Web Science Coursework - Social Media Emotion Data Set - Level M

Patrick Menlove - 2250066M

GitHub (code and data):
`https://github.com/pm990320/twitter-emotions`

February 22, 2020

1 Introduction

This assignment concerns gathering cleanly labelled tweets for different emotion classes from Twitter. The solution used employs the use of the Twitter API and Google's Word2Vec.

The code uses several libraries (all of which can be found in the `environment.yml` file), and uses MongoDB for data storage.

2 Data crawl & Rules

2.1 Twitter API

The code uses the Twitter Streaming API to collect data in real-time when run. To be specific, the `1.1/statuses/filter.json` endpoint (wrapped under the `TwitterAPI` library - `https://github.com/geduldig/TwitterAPI`).

The reason this was chosen was that it would allow the code to ingest a significant volume of tweets and not be limited by twitter's premium search API rate limits. It also gives real-time, current, relevant tweets on which to do analyses.

The API allows the `track` parameter to be specified, which can include various keywords to match. In the implementation, emotion label terms and emoticons are included in this parameter in order to filter results for each emotion.

2.2 Collected data

Here is a table showing the distribution of assumed classes before processing, and the time ranges from which the tweets were collected.

Emotion Label	**Count**	**Date Range**	**Date Range**
anger	34	2020-02-22 10:15:00	2020-02-22 10:26:52
excitement	7	2020-02-22 10:16:20	2020-02-22 10:26:49
fear	111	2020-02-22 10:14:54	2020-02-22 10:27:56
happy	471	2020-02-22 10:14:53	2020-02-22 10:27:52
pleasant	4	2020-02-22 10:15:47	2020-02-22 10:22:43
surprise	17	2020-02-22 10:15:44	2020-02-22 10:27:36

2.3 Processing of Tweets

2.3.1 Clean Data

In order to collect reasonably clean data, the code uses a series of search filters and transformations.

Initially, the `track` parameter of the API call is given all the key words, so the streamed tweets are what twitter believes to have relevance to the kerywords/hashtags that have been identified as synonymous with the emotion label.

Then, these are initially filtered using a naive approach, by checking for the presence of other emotion labels' keywords in the tweets, and discarding the tweet if there is a match. This, in the simplest form, gives some confidence in the labels being reasonably clean.

The tweets are then persisted in MongoDB, and the `process_tweets.py` file can be run, which makes a more accurate re-labelling of the tweets, by utilising Google's pre-trained Word2Vec. By computing the consine similarity of each "emotion" word with every word in the tweet, averaging and comparing the relative similarity scores for emotions, we can make a much better guess at the class of each tweet.

In addition to this, the `process_tweets.py` file removes "RT: @username", any @ mentions and any ellipsis ... from the tweets.

Stemming or lemmaisation was considered, but this would not preserve the tweets human-readable nature for the crowd-sourcing. Though, the raw tweet text could be supplied for crowdsourcing and the lemmaised version be stored for computational use.

2.3.2 Use of Emoticons

Emoticons are used in the Twitter API stage of processing - since emojis can be passed as unicode strings, and the API accepts emojis to search for in the `track` parameter.

Beyond this, emoticons are not taken into consideration in later processing steps. This is because the content of words similarity can be compared with Word2Vec but emoticons cannot.

3 Crowdsourcing Method

3.1 Crowdsourcing Details

3.2 Results and Discussion