Progress Report - CS 410 - FHYYY

yixin10(captain), yuxuan54, yiteng3, hj28, fengyiz3

Back-end part:

1) **Progress made thus far:**

   We have designed the backend part as **Tweets Content Retrieval By User ID** and **Label Calculation With Pre-trained Model**.

   - For the former part, we first decided to implement it by a crawler. However, Twitter's page only retrieves data when the scrolling down action is detected, which makes it harder to retrieve data with crawlers. Fortunately, after looking through Twitter's development documentation, we found there are some open-source APIs that allow us to retrieve users' recent tweets.
   - Currently, we've achieved a basic understanding of Django, the framework for our back-end application. We have completed a server-side Restful API that allows our front-end application to access users' sentimental tendencies by sending HTTP GET requests to our server.

2) **Remain tasks:**
   - Implementing retrieving tweets from Twitter and pre-process the raw JSON-format data before feeding them to the model.
   - Deploy our backend application on a PaaS provider(Not specified yet).
   - Apply user authentication on the backend so that our application can potentially support user-centered customization.

3) **Challenges & Issues:**
   - We decided to implement a more scalable back-end system to support user authentication, which, however, means that the development workload would be doubled. Also, we meet difficulty when finding suitable tools to do user authentication. More effort and work need to be done to achieve this goal.
   - Moreover, storing pre-trained models in the database is not a good idea since reading a GB size file from the database can be slow, bringing a negative user experience. Another approach should be proposed to figure this out.

Model part:

For dataset selection, we choose the Sentiment140 dataset with 1.6 million tweets from kaggle, which contains 1,600,000 tweets extracted using the twitter API. The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment . Thanks to Google Colab, we can have codes runned by free computing resources on the cloud. We cleaned the data first, including defining regex patterns, removing URL, username and emoji,

stemming and lemmatization as well as filtering stopwords. And then we employ the TF-IDF method to extract the features from the cleaned tweet set. Finally, we use a logistic regression method to construct a classification model and train our data where 5% of the tweets will be used for testing. Eventually, we got a model with average precision of 0.73 and a recall of 0.75. It performs well given several samples like “I love you” to give a positive prediction and “I hate you” for a negative prediction. However, the model can be tweaked so as to achieve a higher performance. The next step we would try to modify some super parameters like n-grams or feature dimensions, and explore other popular discriminative models like SVM or generative models like Bernoulli Naive Bayes and so on.

Front-end part:

1) **Progress made thus far:**
    - We have designed the frontend part with our backend team about what kind of parameters need to be transferred to backend and the way to call the backend api.
    - We have designed the module of frontend.
2) **Remain tasks:**
    - We need to implement the specific function about how to post our data to the back-end, and how to deal with the data from the back-end
    - Implementing user-authentication.
    - Designing clearer user interfaces.
3) **Challenges & Issues:**
    - Since it’s our first time to design and work on frontend. There is still some research that needs to be done.
    - Especially we need to implement user authentication. How to interact with backend about authentication is a challenge we need to work on.