ECE1786
Creative Applications of Natural Language Processing

Final Report
Sensify

Word Count: 1570

Ao Li

Haoran Liu

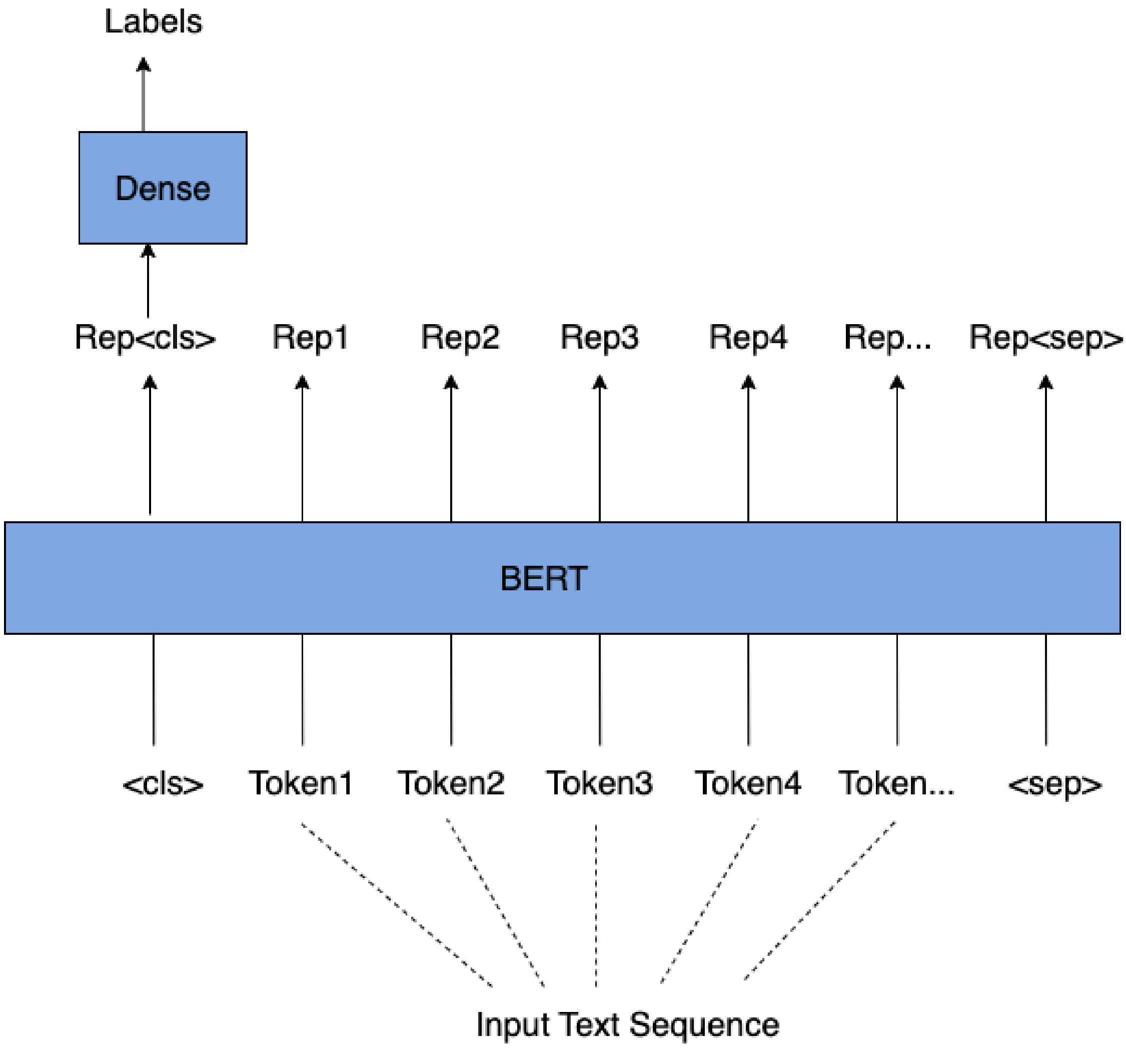
Introduction

Our project aims to build a multi-label sentiment classification model that measures the input on five emotion metrics: surprise, happiness, disgust, anger, and neutral. The model will take a sentence or a paragraph of words as input, and return the probabilities on five emotion metrics.

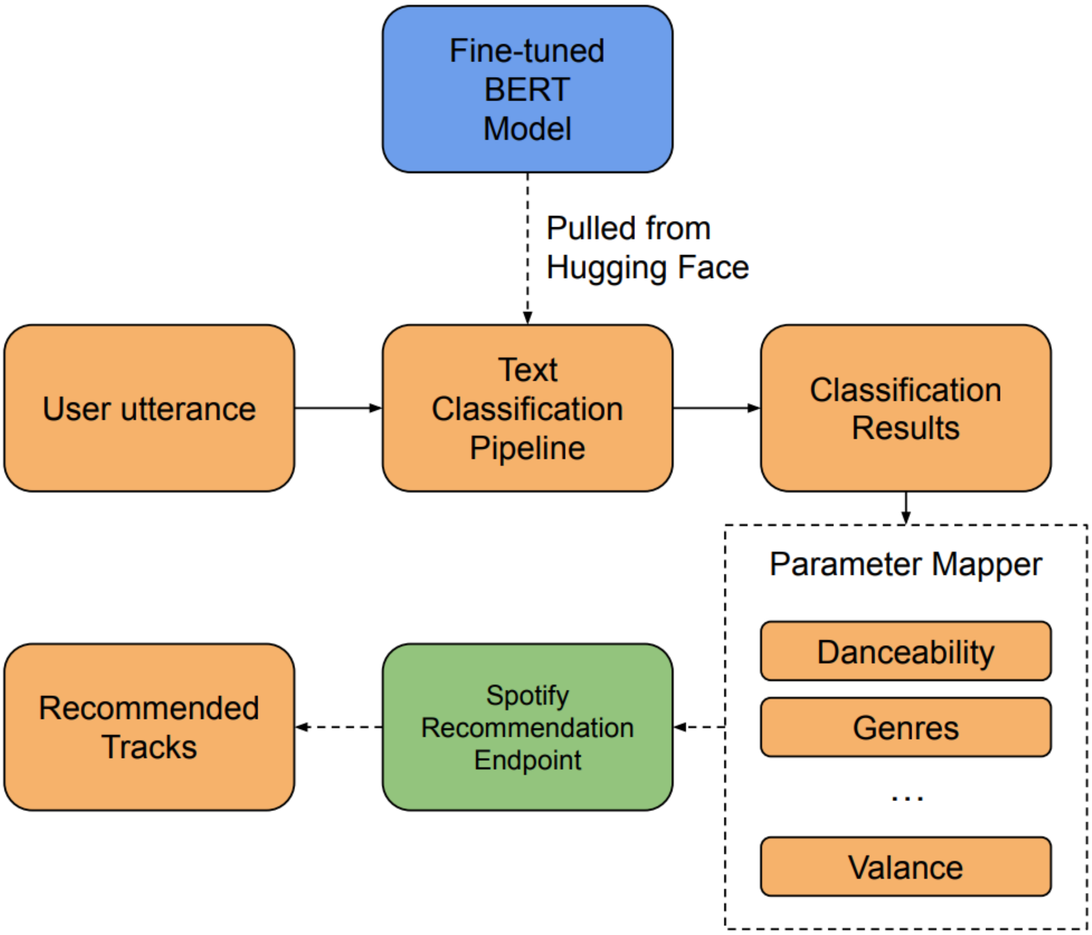
Our motivation behind the project is to use NLP models to assist chatbot products to detect technical or user experience issues by detecting negative sentiments expressed in user utterances when users are experiencing such issues. But due to the complexity of deploying and integrating with a chatbot to demonstrate the model, we decided to build Sensify, an application that can directly show the classification result measured from the utterance in the UI and recommend music using the Spotify recommendation API [2].

Illustration

Here is an illustration of the architecture of our classification model.



Here is an illustration of how the model is being used in Sensify.



Background & Related Work

Zhang, Lei & Wang, Shuai and Liu, Bing, “Deep Learning for Sentiment Analysis: A Survey,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2018

This paper [1] is considered to be one of the best survey papers in the field of sentiment analysis. It presents a variety of deep learning models that have been applied to sentiment analysis, such as CNN, auto-encoder, and RNN, among others. The paper also categorizes sentiment analysis into three levels: document-level, sentence-level, and aspect-level, and discusses the tradeoffs between these levels. The paper provides a comparison and summary of how different models can be used for different analysis tasks, and it motivates us to use a transformer model to compare with previous approaches.

Data and Data Processing

Our dataset consists of two parts:

- The first part comes from Go emotions(https://huggingface.co/datasets/go_emotions), which contains 58,000 Reddit comments labeled with 27 dimensions. We refactored the dataset to only have the 5 dimensions we need: surprise, happiness, disgust, anger, and neutral. And we filtered the dataset to only contain data entries with at least one of the five dimensions labeled as positive. After filtering the data, we exported the first 2000 data entries to a CSV file to use in our model.
- The second part comes from emotion(https://huggingface.co/datasets/emotion), a dataset of English Twitter messages labeled with six basic emotions: anger, fear, joy, love, sadness, and surprise. Some of our 5 target dimensions are labeled in the original emotion dataset, but some are missing. The dataset is also labeled for multi-class classification tasks, which means the classes are mutually exclusive. Therefore, we need to label the missing dimensions for our multi-label classification model. We removed the fear, love, and sadness dimensions, and mapped the original labels to our labels. We then exported the refactored dataset to a CSV file and started the manual labeling process in Excel. We decided to label the first 500 data entries to use in our model. Due to the subjectivity of labeling sentiment dimensions, both of our team members labeled the 500 samples. We discussed discrepancies and made revisions to ensure the most accurate representation of the data.

After cleaning and labeling the datasets, we first shuffled the dataset to make sure data from both sources are evenly distributed. To stratify the data during splitting, we found that the stratification algorithm for multi-label datasets differs from single-label datasets. Therefore we had to employ the iterative_train_test_split method from scikit-multilearn library to split the datasets.

In total, we have 2500 examples in our dataset. We are using a test size of 0.2, which means 2000 training examples and 500 validation examples. In our 2500 examples, there are 508 examples labeled with 'happiness', 173 examples labeled with 'surprise', 152 examples labeled with 'disgust', 347 examples labeled with 'anger', and 1392 examples labeled with 'neutral'.

Here are a few examples from our dataset:

text	happiness	surprise	disgust	anger	neutral
i feel like i should have actively hated every single second rather than just borne it all	0	0	1	1	0

i'm not feeling the outfit but the heels are gorgeous	1	1	0	0	0

Architecture and Software

Transformer Model

Our model is based on the pre-trained BERT model from the Hugging Face library. Specifically, we use the BertForSequenceClassification class, which contains a head designed for sequence classification/regression on top of the BERT model. The input is first tokenized and passed to the BERT model. The special token "<cls>" encodes the information of the entire input text sequence and is used for sequence classification. The representation of the input text is then fed into a small multi-layer perceptron (MLP) consisting of fully connected (dense) layers to output the distribution of all the discrete label values.

Sensify

Sensify is built with python Flask and Bootstrap, the application consists of 4 pages: login page, input page, result page, and listen page.

The login page handles authentication, which allows users to log in using their Spotify account. Sensify will use the temporary authorization header to access the Spotify recommendation and create playlist endpoints. Next, after the users are authenticated, they are navigated to the input page, where they can submit an utterance that expresses their feelings. The utterance is then fed into the text classification pipeline of our BERT model, which was uploaded to Hugging Face (https://huggingface.co/dinolii/ece1786), to be classified into 5 emotion metrics: surprise, happiness, disgust, anger, and neutral.

Next, the classification results are mapped to different parameters of the Spotify recommendation endpoint [2]. Happiness is mapped to the happy genre and high danceability. Surprise is mapped to low popularity, to surprise users with music they most likely haven't listened to. Disgust is mapped to the sad genre, and anger is mapped to low valance and high energy. Neutral is mapped to neutral valance and high popularity.

After mapping the parameters, Sensify passes them to the Spotify recommendation endpoint to randomly recommend at most 25 tracks that match the criteria. The recommended tracks are displayed on the result page, and if users are unsatisfied with the tracks, they can refresh the page to fetch a new list of recommended tracks. Users can save the tracks as a playlist to their Spotify account by giving it a name at the bottom of the result page.

After saving the tracks as a playlist, users are redirected to the listen page to test out their freshly generated playlist.

Baseline Model

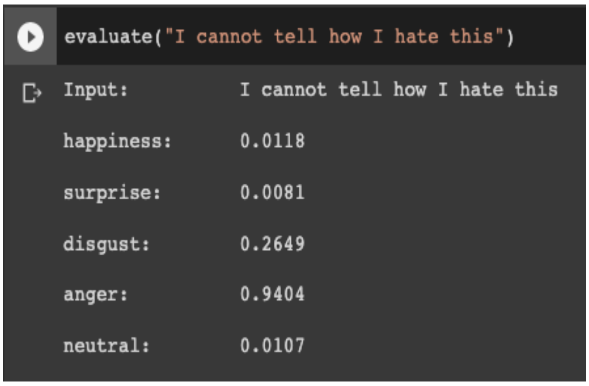
We decided to use a multi-label MLP classification model with one layer for our baseline model instead of multiple single-class MLP classification models that were initially proposed. It feeds the average of word embeddings in a sentence to a fully connected layer that produces scalar outputs with sigmoid activation.

Quantitative Results

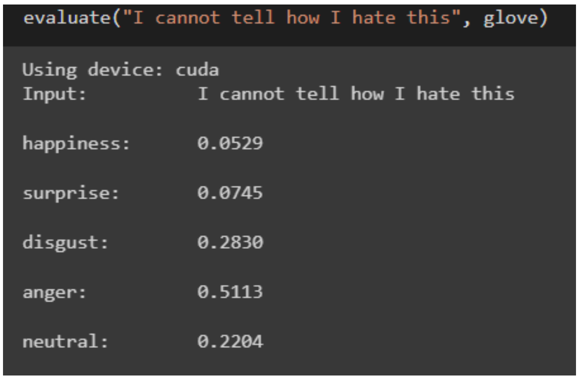
The metrics we use to evaluate our models are accuracy and the F1 score for each label. Accuracy is a simple and intuitive metric that provides a clear way to measure how well a model is performing. The F1 score, on the other hand, offers a more comprehensive view of the model's performance, particularly in terms of how it performs on each class of the problem. By using both of these metrics, we can see how the model performs in terms of overall accuracy as well as its performance on individual labels.

Qualitative Results

Result between BERT model and baseline model:



```
evaluate("I cannot tell how I hate this")

Input:          I cannot tell how I hate this

happiness:      0.0118

surprise:       0.0081

disgust:        0.2649

anger:          0.9404

neutral:        0.0107
```

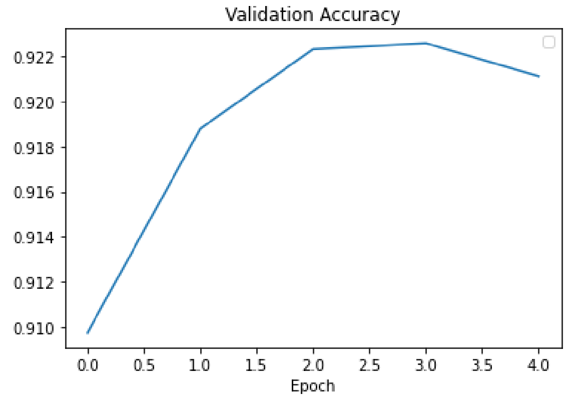


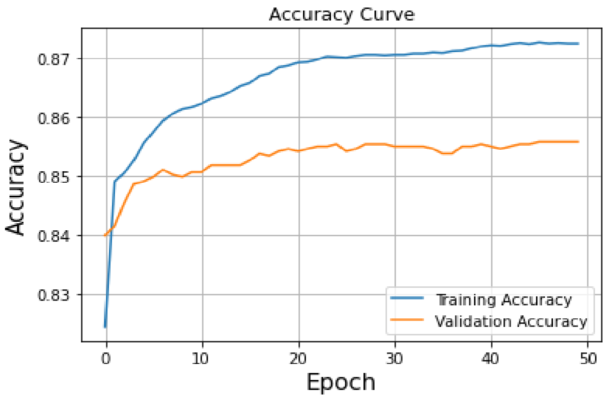
```
evaluate("I cannot tell how I hate this", glove)

Using device: cuda
Input:          I cannot tell how I hate this

happiness:      0.0529

surprise:       0.0745

disgust:        0.2830

anger:          0.5113

neutral:        0.2204
```

BERT Model

Baseline

Accuracy curves between the BERT model and baseline model:

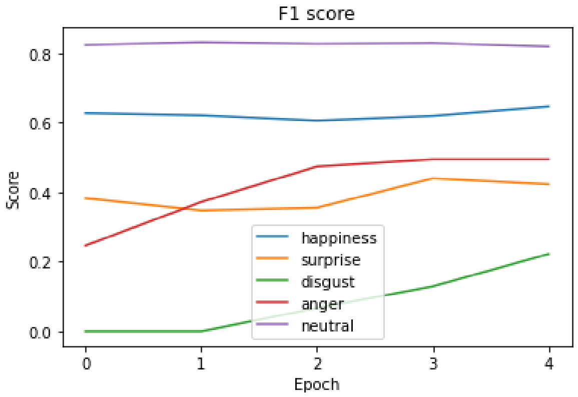


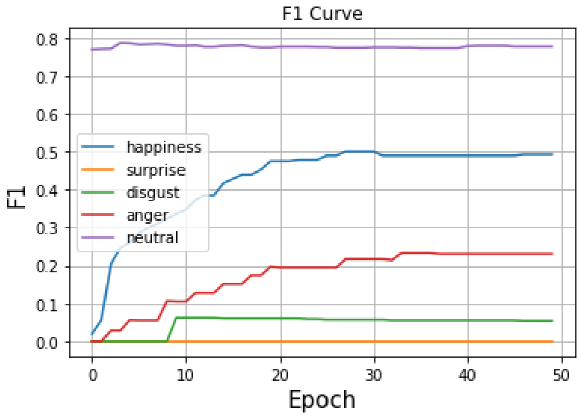


BERT Model

Baseline

F1 scores between the BERT model and baseline model:



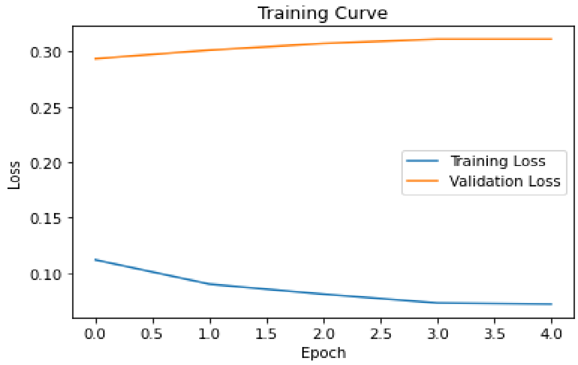


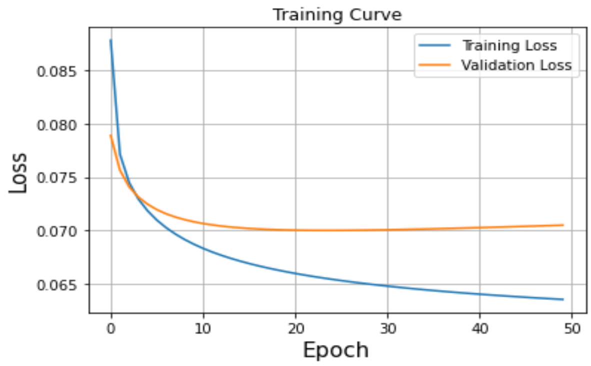
BERT Model

Baseline

Discussion and Learnings (4 points)

Below are training curves between the BERT model and baseline model.





BERT Model

Baseline

- In terms of accuracy, the BERT model tends to have a better capacity for learning. Among the 5 epochs, the transformer in the first epoch already did a better job compared to the baseline.
- The F1 score reveals that the Bert model outperforms the baseline on every label. Furthermore, the ranking of the labels (neutral, happiness, anger, surprise, and digest) by F1 score is the same for both models, which aligns with the label distribution in the dataset.
- A distinct difference between the BERT and baseline models is the tendency of their results. From the result above, the BERT model tends to produce results with a focus on one particular dimension, while the baseline model favors different dimensions. This is likely due to the distribution of our training dataset, which only has a few examples with multiple labels. As a result, the BERT model learned to predict more on a single dimension.
- A potential flaw of the progressing model is about the training time. Since the BERT model consists of 110M parameters, it takes a longer time for doing the training. Even for the small dataset we have it takes approximately 1 minute to train an epoch.

Individual Contributions

- Haoran:
    - manually labeled the Emotion dataset
    - was responsible for building and training the baseline model
    - measured the performance of the baseline model
    - wrote the web app Sensify using python flask and bootstrap
- Ao:
    - manually labeled the Emotion dataset
    - fine-tuning the BERT model.
    - measured the performance of the BERT model
    - Analyzed the performance of the baseline model and the BERT model

Reference

[1] Zhang, Lei & Wang, Shuai and Liu, Bing, “Deep Learning for Sentiment Analysis: A Survey,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2018
[2] “Web API reference: Spotify for developers,” *Home*. [Online]. Available: https://developer.spotify.com/documentation/web-api/reference/#/operations/get-recommendations. [Accessed: 12-Dec-2022].

Permissions

permission to post video: yes/yes

permission to post final report: yes/yes

permission to post source code: yes/yes