An approach to detect offence in Memes using Natural Language Processing(NLP) and Deep learning

Roushan Kumar Giri
Student, SCSE
Galgotias University
Greater Noida, India
kr.beraw33@gmail.com

Subhash Chandra Gupta
Assistant Professor, SCSE
Galgotias University
Greater Noida, India
subhash.chandra@galgotiasuniversity.edu.in

Umesh Kumar Gupta
Assistant Professor, SCSE
Galgotias University
Greater Noida, India
umesh.kumar.gupta@galgotiasuniversity.edu.in

2021 International Conference on Computer Communication and Informatics (ICCCI) | 978-1-7281-5875-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCCI50826.2021.9402406

***Abstract*— Social media is one of the most popular form of platform which is most common with people of our age. With passage of time, memes have gained a significant popularity and are often shared on social media platforms. Memes usually have hilarious content but can be offensive sometimes, containing some hateful message or character image. Such memes may have detrimental social impact in our society or to any individual. [1] Thus, an automated system for evaluation of offensiveness in the meme content is required. This paper presents an approach to detect offense in memes using Natural Language Processing (NLP) and deep learning. 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 a demand for a system that can automate the process of evaluating memes before they agitate a crowd or spread a humour. 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. The dataset used for this work consists of 6,992 memes which were labeled as not offensive, slightly 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 FastText.**

Keywords*— NLP, Offense Evaluation Memes, GloVe, FastText.

I. INTRODUCTION

The term 'meme' was given by Richard Dawkins to describe how cultural information spreads rapidly. Memes are generally images with some piece of text over them, generally humorous. 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 meme culture has been trending since last few months. These memes may contain content based on day-to-day activities across the world, science, religion politics and anything which one can think about (O. Goriunova). These memes are spread as rapidly as fire all over the internet, and thus come in the reach of a large proportion of the population. They may 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 the memes. 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). But 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 meme is solely based on 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 a demand for a system that can automate the process of evaluating memes before they agitate a crowd or spread a humour. 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 main contribution of this research article is summarized as follows.

1) Prepared a model to extract text out of the given meme.

2) Trained a model to classify given text as offensive or not offensive.

3) For the memes that were classified as offensive, we trained another model to classify them as slightly offensive, very offensive, and hateful offensive.

II. LITERATURE / RELATED WORK

Several researches have been, and are still being experimented in the field 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

978-1-7281-5875-4/21/$31.00 ©2021 IEEE

features. In 2010, Razavi applied Naive Bayes (Amir H Razavi), and in 2012, Warner and Hirschberg (wide W. W.) used Support Vector Machine (SVM), and word-tokens to classify offensive language. [2] Also in 2012 Xiang in collaboration with Latent Dirichlet Allocation (Guang Xiang) generated topic distribution also using word-vector features in order to classify offensive messages on the twitter website. 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).

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.[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.

III. METHODOLOGY

The methodology used in this model isn't a rocket science to understand. We have tried to explain lucidly every point that we have used in developing this model to detect offensive memes. Contextually, it can be broadly divided into segments described below in the figure 1.

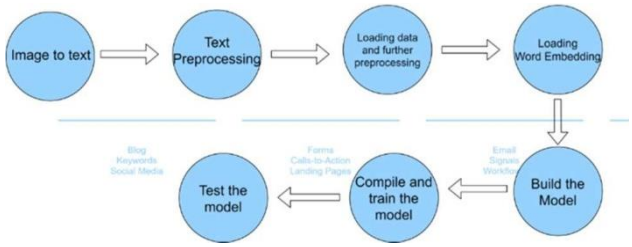


Fig.1. Shows the basic flow of the project

The initial task was to pre-process the text for which we used regex to omit unwanted tokens in our data set (punctuations, emails, contact number, etc) and the very famous *NLTK library* which indeed helped us (1) splitting the words, (2) removing the stop words, (3) also to tokenize the text into list, furthermore assigning them the respective vocabulary index.

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

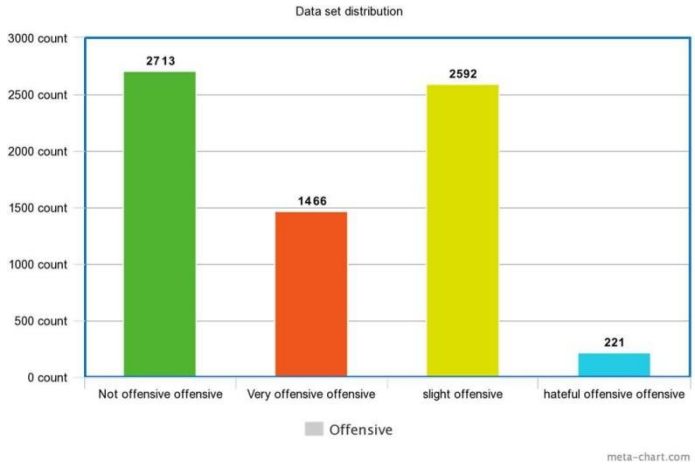


Figure 1.1 Distribution of the data

A. *Word Embeddings:*

[4] Word Embedding is the heart of the project. We have tried to build our model using the Word embedding

- GloVe
- FastText

B. *FastText word embedding:*

[7] It is a library for creating word embeddings and word classification, developed by Facebook. These pre-trained embeddings and models are used as per the requirement in our application. Meme classification comes under sentiment analysis. Sentiment analysis can be done in many ways i.e. by using Naive Bayes, using Neural networks (CNN, RNN with LSTM) etc. [6] This method uses FastText's supervised model and FastText's 300M English Word Embeddings.

FastText's Supervised Model:

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

The FastText supervised model takes two parameters

- The path of the training dataset file
- Hyper parameters

Data:

The train/test data that is given to the fasttext model should be a .txt file and this file should contain 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).

C. GloVe word embedding:

[3] GloVe stands for global vector for word representation. It was developed by the Stanford for generating word embeddings [4].The resulting word embedding Fig1.2 shows linear structures of word in vector space.

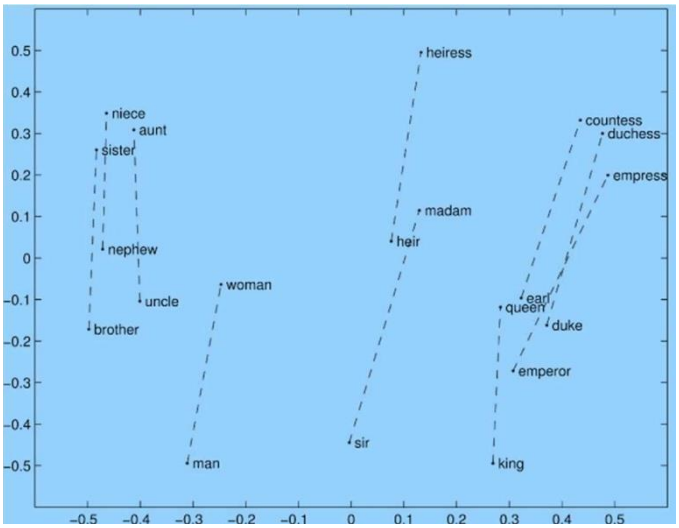


Fig 1.2 An example of linear substructures.

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. [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. 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.

D. Model:

We have used two different types of deep neural networks in this paper. Convolutional Neural Network (CNN) and Long Short Term Memory Network (LSTM), which is one of the variant of Recurrent Neural Networks. [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.

The first hidden layer is the embedding layer which takes all of our input words that are equivalent to the vocabulary size and the weight defined by embedding matrix created using the pre-trained embedding model.

E. Hyper parameters:

- 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 0.2.
- Activation functions: For the 1st Model we used the sigmoid activation for Binary prediction as we had to predict whether a meme is offensive or not offensive.
- For the second model we used SoftMax as it worked well with multiclass prediction to predict if the meme was highly offensive, slight or hateful offensive
- The loss function used for the 1st model was *Binary Cross entropy* and for the 2nd model it was categorical cross entropy.
- The optimal epoch to which was model was trained is 5 with the batch size of 32.

F. 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

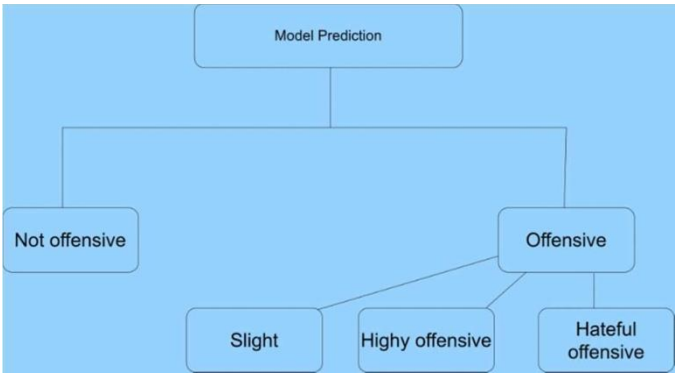


Fig 2: Workflow of the task

IV. EXPERIMENTAL VALUES

Let us visually see the data set provided to us for analysis of the sentiment of the meme in the figure 1.1

The data set can be viewed using command:

import seaborn as sns

sns.countplot (x='offensive', data=df)

We have already discussed the approach used in training the mode and also the word embedding. So, after compiling the model and running for few epochs, we have the following results for the both models.

A. For model result using fasttext refer to table 1 below:

Table for the FastText model		
Parameter	**Model 1**	**Model 2**
Training Accuracy	0.9571	0.997
Training loss	0.1075	0.13
Val Accuracy	0.5375	0.998
Val Loss	1.5436	0.37

Table 1: Results of the model(FastText)

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

Table for the GloVe Model		
Parameter	**Model 1**	**Model 2**
Training accuracy	0.93	0.980
Training loss	0.116	0.05
Val Accuracy	0.7086	0.80
Val loss	1.4058	0.79

Table 2: Results of the model (Glove)

C. Performance of the model using GloVe embedding graphs:

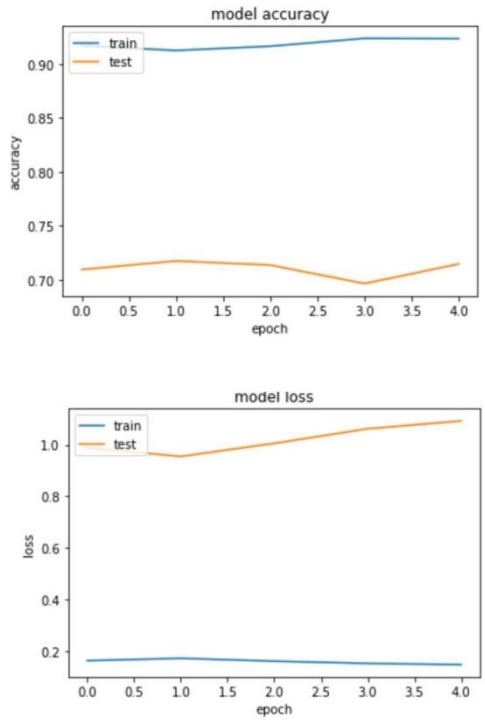
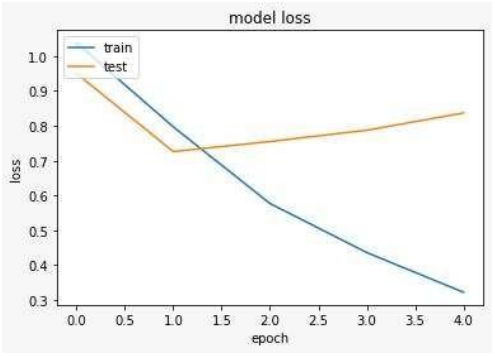


Figure 2.1 Graph shows the model accuracy and loss on first Model



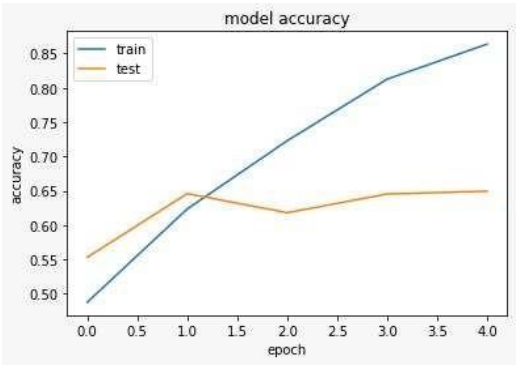


Figure 2.2 shows the model accuracy and loss for the second model.

D. 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

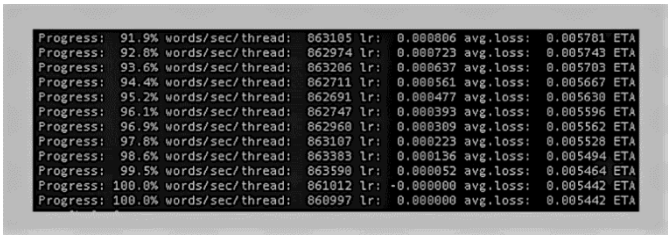


Figure 2.3 shows the average loss of second model using fasttext.

The source code of my paper 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 Oriol Sabat, 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] IEEE Xlpore Evaluation of Deep learning Techiques in Sentiment Analysis from Twitter Data 2019 https://ieeexplore.ieee.org/document/8876896/authors#authors

[7] Keras CNN with FastText embedding https://github.com/facebookresearch/fastText/

[8] FastText reference . https://github.com/facebookresearch/fastText/

[9] B. G.- Learning, M. and Technology, and undefined 2018, "Thinking in hashtags: exploring teenagers' new literacies practices on Twitter," Taylor Fr.

[10] Mathieu Cliche. 2017. BB twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNsand LSTMs

[11] Dogu Tan Araci. 2019. FinBERT: Financial Sentiment Analysis with Pre-trained Language Models.

[12] Python for NLP: Working with Facebook fasttext https://stackabuse.com/python-for-nlp-working-with-facebook-fasttext-library/