Natural Language Processing

Sentiment Emojizer Phase 1

Gathering data

The source of data is twitter. When you run the gathering data script, program gets the most recent tweets from twitter.

To get the new data, we should run the bash script located in run folder named gather_raw_data.bash. You can add arguments to this bash to change default values. First arg is the number of tweets per class and you should enter it when running the script. Second argument is the output directory and you can specify that or leave it as the default value (data/raw). I ran this command to gather 8000 tweets per class:

```
bash run/gather_raw_data.bash 8000
```

After running this bash, you can see the results of getting tweets in console. Note that you should specify the classes details in data/classification_data/classes_data.csv, to get data correctly. Our program starts to get the emojis specified in the file above for each class and send a query to twitter api to get the tweets containing the given emoji. Then save the id and the text of the tweets to the output directory with tsv (tab separated values) format starting with the name of the class specified in the classes information csv. For more info check the “data/raw” folder in the project.

Preprocessing

To run preprocessing you should run the bash script located in run folder with name preprocess_raw_dta.bash. I ran the preprocessing with default arguments:

```
bash run/preprocess_raw_data.bash
```

You can simply run the script with default arguments. After running the script you can see the count of data before preprocessing and after preprocessing in console logs.

- “--ids'' the classes id you want to preprocess. Should be separated by ‘,’ without space. Default value is “1,2,3,4,5,6,7,8”.
- “--out” the output directory. Default value is PREPROCESSED_BASE_DIR in constants
- “--flags” preprocessing methods you want to run on raw data. Accepted values are characters ‘a’ to ‘l’. Note that after part ‘f’ I tokenized the tweets with TwitterTokenizer in NLTK module. I didn’t use sentence tokenizer, because I think this level of tokenizing is not needed in this case. Before removing emojis (part k) we should identify the label of the tweets.
    - ‘a’ : remove incomplete tweets (some tweets from twitter api are not complete and we should remove them)
    - ‘b’: remove retweets
    - ‘c’: remove mentions
    - ‘d’: remove hashtags
    - ‘e’: remove punctuations
    - ‘f’: replace newlines with empty string
    - ‘g’: stemmize tokens (this options is off by default)
    - ‘h’: remove the tweets that have more than 40% emojis in tokens.
    - ‘i’: remove emojis which are not defined for our system. Defined emojis are available in class_data.csv file.
    - ‘j’: remove consecutive repeating emojis.
    - ‘k’: remove emojis
    - ‘l’ : remove non ascii tokens
- “--input” input directory of raw data containing tsv files with name {class_name}_raw_text.tsv for each id available in ids argument.

Labeling

After preprocess data till part k, we should identify the labels. Each tweet can has multiple labels based on the emojis in the tweet. Note that in part i we removed undefined emojis. So this part and part j are important in the labeling process. After removing undefined and consecutive repeated emojis, we can count emojis used in tweet and identify its label. Then the probability that this tweet belongs to a label is the number of emojis associated with that label, divided by the total emojis count in the tweet.

After running the preprocess script, labels will be created and saved in jsons in the labels folder. Each tweet gets an index same as the file name in the labels folder. We can access the label of each tweet with the associated index.

Statistics

To show the statistics details you can run bash script and show the results.

```
bash run/show_statistics.bash
```

Arguments are:

- “--flags” type of statistics you want to show
- “--input” the input directory. Default value is PREPROCESSED_BASE_DIR in constants