IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

ANALYSIS OF EMOTIONAL MODEL ON TWEETS USING DATA MINING

[1]M.Priyadharshan, [2]B.Gayathri,

[1]Assistant Professor, [2] Assistant Professor,
[1]Computer Science and Engineering,
[1]Hindusthan college of Engineering and Technology, Coimbatore, India
[2] Mount Zion College of Engineering and Technology

Abstract: This approach gives an advanced alternative to the existing Sentiment analysis models. Emotional Models are a bit trickier than the traditional Sentiment Models as they analyze human emotions by identifying the context of a text, which is different from a Sentiment Model where only the polarity of the text is calculated and the prediction is either positive or negative. The Model requires a huge dictionary of lexicons which points to different emotions. This can be used to analyze a text based on the lexicons thatbelong to a particular human emotion. The words in the text are embedded with low dimensional vectors, which makes it easier to use in the Neural Network Models. The purpose of emotional analysis is to identify the sentiment of a given text. In general, emotional analysis can predict emotions such as anger, happiness, surprise, sadness and so on. The classification of the emotions can only be achieved through psychological models that define human emotions in terms of mathematical values. Emotion Models are built based on Natural Language Processing and Text Analysis using Deep Learning Techniques. Tweet Analysis helps in classifying data about a specific brand or a product and its interest among the users.

IndexTerms - Twitter Data, Sentiment analysis Attention based Bidirectional LSTM, BERT, Deep learning

I. Introduction

Sentiment analysis emphasizes the fact that opinion mining is a natural language processing activity, where we want to evaluate what the general feeling of the document is. Using natural language processing and deep learning techniques, we can extract the subjective awareness of the text and try to classify it in terms of its polarity as negative, neutral and positive. It’s a really useful tool, because then we can theoreticallydetermine the general sentiment of a sales product, or predict the stock markets for a given business like, if more people think it’s nice, maybe stock prices will increase, and so on.It is intended to serve as an application to understand the opinions, thoughts and feelings expressed in the context of an online resource. Many people use social networking sites to network with others and stay up-to-date with news and currentaffairs. These sites (Twitter, Instagram, Snapchat, and Facebook) provide a forum for people to express their views. For instance,people will automatically post their reviews online as soon as they see a movie and then start a series of discussions about the acting qualities shown in a movie. This type of data is the criterion for people to analyze, determine the quality not only of any film, but also of other products, and to know whetheror not it would have been a success. These sources of tweets are usually loud, representing shifting views, multi-topic information in an unstructured, unfiltered format. Analyzing sentiment expressed, however, is not an easy task. There are several problems in terms of tonality, polarity, lexicon and tweet grammar. They seem tobe highly informal and pseudo-grammatical.

It’s going to be hard to grasp their background. In contrast, the regular use of slang words, acronyms and vocabulary words is very popular when posted online. The categorization of such terms by polarity is difficult for the natural processors involved in this project, to identify negative, neutral and positive tweets. We have obtained results using Bidirectional- Attention based LSTMs, CNNs and fine-tuning Google’s pre-trained BERT architecture, which has generally performedas a state of the art for most NLP tasks.

II .Related Works

While several techniques for sentiment analysis have been built over the years, none of them have been extensively tested with deep learning methods on the Twitter data. Inthis section, we will discuss some of the previous researchthat has been done on these issues. The function of sen- timent analysis is one of the oldest and most significant processes for the development of NLP applications. A variety of methods, such as SVMs and Naive Bayes, have been tried to solve this problem. However, as with most other machine learning activities, the past few years have seen a move away from conventional machine learning approachesto increasingly efficient neural network architectures. Prior to the release of BERT, which can be fine-tuned for most naturallanguage processing activities, state-of-the-art output on most sentiment classification activities was achieved through VD-CNN frameworks first introduced by Facebook Research [1]. Using CNN multi-layer and per-trained embedded vectors, the

Facebook team was able to successfully define polarity inlonger-form text reviews by Amazon and Yelp with 95.72 % accuracy. However, their model has not been tested on Twitterdata.

Using SVMs for emotion analysis, [2] collected optimistic, negative and neutral tweets from a number of sources, in- cluding the Sentiment140 database. The community extracted features from each message, such as characters n-grams, number of hashtags, emoticons, etc., for their classification andobtained an F1 score of 69.02 for their 3-way classification. Authors in [3] carried out similar studies in the assessment of document classification techniques. The group trained several models in the SemEval dataset for the 3-way Tweets classification and obtained the strongest results using the Bi-LSTM model with an F1 value of 68.5.

III .PROCESSING APPROACH

In the processing Deep Learning uses classifiers to categorize sentiments into neutral, negative or positive.

- Pre-Processing Data

There are various ways to collect the data. Some methods found in literature build a system to automatically assemble a pool of tweets focused on "positive" and "negative "sentiments, by parsing the Twitter data with two forms of emoticons.

- Positive emoticons
- Negative emoticons

Others render their list of tweets by manually storing andtranscribing them, which is quite lengthy and tedious process. For the sake of study presented in this paper, we are using SemEval-2016 Twitter data set. It consists of three columns that are *Item ID*, *Sentiment* and *Sentiment Text*. Table I dis-playing the four twitter articles. We may already find certain peculiarities and difficulties we may experience during the preprocessing phase.

Table 1: example of twitter tweets transcribed with their respective sentiment

Item ID	Sentiment	Sentiment Text
628949369883000832	negative	dear @Microsoft ...
628976607420645377	negative	@Microsoft how ...
629186282179153920	neutral	If I make a game ...
629226490152914944	positive	Microsoft, I may ...

The accuracy of the classification can be related to thetweet's grammar layout, or whether a post is subjective or objective, etc. It is extremely difficult to deal with languages, and even more so when we try to translate text typed byusers on the internet. People don't like waste time making grammatically correct sentences in most situations, just use a lot of acronyms with phrases that are less than proper English.It may be for many reasons, such as the restriction of the number of characters permitted or the inability to actuallywrite the correct spelling or to follow a common lingo forthe special word. Fig. 1 describes a bar plot as per different sentiments in our data set.

Once we've got the pool of tweets and all the tools that might be helpful, we can pre-process the posts. This is really important because all the modifications which we are attempting to do during this phase would have a critical impacton the efficiency of the classifier. The pre-processing includescleaning, transformation, removing noise, etc. The outcomeof the pre-processing stage must be uniform and consistent to data to maximize the classifier's performance.

a) Emoticons: Using the emoticon dictionary, we over- ride all emoticons by their polarity of positive or negative sentiments. We run through each tweet to do the substitution and, using a regex, we figure out if it includes emoticons, ifso, they are substituted by their respective polarity.

b) URLs: We removed all the URLs with open spacesas they are not useful for our case, so just like emoticons, we used regex to find all URL's and deleted them. In this case regular expression matching generally works well.

c) HTML Entities: HTML instances are characters/tags that are defined in HTML. To have character entities we need to decipher them to make them understandable. The package Beautiful Soup is used to remove data from HTML scripts. The name of the Beautiful Soup library is bs4 that stands for Beautiful Soup, version 4.

d) Case: While it is often important to distinguish be- tween the proper nouns and other words, when working to find feelings, we did not see much of a difference. Consequently, the reduction of all letters to the lowercase does not result in any loss for the reason.

e) @-Mentions: The objective refers to the usernamesin Twitter that are followed by the '@' symbol. It's used to send a tweet to another tagged user, or it's just to grab user's attention. Hence, we removed all usernames.

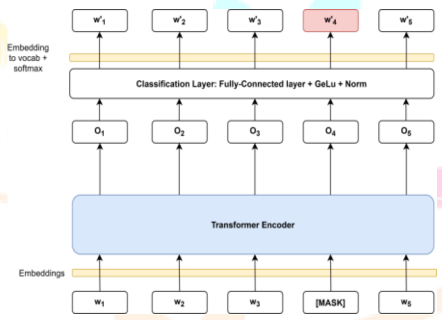
f) Hashtags/Numbers: Often the text used with hashtag will also provide valuable details about the tweet. Along with the hashtag it may be a bit tempting to get rid of all the text, but here we removed hashtags and numbers for convenience.

g) Acronyms: We substitute all the acronyms in our data set with their translations. The acronym is nothing but a shortened form, created by a phrase or a word from theirinitial components. Typically, these elements are individual characters, parts of names or words. Fig. 2 describes a bar chart showing several acronyms that are included in our tweet data set collection.

h) Repeated Characters: We substitute two characters for all sequences of recurring characters. (e.g: "helloooo" = "hello") to preserve the word's accented meaning.

- Classification Model

We used SemEval -2016 Twitter Dataset for sentiment analysis and built a system based on Convolutional Neural Networks with filter sizes [3,4,5] and pretrained Glove embed- dings, Bidirectional-Attention based LSTMs with pretrained Glove embeddings and Google BERT. The sentiment training set consists of raw tweets labelled neutral, negative and positive. In an effort of making state-of-art sentiment classifier,we compare and explore all three models.



w'1
w'2
w'3
w'4
w'5
Embedding to vocab + softmax
Classification Layer: Fully-Connected layer + GeLu + Norm
O1
O2
O3
O4
O5
Transformer Encoder
Embeddings
w1
w2
w3
[MASK]
w5

Fig. 1. BERT architecture

BERT*:* Bidirectional Encoder Representations fromTransformers also known as BERT, released at the end of 2018[4]. BERT is a model of pre-training language representations that has been used to build models that NLP professionalscan access and use for free. BERT uses Transformer, is adeep learning model which mainly uses attention to recognize the contextual relationship between words (or sub-words) in the text. Transformer comprises two discrete mechanisms inits elementary form – an encoder that reads the input of text, and a decoder that generates a prediction for the task. Because the aim of BERT is to create a language model, it justrequires the encoder mechanism. The entire input sequenceis read only once by the transformer encoder, as opposed to directional models which reads the input sequentially (rightto left or left to right). It is therefore called bidirectional.This feature allows the model to understand the meaning ofa word depending on all its neighbours (left and right of the word). Fig. 1 describes BERT architecture. Transformer was introduced in the 2017 paper "Attention is All You Need" [5]. In specific tasks, the Transformers outperform the GoogleNeural Machine Translation model. A stack of encoders formsthe encoding part. The decoding part is a stack of the same counted decoders. In structure the encoders are all similar. Each of them is split into two sub-layers: the inputs of the encoder first pass through a self-attention layer – a layer that lets the encoder looks at the other terms in the input expressionas a specific word is encoded. The inputs for the feed forward network come from the outputs of self-attention layer. Decoder has both layers, so the attention layer between them allowsthe decoder to concentrate on the appropriate parts of theinput paragraph. When the model processes each word, self- attention helps you to look at certain positions of the input textfor hints that may help to contribute to a improved encodingof the word. First step is to determine the Value, Key, and Query matrices. The outputs of the self - attention layer are computed as in :

$$Z = \text{softmax}\ Q\ KT\ \sqrt{}\ d. \qquad (1)$$

In which: K: key matrix, Q: query matrix and d_k: dimensionof key vector. We used the pretrained BERT model from huggingface transformer library for PyTorch [6].

a) Word Embeddings: In the case of a sentence con- sisting of K words U = $u_1, u_2, ..., u_K$, each word u_i is transformed into a vector e_i. With each word in U, first we look in the embedding matrix E, with L is set vocabulary size, and d^w refers to the embedding size. The word embedding $\mathbf{e}_i$ for a word $\mathbf{u}_i$ is calculated using the following product:

$$\mathbf{e}_i = \mathbf{W}\mathbf{l}_i \qquad (2)$$

where li: vector of size L. The embeddings $e_1, e_2, ..., e_K$ are then fed to the next layer.

b) Bidirectional Network: LSTM units were introduced in the first place by Hoch Reiter and Schmidhuber in 1997 [7] to resolve the vanishing gradient problem. The key concept is to implement an efficient gating mechanism, which will decide the extent to which LSTM units retain the derived features of the new data input and hold the previous state. Usually, the LSTMs are composed of four components: input gate i_t and its subsequent weight matrices $W_{ui}, W_{vi}, W_{ci}, b_i$; forget gate and its subsequent weight matrices $W_{uf}, W_{vf}, W_{cf}, b_f$;

Output gate and its subsequent weight matrices W_{uo}, W_{vo}, W_{co}, b_o, p_{i-1} refers to the current state of the cell and v_{i-1}state generated by the previous stage. The equations below

Demonstrate whether to forget the stored memory, take inputs and output the generated state later.

$$i_t = \sigma(W_{ui}u_t + W_{vi}v_{t-1} + W_{pi}p_{t-1} + b_i) \qquad (3)$$

$$f_t = \sigma(W_{uf}u_t + W_{vf}v_{t-1} + W_{pf}p_{t-1} + b_f) \qquad (4)$$

$$h_t = tanh(W_{up}u_t + W_{vp}v_{t-1} + W_{pp}p_{t-1} + b_p) \qquad (5)$$

$$p_t = i_t h_t + f_t p_{t-1} \qquad (6)$$

$$o_t = \sigma(W_{uo}u_t + W_{vo}v_{t-1} + W_{po}p_t + b_o) \qquad (7)$$

$$v_t = o_t tanh(p_t) \qquad (8)$$

We used BiLSTM in this paper. Shown in Fig. 4, our network includes two sub-networks for the sense of the right and the left context, both backward and forward respectively. Equation

(9) shows the output of the j^{th} word [8]:

$$v_j = [\bar{\vec{v}}_j \oplus \bar{\overleftarrow{y}}] \quad (9)$$

outputs of forward and backward pass are combined using element-wise sum.

Attention: Attention mechanism based neural net- works have recently been shown to be effective around the board for a variety of tasks starting from speech recogni-tion, machine transliteration, image captioning, to questionanswering. The encoder produces a matrix V consisting of$[v_1, v_2, ..., v_T]$ consisting of output vectors, where sentence length is T. The below equations shows the representation yof the input sequence is given by a weighted - sum of these output vectors [8]:

$$Z = tanh(V) \quad (10)$$

$$\beta = softmax(w^T Z) \quad (11)$$

$$y = V \beta^T \quad (12)$$

a) *Convolutional Neural Networks:* Convolution is a method for gathering input data and choosing a set of features from it. It consists of a series of layers. It requires data thatis preprocessed and translated to a vector format, and then the result is moved to a convolution layer. Convolution layer output undergoes pooling the mainly used method of poolingis max pooling, dropout is done after pooling to increase the accuracy [9]. A convolution process requires a filter, then it is convolved with the input maintaining a window of m words toconstruct a new feature. This filter is extended to any possible

window of words $\{u_{1:m}, u_{2:m+1}, ..., u_{n-m+1:n}\}$ to generate a feature maps [10].

$$f = [f_1, f_2, ..., f_{n-m+1}] \quad (13)$$

The obtained feature maps are subjected to maxpooling opera-tion where we take the maximum value of $\hat{f} = max\{f\}$. The aim is to retrieve the most appropriate feature for each feature map the one with highest importance. Naturally, this pooling scheme tackles sentences with variable lengths [11]. Suchfeatures then constitute the penultimate layer and transferredto a fully connected softmax layer whose non - normalized output corresponds to a probability distribution over predicted labels. The pretrained word vectors can be kept static whilethe other parameters of the model learn or we can fine-tune. The word vectors after few epochs when the rest of our model learns something useful. Dropout is used on the penultimate layer for regularization with a restriction on the weight vector'sl_2-norms. Dropout prohibits hidden units from co-adapting by dropping out a portion k of the hidden units randomly during both forward, back-propagation [12]. Gradients flow through only the unmasked cells. Shown in Fig. 2 the CNN architectureused.

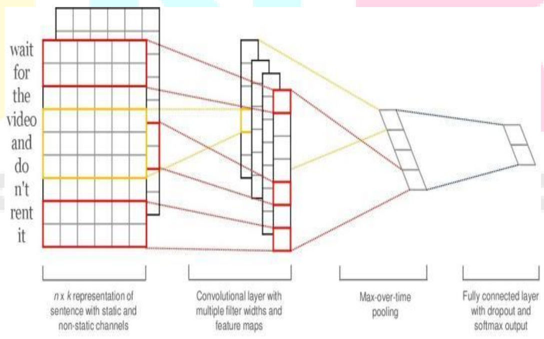

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Fig. 2. CNN architecture

IV. Experiments and Evaluation

4.1 Experiments

We trained three classifiers based on a SemEval-2016 twitter dataset (Subtask A), the training set contains 6000 tweetsin which 3,094 tweets labeled positive, 2,043 tweets labeled negative and 863 tweets labeled neutral. We also evaluatedthe efficiency of the classifiers on SemEval-2016 test set containing 20,632 tweets. Table II depicts the particulars of ourdataset and number of instances in each class label [13]. The experiment shows that the Google BERT outperforms both Bi -Attentive LSTM + Glove Embedding's(Grad) with embedding'sdimension of 50 and Convolutional Neural Networks + Glove Embedding's(Grad) with embedding's dimension of 50 and windows of sizes (3,4,5). In addition, we pre-processed the rawtweets and built new data set by filtering hashtags, numbers, emoticons, URLs, acronyms from our data set and applied the same transformations on both validation and test data. Subtask A is a *single-label multi-class* classification task. Each tweet must be classified as belonging to exactly one of the followingthree classes $C = \{Positive, Neutral, Negative\}$.

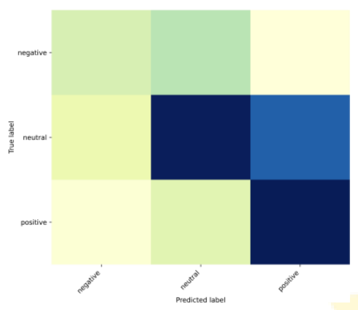


Fig. 3. Confusion Matrix Bi-Attentive LSTM

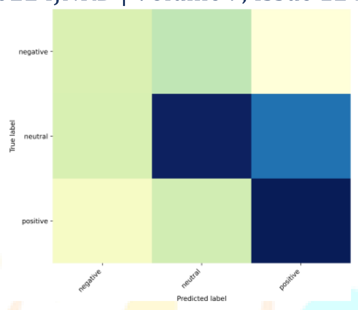


Fig. 4. Confusion Matrix CNN

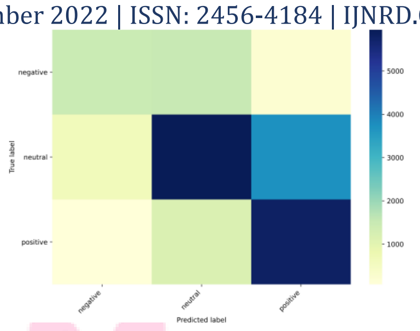


Fig. 5. Confusion Matrix BERT

Table 2: Data statistics

	Positive	Neutral	Negative	Total
TRAIN	3,094	863	2,043	6,000
DEV	844	765	391	2,000
DEVTEST	994	681	325	2,000
TEST	7,059	10,342	3,231	20,632

Note that we can use two approaches to test our classifier, the F1 score and confusion matrix. Since our data set is not uniformly distributed among all classes as can be observed in Fig. 1, assessing the performance of the model on the basis of accuracy may not be a good idea [14]. The composite average of the recall and precision corresponds to the $F1$-score. It's an indicator of the accuracy of the classifier. Equation (14) below refers to the calculation of $F1$-score, where σ and γ refer to the precision and recall respectively.

V. CONCLUSIONS

Three well-known deep learning models implemented for twitter sentiment analysis and preprocessed data to remove noise from data and increase the accuracy of models. In addition, we compared these models and evaluated their performance on the test data and found that the BERT model outperforms the rest. It can be inferred that BERT's superior performance can primarily be attributed to a pre-trained Wikipedia language model and a book corpus thatoffers a clearer understanding of the English language andthus performs better than the other two classifiers.

REFERENCES

[1] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," *arXiv preprint arXiv:1606.01781*, 2016.

[2] S. M. Mohammad, S. Kiritchenko, and X. Zhu, "Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets," *arXiv preprint arXiv:1308.6242*, 2013.

[3] J. Barnes, R. Klinger, and S. S. i. Walde, "Assessing state-of-the-art sentiment models on state-of-the-art sentiment datasets," *arXiv preprint arXiv:1709.04219*, 2017.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advancesin neural information processing systems*, pp. 5998–6008, 2017.

[6] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cis- tac, T. Rault, R. Louf, M. Funtowicz, *et al.*, "Transformers: State-of- the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[8] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, "Attention- based bidirectional long short-term memory networks for relation clas- sification," in *Proceedings of the 54th annual meeting of the associationfor computational linguistics (volume 2: Short papers)*, pp. 207–212, 2016.

[9] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural informationprocessing systems*, pp. 649–657, 2015.

[10] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[11] H. Kim and Y.-S. Jeong, "Sentiment classification using convolutional neural networks," *Applied Sciences*, vol. 9, no. 11, p. 2347, 2019.

[12] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural networks for perception*, pp. 65–93, Elsevier, 1992.

[13] P. Nakov, A. Ritter, S. Rosenthal, F. Sebastiani, and V. Stoyanov,"Semeval-2016 task 4: Sentiment analysis in twitter," *arXiv preprint arXiv:1912.01973*, 2019.

[14] M. Bouazizi and T. Ohtsuki, "Multi-class sentiment analysis on twit-ter: Classification performance and challenges," *Big Data Mining and Analytics*, vol. 2, no. 3, pp. 181–194, 2019.

[15] Classification?," in *China National Conference on Chinese Computa- tional Linguistics*, pp. 194–206, Springer, 2019.

[16] Q. Zhou and H. Wu, "Nlp at iest 2018: Bilstm-attention and lstm-attention via soft voting in emotion classification," in *Proceedings of the9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp.189–194, 2018