Machine Learning Engineer Nanodegree

Capstone Project

Cheuk San Yip
July, 2019

Proposal

Sentiment Analysis on Conversational Data

Domain Background

One of the popular research topics in current natural language processing (NLP) is the sentiment analysis on texts corpora. It has become more popular due to widespread Internet usage and the texts freely available online on social media.

Sentiment analysis deals with using automatic analysis to find sentiments, emotions, opinions and attitudes from a written text towards a subject. Based on the words associated with negative, neutral or positive sentiments, the documents are classified into 3 sentiment categories.

However, challenges with conversational data concern their unconventional characteristics: they are short, highly situational and produced alternatingly by the participants[1].

To the best of my knowledge, a lot of existing open-source sentiment/emotion models were trained on data collected from social network and movie reviews. There are not many models that were trained on conversational data. Therefore, it became my intension to develop a model that is more suitable for analysing sentiment in a conversational context.

Problem Statement

The problem to solve is predicting the underlying sentiment when given textual features extracted from dialogues.

In essence, this is a multi-class classification problem. The 3 classes of sentiment – negative, neutral and positive, were derived from the polarities of human's basic emotions. The table below maps the relationship between sentiments and human emotions.

Emotion	**Sentiment**
Angry	Negative
Disgust	Negative
Fear	Negative
Joy	Positive
Neutral	Neutral
Sadness	Negative
Surprise	Positive / Negative

One of the assumptions is that for each utterance (i.e. short sentence) drawn from a conversation, it possesses **one** of these emotions of the speaker at the time it was said.

[1] Reference: *Sentiment analysis on conversational texts* (https://www.aclweb.org/anthology/W15-1829)

Through the use of natural language processing and text analysis, we should be able to develop models which have the ability to detect sentiment of the speakers from their speeches/dialogues.

Dataset

The Multimodal EmotionLines Dataset (MELD)[2] is the sole dataset used in this project. It contains about 13,000 utterances from 1,433 dialogues from the TV-series *Friends*. Each utterance is annotated with emotion and sentiment labels, and encompasses audio, visual, and textual modalities[3].

Compared with data from Twitter and IMDb, the MELD dataset is more suitable for my purpose given the source it has been collected from. Dialogues from *Friends* carry the characteristics that are more closely resembling our daily conversations in terms of the ways that emotions were expressed and the conversational word choices under different emotions. The utterances are short and constructed jointly by the participants in the dialogue context.

The dataset has been saved in 3 separated .csv files, namely Dev, Train and Test. Each file contains utterances in the respective set along with Sentiment and Emotion labels.

The table below outlines the label distribution.

Sentiment	**Dev**	**Train**	**Test**
Negative	406	2945	833
Neutral	470	4710	1256
Positive	233	2334	521

(**note*: despite that the raw data have already been separated into Dev, Train and Test sets, concatenating them together into one full set during pre-processing and splitting with a different ratio remains as an option)

Solution Statement

There is a variety of approaches to tackle sentiment analysis. The one I will take in this project is by leveraging statistical methods. This involves traditional supervised learning and deep learning. The performance of these models will be assessed and the best performing model will be determined at the end of the project.

The key Python packages chosen to assist with model development includes:

- Pandas for data pre-processing;
- scikit-learn(sklearn) for feature extraction and traditional supervised learning algorithms;
- Keras for constructing various neutral network architectures.

Benchmark Model

My benchmark model is a simplest model that will predict the same class for any given sentences/utterances. The class it will predict depends on the majority class seen in the training

[2] Github source: https://github.com/SenticNet/MELD1

[3] Reference: https://arxiv.org/pdf/1810.02508.pdf

dataset. Since I intend to conduct rebalancing to the dataset as part of pre-processing, the resulting sample size for each of the 3 classes will be relatively even, and the weights put on should be equal. Providing this, the expected accuracy for the benchmark model should be around 33%. Any models that can out-perform this baseline model from an accuracy perspective will be considered as 'good enough' models.

Evaluation Metrics

F1 score for Sklearn's models; the F1 score is calculated by the following formula:

F1 = 2 * (precision * recall) / (precision + recall)

F1 score, the weighted average of precision and recall, which takes both false positives and false negatives into account. It is generally considered as a better evaluation metric than accuracy.

However, for easy comparison with Keras' deep learning model, accuracy will also be considered as the evaluation metrics for Sklearn's models. Precisely this would be the test accuracy as opposed to train or validation accuracy.

Project Design

Step 1: Commence data cleansing after importing; investigate and understand the basic statistics of the dataset, for instance the distribution of labels.

Step 2: Pre-processing and feature extraction. In the case where dataset is highly imbalanced, re-balancing needs to be done. This can be achieved by either under-sampling or over-sampling. Sentences (texts) cannot be directly fed into machine learning model as features, hence utterances will then need to be tokenised with the appropriate tokeniser, such as applying CountVectorizer or TfidfVectorizer from Sklearn.

Step 3: Prepare a baseline model prior to building more complicated models, so as to benchmark the performance of these advanced models.

Step 4: Carry out cross-comparison on different models and evaluate their performance against the evaluation metrics set. Models proposed to investigate include:

- Support Vector Machine
- Random Forest Classifier
- Neutral network with multiple fully connected layers
- Convolutional neutral network
- Convolutional LSTM with transfer learning from Glove

The selection of hyper-parameters and neural network layers will form a crucial part of the model development, as these factors could be influential to the models' accuracy and generalisability.

- Propose to leverage GridSearch to assist with hyper-parameter tuning for Sklearn's models;
- Propose to consider different layers (e.g. Dropout, Pooling, Batch normalisation etc.), L1/L2 regularisation methods, optimisers in the deep learning;