Text-Based Emotion Classification

Xiaohan (Aria) Wang

Abstract

This project aims to detect the emotions behind text expressions. Similar to sentiment analysis, emotion analysis recognizes different types of feelings through the expression of texts. In this project, the author implemented 4 different text classification methods (Logistic Regression, Support Vector Machine, Fasttext and Long Short Term Memory models) and compared their performances in correctly recognizing 6 emotions (sadness, surprise, anger, fear, joy and love) from the text. All source code can be found in this GitHub page.

1 Introduction

Articulating emotions through words is one of the most important ways for people to interact with others. By writing texts, people may deliberately or unconsciously express their emotions. Therefore, text-based emotion recognition plays a crucial role in Natural Language Processing (NLP) because it requires the machine to not only capture the explicit meanings behind words, but also understand the linguistic expressions that subtly convey emotions. Emotion detection in texts also has wide applications in industries. For example, it would help companies to have a better understanding of their reputations by learning about people's emotions towards their products on social media, such as on Facebook and Twitter; implementing emotion detection tools can also facilitate the human-machine interactions, such as Chatbots tools. Similar to sentiment analysis, in which machines are trained to recognize the positive, neutral or negative attitudes from texts, emotion analysis aims to detect different human feelings, such as happiness, fear and sadness, from texts. In this project, the author implemented several different algorithms to classify the human emotions from texts.

2 Related Work

Compared to emotion recognition in speech and facial, text-based emotion detection has less sufficient work (Shivhare and Khethawat, 2012). The challenges in classifying emotions in texts into one of the categories are that 1) as languages evolve, there are constantly new ways for people to express their feelings, and 2) there is no universal standard for people to express certain feelings.

Shivhare and Khethawat (2012) proposed using a combination of keyword-based and learning-based methods to approach the text-based emotion recognition problem. With predefined keywords that are already classified into a category (angry, sad, etc.), the algorithm aims to identify key tokens that are associated with an emotion category, and the algorithm also assigns probabilities of other words belonging to a certain emotion class. Then the algorithm learns to classify the corpus into an emotion category with the captured signals. However, this algorithm has some disadvantages, in that it only looks at the corpus-specific meanings of words, and it fails to take into account the information on a higher level than the word level.

Chatterjee et al. (2019) improved the issue of the lack of contextual information in text-based emotion recognition by using the Bi-directional Long Short Term Memory networks (LSTM). They demonstrated the performances of different models, including BERT, ELMo, and they concluded that the LSTM model has the best performance in capturing the context and thus the underlying emotions within the corpus.

Based on these previous works, this project trained 4 different popular text classification models, Logistic Regression, Support Vector

Machine (SVM), Fasttext and LSTM, and the results are discussed in the following sections.

3 Dataset

The data was initially collected by Saravia et al. (2018) as an emotion dataset through noisy labels, annotated via distant supervision by Go et al. (2009), and then the data was posted on Kaggle by @Praveen. The open-source data can be downloaded here.

The dataset in the txt format contains a list of documents with emotion flags. Each line of the data represents a text corpus followed by its emotion label (example line*: i feel like I am still looking at a blank canvas blank pieces of paper;sadness*). The dataset is already normalized and split into train, test and validation for building the machine learning model. As is previously mentioned in this paper, this is a multi-class text classification problem. There are in total 6 classes in the outcome, sadness, anger, love, surprise, fear and joy, and the distribution is relatively balanced: out of 20,000 text corpora, 5,797 of them are labeled as sadness, 2,709 are labeled as anger, 1,641 are labeled as love, 719 are labeled as surprise, 2,373 are labeled as surprise, and 6,761 are labeled as joy.

Before training any algorithm, the author cleaned the data by removing numbers, symbols and stop words from the corpora. The clean data has corpora with an average word length 62. Then 80% of the full data was split into training set and the rest 20% was split into test set to evaluate model performances.

4 Method

In order to solve this multi-class classification problem in text analytics, the author implemented feature extraction using TF-IDF and converted each text corpora into a 250 entries' long vector. Both 1 and 2 gram in bag-of-words were used in this process to capture a more comprehensive context. Then the author trained 4 classifiers (Logistic Regression, Support Vector Machine, Fasttext and Long Short-Term Memory models) with hyperparameter tuning process to select the model with the highest classification accuracy.

4.1 Logistic Regression

Logistic regression model was selected as the benchmark for this project. Since this is a multi-class classification problem, the `OneVsRestClassifier()` function was used to accommodate for this specific problem. The author tuned the most important parameters, penalty and C (inverse of regularization strength) for the Logistic Regression model and compared their prediction results.

Hyperparameters	**F1 score**	**Accuracy**
penalty = ‘l1’, C = 15	85.2%	88.3%
penalty = ‘l2’, C = 15	84.3%	88.0%
penalty = ‘l2’, C = 30	86.1%	90.5%

Table 1: Prediction Results for Logistic Regression

As Table 1 shows, the parameter set that has the best performance is L2 regularization and C equal to 30. Compared to other 2 sets of hyperparameters, this set implement less rigorous penalty and favors a relatively more complicated model.

4.2 Support Vector Machine (SVM)

SVM with linear kernel is another frequently used benchmark algorithm in text classifications, in that it manages to find the hyperplane that to the largest degree separates support vectors in different classes, which results in good performance when dealing with large corpus data. The author compared the model performances with different loss functions and values of C, and below are the prediction results.

Hyperparameters	**F1 score**	**Accuracy**
C = 1, loss = ‘squared_hinge’	87.2%	91.5%
C = 50, loss = ‘squared_hinge’	88.5%	91.6%
C = 1, loss = ‘hinge’	86.6%	91.4%
C = 50, loss = ‘hinge’	87.2%	91.3%

Table 2: Prediction Results for SVM

Table 2 indicates that with other parameters set to the default values, the SVM model with C equal to 50 and squared hinge loss function has the best performance.

4.3 Fasttext

Fasttext is an algorithm that is frequently used for both supervised and unsupervised text analytics problems. It usually has comparable classification performance with more complicated deep learning algorithms, but Fasttext is far more efficient. When looking for the best set of parameters, the author kept the ngram parameter to be 2 and trained different values of lr (learning rate), ws (size of the context window) and the number of epochs.

Hyperparameters	F1 score	Accuracy
lr = 0.1, ws = 5, epochs = 100	89.5%	91.2%
lr = 0.2, ws = 10, epochs = 100	89.5%	91.3%
lr = 0.2, ws = 10, epochs = 300	89.3%	90.8%
lr = 0.2, ws = 15, epochs = 100	89.7%	91.3%

Table 3: Prediction Results for Fasttext

As is shown in Table 3, the best hyperparameter set has learning rate 0.2, context window size 10 and 100 epochs. This parameter set favors a model that takes into account a larger context window and larger learning rate.

4.4 LSTM

LSTM has proven its great performance in dealing with data in sequences in many researches. It captures the context in corpora data by taking the history as input, and it handles noise in the history by leveraging whether or not to temporarily "forget" the information. Thus, for this project, the author also developed a LSTM model with hyperparameter tuning to compare its performance with those of other models. The LSTM architecture used in this project is simple; there is one embedding layer, one LSTM layer with dropout layer and one SoftMax output layer. The loss function used in the back propagation is categorical cross entropy function. The author implemented the Adam optimizer and used accuracy as the main evaluation metric. The number of epochs and batch size were set to 5 and 64 respectively. The table below compares the performances of LSTM with different numbers of hidden nodes and learning rates.

Hyperparameters	F1 score	Accuracy
lr = 0.001, hidden nodes # = 100	87.1%	90.1%
lr = 0.001, hidden nodes # = 200	88.7%	91.4%
lr = 0.0005, hidden nodes # = 200	86.8%	89.1%

Table 4: Prediction Results for LSTM

As Table 4 indicates, the best parameter set has learning rate 0.001 and 200 hidden nodes. With the same number of hidden nodes but smaller learning rate, both the F1 score and the prediction accuracy drop, probably due to the model stuck at a local minimum.

5 Result

Based on the best performance for each algorithm with different sets of hyperparameters, below is a table that compares the classification performances across model.

Algorithm	F1 score	Accuracy
Logistic Regression	86.1%	90.5%
SVM	88.5%	91.6%
Fasttext	89.5%	91.3%
LSTM	86.8%	91.4%

Table 5: Best Performances for All Models

Among all algorithms talked about in this paper, SVM has the best performance in terms of both the F1 score and accuracy. Fasttext and LSTM have similar accuracy scores, but Fasttext has a better score in F1. In other words, Fasttext did better in balancing out accuracy in multiple emotion classes compared to LSTM. Logistic Regression also has a decent classification performance.

6 Discussion

The SVM, Fasttext and LSTM models all outperformed the benchmark Logistic Regression model. Among these 3 models, SVM has the highest F1 score and classification accuracy. As for training efficiency, both SVM and Fasttext are easy to train in a short time, while LSTM has more hyperparameters to tune and it takes the longest training time. With longer time and more computing power, further tuned LSTM with more layers would probably outperform other models in this project. Furthermore, it would be a good practice to try other models, such as BERT, ELMo and CNN, on this specific text-based emotion detection problem.

References

Ankush Chatterjee, Kedhar N. Narahari, Meghana Joshi, and Puneet Agrawal. 2019b. Semeval-2019 task 3: Emocontext: Contextual emotion detection in text. In *Proceedings of The 13th International Workshop on Semantic Evaluation (SemEval-2019).* Minneapolis, Minnesota.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12)

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, 2018.

Shiv N. Shivhare and Saritha Khethawat, "Emotion Detection from Text", *Second International Conference on Computer Science Engineering and Applications(CCSEA-2012)*, May 26-27, 2012, ISBN 978–1-921987-03-8.