2.2 million vocabulary with the relative dimension of 300.The GloVe implementation in python is available in the library glove-python and can be installed using the command in window's terminal using pip install glove. In the case of GloVe, the counts matrix is preprocessed by normalizing the counts and log-smoothing them. Compared to word2vec, GloVe allows for parallel implementation, which means that it's easier to train over more data. [5] The glove model is trained using non-zero entries of global word-word occurrence matrix which tabulates how frequently words occur in the given sentence.

Model:

The Keras is the widely used library for Deep Learning. It is build above the Tensorflow framework. Keras has the functionality to define various types of model namely- Sequential and Functional. [5] The model built is the sequential which helps to create models layer by layer. It is already pre-defined in high level API's of keras and tensorflow as backend. We have used the Convolutional Neural Network (CNN) and Long Short Term Memory Network (LSTM) which is one of the variant of Recurrent Neural Networks..

Hyper parameters:

- The optimal epoch to which was model was trained is 7 with the batch size of 45 as we have less number of dataset so it tends to overfit.
- Dropout: We tweaked the model by tuning the dropout, basically it helped in regularization to reduce the overfitting and improve the generalization error when dropout was kept at 0.3.
- Activation functions: For the 1st Model we used the sigmoid activation as it was Binary prediction since we had to find out whether a meme is offensive or not offensive like 0 and 1.
- For the second model we used SoftMax as it works well with multiclass prediction.

- The loss function used for the 1st model was *Binary Cross entropy*and for the 2nd model it was categorical cross entropy.

Model Prediction:

The model built successfully will help to predict if the meme is offensive or not and if the meme is offensive it will try to predict the category of offensiveness i.e. is it slight offensive or hatefully offensive which can be considered very severe or very offensive that can hurt the sentiment and emotion of people. The workflow of the task can be seen in figure 2

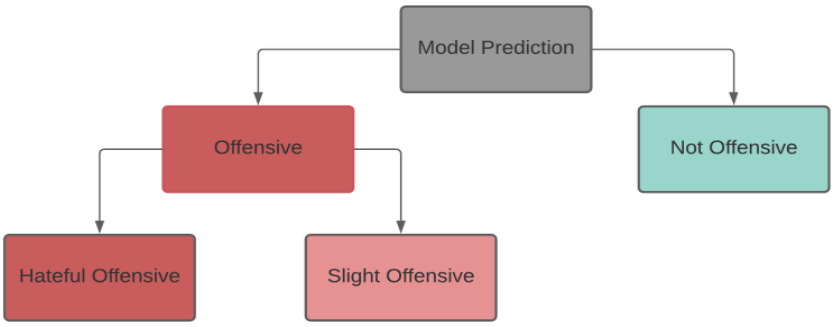


Fig 2: Model Prediction

Experimental Values

This shows the result of our experimentation. We have already mentioned the approach and the number of epoch to which we have trained. We have also mentioned the hyper-parameters used in the approach. You are free to tweak the hyper-parameters and the epoch to adjust your result.

After compiling the model that we have built using CNN amd RNN, the results are mentioned below.

For model result using fasttext refer to table 1 below:

Table for the fasttext model

Parameter	**Model 1**	**Model 2**
Training Accuracy	82.05 %	86.76%
Training loss	0.1795	0.1324
Val Accuracy	80.72 %	70.11%
Val Loss	0.1928	0.2989

Table 1: Results of the model(fasttext)

For the model result using GloVe refer to the table 2 below:

Table for the GloVe Model

Parameter	**Model 1**	**Model 2**
Training accuracy	85.52 %	98%
Training loss	0.1448	0.05
Val Accuracy	74.68 %	80 %
Val loss	0.2532	0.20

Table 2: Results of the model (Glove)

Performance of the model using GloVe embedding graphs:

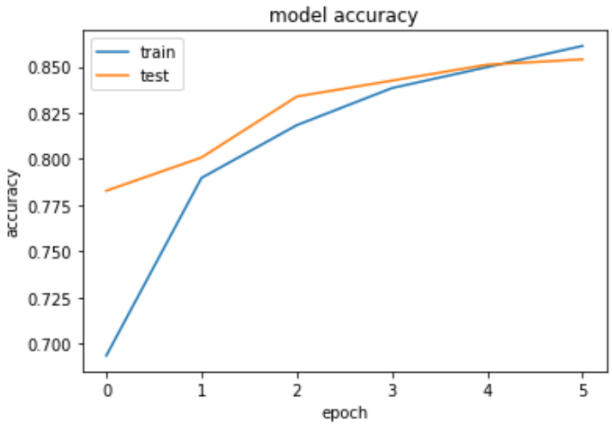
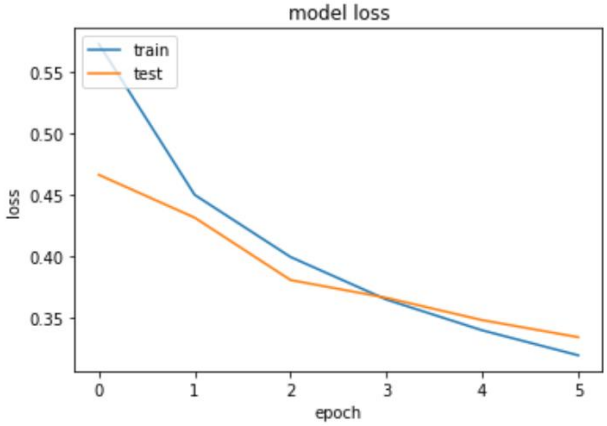
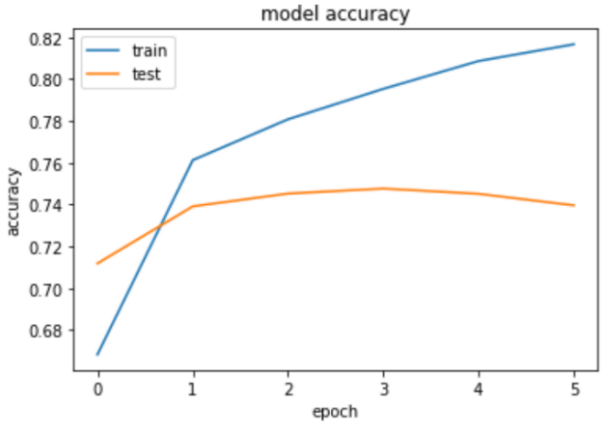


Figure 2.1 Graph shows the model accuracy and loss using glove



Performance of the model using fasttext:

The training is simple and fast using the fasttext model if we use the model provided by them. Upsampling is done so that we do not overfit the model. The following figure 2.2 shows the average loss



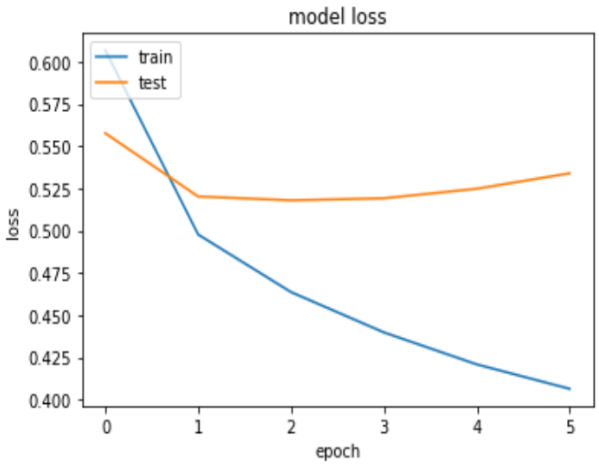


Figure 2.2 shows the model accuracy and loss for the second model.

Performance of the model using fasttext:

The training is simple and fast using the fasttext model if we use the model provided by them. Upsampling is done so that we do not overfit the model. The following figure 2.2 shows the average loss

The source code of my project is given below in this repository along with the dataset used:

https://github.com/roushangiri/Offence-Evaluation

Conclusion

Our work on offence evaluation in meme proposed a filter which may be used in the future for filtering many memes across the networking sites as it has become the norm for people to actively share memes. The time and technology have been evolving and hence a simple model like this alone won't be able to detect and evaluate meme, it may also require human moderator as well because sentiment is something very abstract in nature but nevertheless it can provide a solid group on harmful content. Hence, a further and more advanced implementation of this model with more feature engineering can increase the accuracy to a greater extend. There is always a room for improvement and so be it..

References

1. Hate Speech in Pixels: Detection of Offensive Memes towards Automatic Moderation . Project work by Benet OriolSabat, Cristian Canton Ferrer, Xavier Giro-i-Nieto https://arxiv.org/pdf/1910.02334.pdf
2. T. Nasukawa and J. Yi, "Sentiment Analysis: Capturing Favorability Using Natural Language Processing," 2003.
3. Medium blog https://bit.ly/3p52b5o
4. Glove Vector for word representation from the website https://nlp.stanford.edu/projects/glove/
5. Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In Proceedings of the 21st ACM international conference on Information and knowledge management, pages 1980–1984. ACM.
6. An Approach to Detect Offence in Meme using NLP and Deep Learning
7. IEEE Xlpore Evaluation of Deep learning Techiques in Sentiment Analysis from Twitter Data 2019
8. Keras CNN with FastText embedding https://github.com/facebookresearch/fastText/
9. FastTextreference .https://github.com/facebookresearch/fastText/
10. B. G.- Learning, M. and Technology, and undefined 2018, "Thinking in hashtags: exploring teenagers' new literacies practices on Twitter," Taylor Fr.

11. Mathieu Cliche. 2017. BB twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNsand LSTMs
12. Dogu Tan Araci. 2019. FinBERT: Financial Sentiment Analysis with Pre-trained Language Models.