Emotion Detection From Textual Expressions

Ali Baran Tasdemir
Hacettepe University
Computer Engineering
alibaran@tasdemir.us

Akif Cavdar
Hacettepe University
Computer Engineering
akifcavdar@gmail.com

Abstract

Nowadays, social media usage is quite common. People can easily write all kinds of emotion in these media. They want to show their feelings to other people about a certain subject. In this case, people can transfer their feelings to other people through certain words or phrases within the sentence they have established. Our goal is to understand the sense of a sentence written by other people, using certain algorithms is possible to do on computers.

Keywords: Emotion detection, ISEAR dataset, LSTM, Neural Networks, NLTK, Natural Language Processing

1. Introduction

This project aim is to develop an algorithm that can be used to detect the emotions of people with the state-of-art algorithms and methods shared in papers.

Emotion detection from textual data is, sentences were written by people with specific words, or with specific patterns is to detect peoples feelings. The frequency and the words used in the sentence are important. It may be necessary to look at certain words to detect emotion. Because people can put a lot of meaning into some words or words. Here is the right method to perform the analysis and to make accurate estimates.

Emotions are the relationship between humans. Humans can express their feelings and care about their feelings. So emotions have a big effect on everyone's life. It affects directly to our behaviors. If we can make computers, emotionally sensitive we can increase the effect of the computers slightly. Thats why its a very exciting work that can maximize human-computer interaction. For example, with that understanding, we can make personal assistants more humanized. The biggest effects can be seen by robots. With that emotional sensitivity, communications of human-robot will be more accurate and perhaps make the human-robot separation difficult.

2. Related Works

There are some categorizations for emotions proposed for different psychologists. But the most suitable model for our dataset and to us was Ekman et al.[6] model. The emotions are primarily in 6 categories: Anger, disgust, fear, joy, sadness, surprise.

The number of papers in emotion detection field is very high. However, our subject "Emotion detection from textual data" is quite limited. So we find some researches in this area. There are some research groups using machine learning techniques[5], rule-based approaches[9] and deep learning methods[7].

In the research uses a rule-based approach[9], they find some dominant words in the sentence and used these sentences to make a prediction. They removing stop-words. And building a tree which will be used in the prediction algorithm.

In the research uses machine learning techniques[5], they are using Multinomial Naive Bayes algorithm to make a prediction. They are using stemming, stop-word removing, N-gram featuring. The most interesting part, they are using some punctuations in the data set. In the research uses deep learning techniques[7], they are using word embeddings models like Glove[8], Word2Vec, SSWE, and LSTM models.

3. The Approach

This section is about description of proposed system for emotion detection from textual data. Used data set is ISEAR data set[1]. Preprocessing and prediction made as in 1.

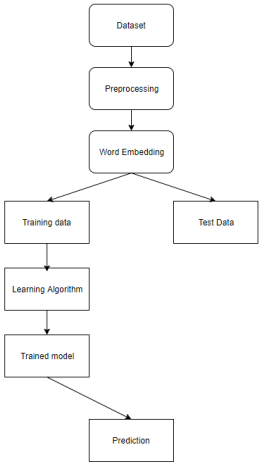

Figure 1. Scheme of system.

3.1. Data set

ISEAR data set used in this project. This data set, collected for a project named ISEAR and directed by Klaus R. Scherer and Harald Wallbott. Respondents had chosen from a group of students both psychologists and non-psychologists. And they were asked to report the situations they had experienced all of 7 major emotions. These emotions are joy, fear, anger, sadness, disgust, shame and guilt. In each case, the questions were about how they reacted to the situation and how they had appraised. The final data set contained reports on 7 emotions which answered by over 3000 respondents in 37 countries.

Emotion	Sample
Anger	1070
Disgust	1058
Fear	1079
Guilt	1070
Joy	1065
Sadness	1080
Shame	1081
Total	**7503**

Table 1. Data set distribution

[1]https://www.unige.ch/cisa/research/materials-and-online-research/research-material/

3.2. Pre-process

The ISEAR data set contains some sentences and for each sentence, there is an emotion labeling. So for this project, we did not interested in other features like country, gender, religion, etc.. We only used the situation(sentence) and emotion label. We perform a basic preprocessing for text data.

After that, we cleared shortened expressions and replaced them with full words. For example, if there is a word ”What’s” we changed this word to ”what is”. The dataset contains some answers from subjects like ”[No response]”. So to deal with that noisy data we cleared all sentences which contains ”[”.

We used N-gram features to use for some algorithms like Naive Bayes or SVM. For these purposes, preprocessed text tokenized. For some cases, we used Stemming or Lemmatizing to make tokenization better. These techniques, to eliminate some words presence in different forms.

3.3. Lemmatizing

Lemmatization returns the lemma for a given word. It gives the dictionary form of a word. In some ways, it can be considered an advanced form of a stemmer. It can also be used for similar purposes. It can ensure that all different forms of a word are correctly linked to the same concepts.

For instance, it can transform all instances of cats in cat, for search purposes. However, it can also distinguish between the cases of run as in the verb to run and run as in the noun synonym of a jog.

3.4. Stemming

Stemming is the technique for finding the roots of words. But the resulting root of this technique is not the same root as we see in the dictionary. For example, for the word ”flies” the stemming produces ”fli” as root. So this technique cuts off the ending character from word which makes this technique more computational wise cheaper.

3.5. Stop-words

We removed all stop-words in data. Because the words in stop-words list has no effect on the prediction. For example, we removed words like ”the”, ”and”, ”for”, ”to”.

3.6. Model

Before any deep learning model we have some works to do with textual data. In deep learning layers we will make some mathematical operations. Which can not be done with character. So we will use a method named word embedding. Word embedding is representation of words in n-dimensional vector space. There are two famous word representation model. One of them is GloVe and the other one is Word2Vec. We will use GloVe for our model.

Our word vectors will be 100 dimensional GloVe vectors. But in our method we will also train this vectors too. Because this vectors should be corralated with their emotional meainings. But this trained vectors will be a good initialization for our word vectors. Our proposed model is one of the deep learning methods Bidirectional LSTM. LSTM -Long Short Term Memory- is a module which contains gates responsible for some operations. It is a model that extends the memory of Recurrent Neural Networks. It reduces the vanishing gradient problem, which is where the neural network stops learning because the updates to the various weights within a given neural network become smaller. It does this by using a series of gates.

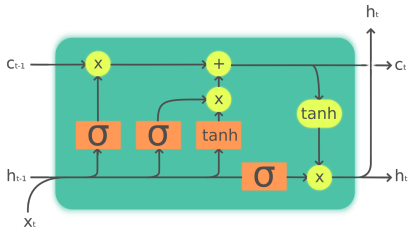

Figure 2. Inside of an LSTM module.

There are three types of gates within a unit:
Input Gate: Scales input to cell (write)
Output Gate: Scales output to cell (read)
Forget Gate: Scales old cell value (reset)

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$
$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$
$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$
$$h_t = o_t \circ \sigma_h(c_t)$$

f_t Denotes forget gates activation vector.
i_t Denotes input gates activation vector.
o_t Denotes output gates activation vector.

LSTM modules works feed-forward. But bidirectional LSTM modules works for both forwards and backwards. So for textual analysis, bidirectional LSTM modules fits very well.

Because of that, we used LSTM module in our neural network. Our model structure contains 3 layers. One embedding layer, one bidirectional LSTM layer and a Dense layer with 'Softmax' activation (3.6). Softmax activation is one of the best function for multiclass predictions.

$$f_x = \sum_{i=1}^{N} \frac{\exp x_i}{\sum_{j=0}^{N} \exp x_j}$$

So our neural network model:

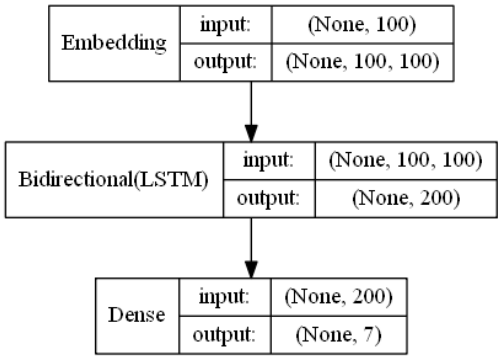

Figure 3. Neural Network Model.

4. Experimental Results

4.1. Experimental Setup

ISEAR data set used for all experimental results. First of all removed all unnecessary features and N/A lines in dataset with Python's pandas framework. Stop-words, taken from Python NLTK[2], removed from data. And all of the punctuations removed. We used Python NLTK framework's SnowBall Stemmer[3] and WordNetLemmatizer[4] to stem and lemmatize words. After the cleaning completed we tokenized words to build sequential data. Tokenizing done by Python Keras Tokenizer framework[1]. With Keras framework we convert tokenized sentences to sequences. To pass our input to neural network structure we need vectors with fixed sizes. Because sentence sizes are not convenient. So we add padding to sequences with Keras framework.

We split the dataset to train and test with ratio 80:20. So after that, we can measure all the things we need.

First of all we use this inputs to select good baseline machine learning or deep learning model.

4.2. Base Model Selection

In this section we will mention our results from different algorithms. Firstly, we used Multinomial Naive Bayes algorithm. Because we know Naive Bayes algorithm works well with textual data. After cross-validation tests the found optimal hyper-parameters for MNB. We used Laplace smoothing, bi-gram feature, Tf-Idf transformation, Stemming, and MNB alpha=0.01.

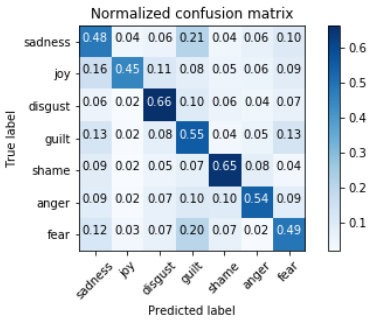

Figure 4. MNB confusion matrix.

As we see in Figure 4 class predictions are consistent. Almost for every class MNB predicted more than 50%. But with 54.47% overall prediction accuracy is quite low for our goals. And MNB failed to predict more than 50% for Sadness, Fear and Joy classes.

After MNB, we tried Logistic Regression. After cross-validation tests the hyper-parameter we found works best was;

L2 Regularization(Ridge Regression), c value 1.0, Lemmatizing and uni-gram feature.

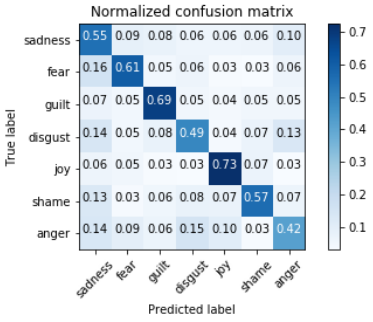

Figure 5. Logistic Regression confusion matrix.

As we see in Figure 5 class predictions are consistent too as MNB. This results slightly better than MNB. But still there is some problems in some classes. For example, Disgust and Anger classes predicted under 50%. But with 58.0% overall accuracy score, results are promising. Logistic regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a sigmoid function(4.2). So it works good with our data. The results was not unexpected.

$$f(x) = \frac{1}{1 + e^{-x}}$$

After Logistic Regression, we implemented our last machine learning algorithm Support Vector Machine. We wanted to try SVM. Because SVM are used commonly for classification problems. Again after some cross-validation tests we found that optimum hyper-parameters.

L2 Regularization, Bi-gram feature, Lemmatizing, Squared-Hinge function as error function, c=1.0.

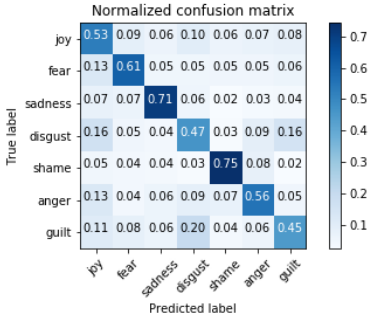

Figure 6. SVM confusion matrix.

As we see in Figure 6 class predictions are very similar to Logistic Regression results in Figure 5. But the problems still remains with SVM. There are some well predicted classes but still we have 2 class under 50% prediction accuracy. And SVM's overall prediction score 58.29%.

After these machine learning methods we experimented some deep learning methods. First of all we implemented CNN for our textual data. CNN usually works with images. But with word embedding we mentioned earlier we transformed our textual data to numerical image-like arrays. And we applied some hidden layers to get prediction. You can see CNN model Figure 7.

In our model we used one Dropout layer, because main issue with neural networks is overfitting problem. This dropout layer helps with overfitting issue. Convolutional Layer is 5x5x64 'relu' filter. 'relu' (rectified linear unit) is a function that, returns the max value (4.2).

$$f(x) = max(0, x)$$

And as fourth layer we have a max-pooling layer. This layer chooses the max value in 4x4 area. Fifth layer is LSTM layer. The details of LSTM layer is in (3.6). And the final layer is a Dense layer with 'Softmax' (3.6) activation.

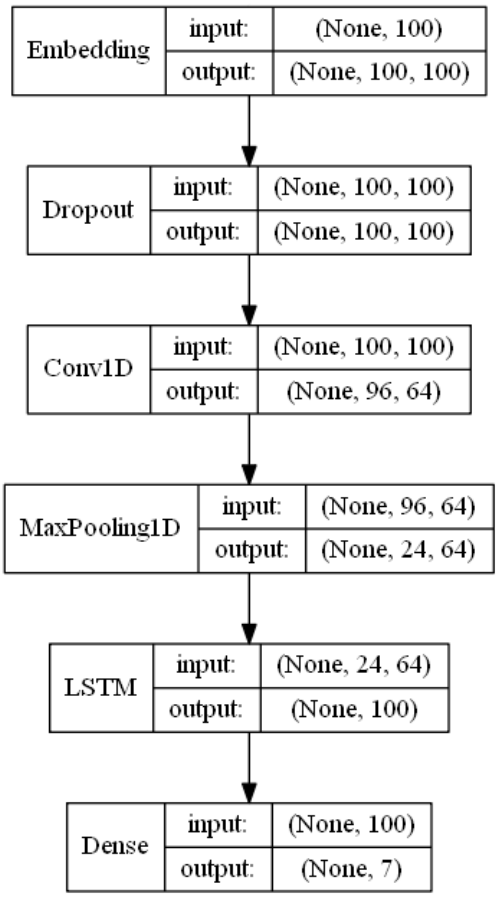

Figure 7. CNN layers scheme.

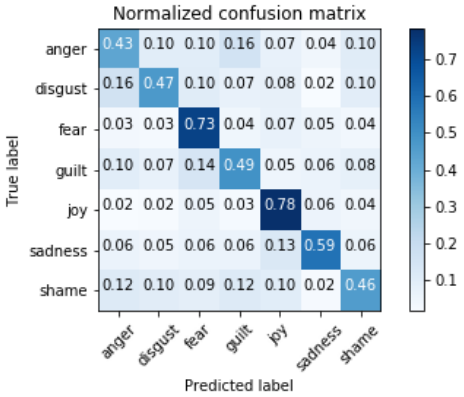

Figure 8. CNN confusion matrix.

The results of CNN model was not good like SVM or Logistic Regression. Biggest issue is overfitting problem. To prevent overfitting we used early stopping epochs. Because of that our data is not trained well. So the result are not too bad but SVM and Logistic Regression is much accurate models than CNN.

The last method we used is Bidirectional LSTM. We choose this method as final method. We already know Recurrent Neural Networks works very well with our data. We added LSTM module to RNN structure. Our model is Figure 3.

The results of Bidirectional LSTM model is much more better than other models. With 75.48% overall prediction accuracy takes the first place. All classes has prediction accuracy over 50%.

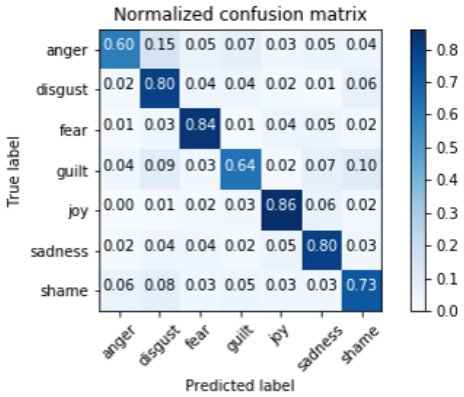

Figure 9. Bidirectional LSTM confusion matrix.

Algorithm	Accuracy (%)
MNB	54.47
Logistic Regression	58.0
SVM	**58.29**
CNN	56.42
B-LSTM	**75.48**

Table 2. All algorithms we implemented with accuracy %

5. Conclusions

This paper describes both machine learning and deep learning approaches for emotion detection from textual expressions. We used some natural language processing tools to getting numerical features from textual data. We did not interested with semantic characteristics of words. We tried to retrain GloVe word vectors with emotional meanings. Our proposed model is very successful on ISEAR data set.

As future work, we can test and modify our model to handle other data sets. And some problems we faced like noisy data, the expression contains irony or humour, can be solvable with more data analyzing.

References

[1] Keras framework. `https://keras.io`.

[2] Nltk toolkit. `http://www.nltk.org`.

[3] Snowball stemmer. `https://snowballstem.org`.

[4] Wordnetlemmatizer. `https://www.nltk.org/_modules/nltk/stem/wordnet.html`.

[5] D. K. A. Bincy Thomas, Vinod P. Multiclass emotion extraction from sentences. *International Journal of Scientific Engineering Research*, 5, 2014.

[6] F. W. V. . E. Ekman, P. *Emotion in the human face*. New York: Cambridge University Press, 1 edition, 1982.

[7] U. Gupta, A. Chatterjee, R. Srikanth, and P. Agrawal. A sentiment-and-semantics-based approach for emotion detection in textual conversations. *CoRR*, abs/1707.06996, 2017.

[8] R. S. J. Pennington and C. D. Manning. Glove: Global vectors for word representation. *EMNLP*, 14:1532–1543, 2014.

[9] C. F. Mohammed Abdel Razek. Text-based intelligent learning emotion system. *Journal of Intelligent Learning Systems and Applications*, 9, 2017.