Text Mining

2021/2022

Opinion Mining Evaluation

BlackBerry

Angie Kief	**20211913**
Francisco Ornelas	**20210660**
Luís Santos	**20210694**

1. Objective

The goal of this project is to tackle the challenge of emotion detection. For this purpose, given an input sentence, we were tasked with the goal of labeling it according to a set of 8 different emotions: 1: Anger; 2: Anticipation; 3: Disgust; 4: Fear; 5: Joy; 6: Sadness; 7: Surprise; 8: Trust.

2. Our Best Approach

Our best results were obtained with the use of a Gated Recurrent Unit (GRU) neural network and a pretrained Word2Vec (W2V) embedding technique. Our approach was structured in the following way:

Embedding Layer - Instead of trying to train our own W2V model, which would not have granted us satisfactory results due to the lack of text in our data, we used 300-dimensional word vectors pre-trained on Wikipedia articles.

Deep Network - Uses the sequence of embedding vectors as input and coverts them to a compressed representation. It then captures all the information in the sequence of words in the text.

Dense Layer – Takes the deep representation from the GRU and transforms it into the final output classes.

Output Layer – Has a total of 8 neurons, each for each emotion and uses a SoftMax function to solve our multiclass classification output.

2.1. Preprocessing

For our preprocessing we first cleaned the data, dealing with any type of contraction using the contraction library and then removing anything that was not text. We then proceeded to remove stopwords as well as unwanted spaces left in our data with the use of the RegEx and Natural Language Toolkit (nltk) libraries.

To further prepare the input data for our Neural Network Model, tokenized our text and counted the unique tokens using the tokenize package in the nltk library.

Lastly, we added padding, so that each input would have the same length (the longest input in our dataset being ~60 words), then categorized and converted each label to integers.

2.2. Pretrained Word Vectors

We downloaded and imported the 300-dimentional pretrained W2V from: https://fasttext.cc/docs/en/english-vectors.html, and proceeded to map each word in our corpus to an existing word vector, discarding any words not found in the pre-trained word vectors, which would most likely be spelling errors.

2.3. GRU layer and Output level

The GRU layer will receive the word embeddings for each token in each piece of text as inputs. With the use of a bidirectional wrapper, we will be in fact training two GRUs, one taking the input in a forward direction and the other in a backwards direction. Finally, in the output level we will be using a SoftMax activation function for the 8 final neurons.

2.4. Evaluation

We evaluated our approach with commonly used intrinsic metrics in NLP systems such as: Accuracy, Precision, Recall and led our evaluation with the F1 score, which is a combination of the prior two and evaluated to around 42% as can be seen bellow.

The scores for this approach also represented a significant improvement when compared to the scores of our baseline model, which consisted in the implementation of the same preprocessing techniques followed by using a Term frequency-inverse document frequency (TF-IDF) as a text vectorizer, that fed a created vector to a linear Support Vector Machine (SVM) model, achieving no score above 33%.

2.5. Results

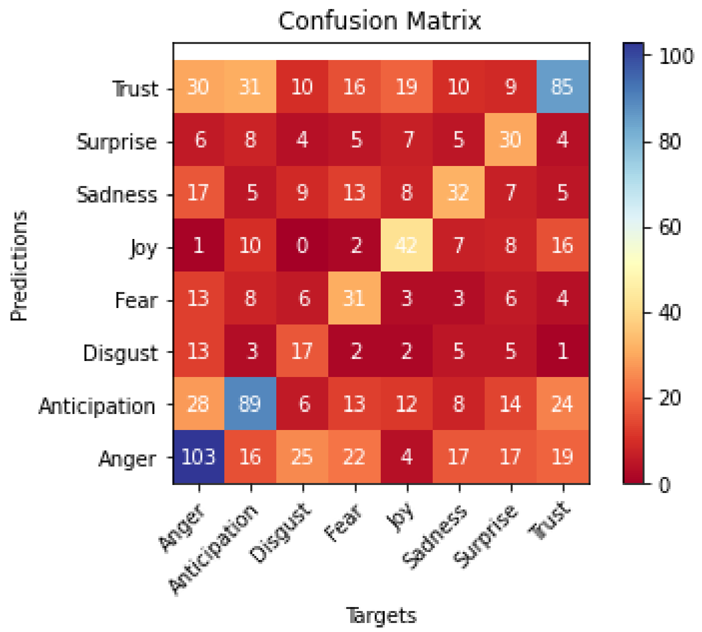
The results for the Precision, Recall, F1-Score and Accuracy in both our best approach (left) and baseline (right) were the following:

	Precision	Recall	F1-Score	Support
Anger	0.49	0.46	0.47	223
Anticipation	0.52	0.46	0.49	194
Disgust	0.22	0.35	0.27	48
Fear	0.30	0.42	0.35	74
Joy	0.43	0.49	0.46	86
Sadness	0.37	0.33	0.35	96
Surprise	0.31	0.43	0.36	69
Trust	0.54	0.40	0.46	210
Accuracy			0.43	1000
Macro Avg	0.40	0.42	0.40	1000
Weighted Avg	0.45	0.43	0.43	1000

Best approach: Preprocessing, W2V, GRUT

	Precision	Recall	F1-Score	Support
Anger	0.47	0.35	0.40	291
Anticipation	0.36	0.34	0.35	181
Disgust	0.16	0.16	0.16	75
Fear	0.31	0.30	0.30	106
Joy	0.39	0.31	0.34	124
Sadness	0.25	0.30	0.28	73
Surprise	0.16	0.29	0.20	52
Trust	0.26	0.38	0.31	108
Accuracy			0.32	1000
Macro Avg	0.29	0.30	0.29	1000
Weighted Avg	0.34	0.32	0.33	1000

Baseline: Preprocessing, TF-IDF, SVC



It's evident that the implementation of a neural network help significantly in the labeling task, improving as much as 11% in the F1-Score.

It's also apparent from the confusion matrix on the left and the displayed scores above that our model best labels anticipation and anger in text, having a much harder time capturing and differentiating disgust.

2.6. Future work

In order to improve the performance of our proposed approach, a more complex LSTM could be used instead of a GRU.

In addition, we could experiment with different Transformer language models such as BERT, RoBERTa or XLNet. In 2020 Adoma et al. published a comparative analysis on different pre-trained transformer models in the field of emotion detection with RoBERTa achieving the highest accuracy score. A similar approach could be used for our task.

3. Bibliography

Lukas Garbas, Dec 2019, 'Emotion Classification in Short Messages', https://github.com/lukasgarbas/nlp-text-emotion.

Michael Phi, Sep 2018, 'Illustrated Guide to LSTM's and GRU's', https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21

Simeon Kostadinov, Dec 2017, 'Understanding GRU Networks', https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

Vatsal, Jul 2021, 'Word2Vec Explained', https://towardsdatascience.com/word2vec-explained-49c52b4ccb71

Kurtis Pykes, Apr 2021, 'The Most Common Evaluation Metrics', https://towardsdatascience.com/the-most-common-evaluation-metrics-in-nlp-ced6a763ac8b

N.-M. H. W. C. Acheampong Francisca Adoma, Comparative Analyses of BERT, RoBERTa, DistilBERT, and XLNet for Text-Based Emotion Detection," in *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2020.

Python Libraries used: Natural Language Toolkit; RegEx; SkLearn; Keras; Pandas; Numpy; MatPlotLib;