I Couldn’t Find a Therapist so I Tried to Build One - Part I

Capstone 1 Milestone + Final Report

Kevin Chung
Springboard 2021
05/01/2021

Table of Contents

1. **Problem Statement & Project Proposal**
    a. Introduction
    b. Project Plan
        i. Scope
        ii. Milestones
        iii. Metrics

2. **Data Wrangling Report**
    a. Sourcing Data
    b. Wrangling Data

3. **Exploratory Analysis**
    a. Data Story - exploratory data analysis

4. **Machine Learning**
    a. Introducing BERT
    b. Methods & Requirements
    c. Results
    d. Visualizations

5. **Discussions**
    a. Limitations
    b. Next Steps

Problem Statement & Project Proposal

Introduction

According to the National Institute of Mental Health (NIMH), in the year 2017, it was estimated nearly 46.6 million Americans suffer from a mental illness. That translates to roughly 1 in 5 Americans. In the same year, it was also estimated that roughly 11 million Americans have experienced at least one episode of severe and debilitating major depression. Of those who reported they live with a mental illness, only 42.6% received treatment; and nearly 35% of those experiencing a major depressive episode did not receive treatment. Although accessibility to mental healthcare is slowly improving, out-of-pocket costs of treatment, stubborn social and cultural stigmas, and an unmet demand for mental health professionals remain as huge barriers for the most vulnerable groups to find viable treatment options.

However, an interesting phenomena occurred alongside with the explosive emergence of social media--as global social networks grew, people became more open about sharing their experiences, and giving emotional validation and support to their fellow netizens. In other words, people began hacking a way through the social network landscape to establish support systems which were otherwise limited in accessibility, and/or availability in the “offline” space. This resulted in the massive proliferation of textual data which document the cognitive processes of individuals who may, or may not, have a mental illness, and certainly of those who are going through powerful experiences seeking validation and support. I believe this data presents an opportunity to serve a chronically underserved population, and is also an underrated source for insights in the mental health sciences.

Scope

In this project, I propose to create a multiclass classifier capable of distinguishing posts from one mental health support group from five other mental health support groups -- each forum facilitating discussions around a specific mental illness diagnosis. The goal then, is to be able to predict the class that an input forum post belongs to, of the six mental illness diagnoses I trained a model to classify.

Milestones

This capstone project, part one of two, will accomplish the above task by leveraging an out-of-the-box solution that BERT provides, called run_classifier.py. However, before I can get started, there remains a huge challenge of finding a dataset filled with highly intimate anecdotes about one’s experiences with a mental illness of a specific diagnosis. I have to find six of them to be precise (one for each of my classes). Despite my best efforts, I could not discover such convenient and publicly accessible datasets which define the first major milestone of this project -- constructing the datasets for each class.

Once the datasets have been constructed, the next appropriate step is to explore them. Since NLP tasks involve working with largely text data, inferencing with traditional statistics is often limited in utility, or outputs useless information. However, there are simple methods to explore the sentiments and diction of text datasets, and ways to discover important features. I will employ those techniques to achieve a better understanding of the datasets.

As an extension from the previous milestone, but deserving of its own milestone definition, is preprocessing the data. Preprocessing will remove all unimportant characters and stopwords which will help make the dataset simpler for the model to understand. Higher dimensionality is not always good, if the data is too sparse to contain any useful information. Preprocessing can help reduce unimportant features and make the topology of the dataset less sparse. This step will also create proper input formats for BERT to take in to run its classification training and prediction loops.

Metrics

To evaluate the model’s viability I will run prediction loops over an unseen holdout dataset, and visualize the prediction results as a heatmap. Then, to evaluate the model’s performance for each class, I will visualize subplots for each class, comparing true labels against predicted labels.

Data Wrangling Report

Sourcing Data

The first milestone -- obtaining the data -- was a formidable challenge. Not only is the classification task highly specific, it also contains highly sensitive material. Therefore, finding already-curated datasets of personal anecdotes recounting experiences with specific mental disorders is extremely difficult. I decided it would be easier to instead collect and organize the data myself.

The first place I looked at was Reddit. I was not surprised to find a subreddit for each of the classes I chose. The classes I chose are: **Depression, PTSD, CPTSD, Borderline Personality Disorder (BPD), Bipolar, and Dissociation**. The six subreddits I targeted for extracting data were chosen because of their commonality and strong community engagements. I also intentionally chose disorders with known comorbidities with each other to try and make the classification task more difficult.

Reddit offers a python wrapper called PRAW to interface its API. I was able to extract the posts from the subreddits using PRAW; however, there was a limit in the amount of data I could extract per token, and I was only able to extract 1000 posts per category at a time. To combat this problem I chose several categories to work with, drawing as many samples as Reddit would allow me to extract. The categories I selected were:

- top()
- rising()
- new()
- hot()

The resulting dataset had issues, however. The first problem I noticed was, since typing the “body” portion of a Reddit post is optional, some subreddits had disproportionate amounts of null value cells compared to others. This led to having to rely on “titles” of the posts, which were far shorter in word count, and also lacked the richness in context that a full post contains. It was clear I would need to find more samples to add to the dataset. But I decided to roll with what I had because I was both curious and eager to see if BERT would be able to classify Reddit posts specifically. In part II of the capstone, I obtain more data via web scraping.

Wrangling Data

Text data requires preprocessing and conversions to numbers before it can be fed into a model for training. The theoretical and application challenge of NLP is thus heavily rooted in figuring out how to best *represent* text in terms of numbers. How does one capture the meaning of words in numbers? How does one represent their significance over each other to the overall document? A full discussion about the evolution of text embeddings is beyond the scope of this paper; however, several embedding methods will be employed and discussed throughout the capstone.

The first task I had to complete before I can dig further into the dataset for insights was to preprocess the text documents. The first strategy I employed was to tokenize the text using regular expression tokenizer (regexptokenizer), and removing stopwords. Then, I lemmatized the text. The NLTK package enabled me to complete these tasks with ease. After these steps, the dataset had been cleaned of special punctuations and characters, clutter of stopwords, and words have been reduced down to their root form. The dataset was now ready for some EDA.

Exploratory Analysis

Data Story

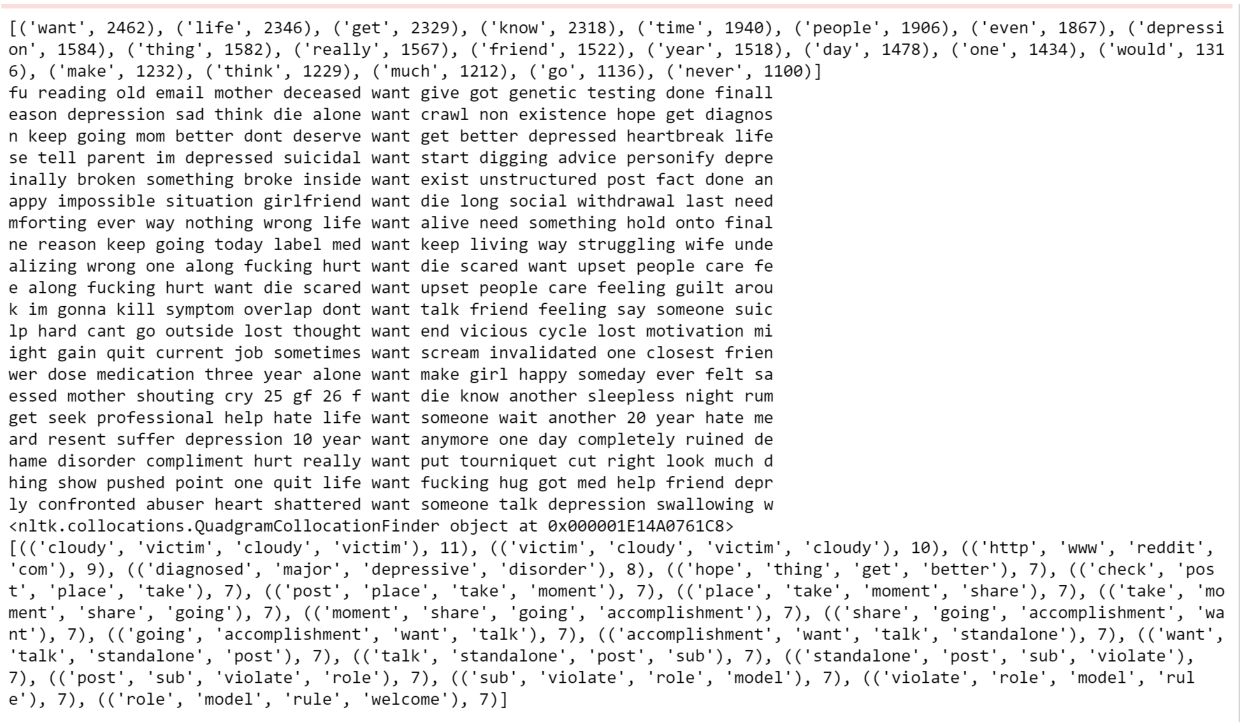
The stories I was able to collect through Reddit were incredibly rich and eye-opening. While I love to read through the dataset when I have time, this is an inefficient way for me to get to know the dataset.

The first exploratory task I did was to find the Part-of-Speech (POS) tag for every word. I then counted the numbers of unique tags present. Those tags were later plotted against each other to observe for any patterns of overarching writing styles. Because the posts had text in both the title section and body section, I stacked title and body posts as if they were individual documents.

Before plotting the results, I wanted to look out for any words that contributed little meaning to the overall document, but appear frequently enough to overshadow keywords that hold more weight. I first took the frequency distribution of the words, then observed concordances and collocations of the top words. I repeated the above steps for all six classes.

The exclusion of words that did not contribute much meaning to the overall context proved to be an important step because I discovered it is customary for certain subreddits to begin the document in certain ways. For example, r/depression often started with “Anyone else feel like [...]”, which may be an important classifying feature during training, but did not help my understanding of the data. The collocation search identified other words that appeared together frequently, which helped to update my exclusions list. I then rebuilt the corpus with exclusions in place and found that the concordance and collocation results were much more insightful and revealed more context.

TW/CW - potentially triggering words/sentences shown in figures in the next page



```
[('want', 2462), ('life', 2346), ('get', 2329), ('know', 2318), ('time', 1940), ('people', 1906), ('even', 1867), ('depressi
on', 1584), ('thing', 1582), ('really', 1567), ('friend', 1522), ('year', 1518), ('day', 1478), ('one', 1434), ('would', 131
6), ('make', 1232), ('think', 1229), ('much', 1212), ('go', 1136), ('never', 1100)]
fu reading old email mother deceased want give got genetic testing done finall
eason depression sad think die alone want crawl non existence hope get diagnos
n keep going mom better dont deserve want get better depressed heartbreak life
se tell parent im depressed suicidal want start digging advice personify depre
inally broken something broke inside want exist unstructured post fact done an
appy impossible situation girlfriend want die long social withdrawal last need
mforting ever way nothing wrong life want alive need something hold onto final
ne reason keep going today label med want want keep living way struggling wife unde
alizing wrong one along fucking hurt want die scared want upset people care fe
e along fucking hurt want die scared want upset people care feeling guilt arou
k im gonna kill symptom overlap dont want talk friend feeling say someone suic
lp hard cant go outside lost thought want end vicious cycle lost motivation mi
ight gain quit current job sometimes want scream invalidated one closest frien
wer dose medication three year alone want make girl happy someday ever felt sa
essed mother shouting cry 25 gf 26 f want die know another sleepless night rum
get seek professional help hate life want someone wait another 20 year hate me
ard resent suffer depression 10 year want anymore one day completely ruined de
hame disorder compliment hurt really want put tourniquet cut right look much d
hing show pushed point one quit life want fucking hug got med help friend depr
ly confronted abuser heart shattered want someone talk depression swallowing w
<nltk.collocations.QuadgramCollocationFinder object at 0x000001E14A0761C8>
[(('cloudy', 'victim', 'cloudy', 'victim'), 11), (('victim', 'cloudy', 'victim', 'cloudy'), 10), (('http', 'www', 'reddit',
'com'), 9), (('diagnosed', 'major', 'depressive', 'disorder'), 8), (('hope', 'thing', 'get', 'better'), 7), (('check', 'pos
t', 'place', 'take'), 7), (('post', 'place', 'take', 'moment'), 7), (('place', 'take', 'moment', 'share'), 7), (('take', 'mo
ment', 'share', 'going'), 7), (('moment', 'share', 'going', 'accomplishment'), 7), (('share', 'going', 'accomplishment', 'wa
nt'), 7), (('going', 'accomplishment', 'want', 'talk'), 7), (('accomplishment', 'want', 'talk', 'standalone'), 7), (('want',
'talk', 'standalone', 'post'), 7), (('talk', 'standalone', 'post', 'sub'), 7), (('standalone', 'post', 'sub', 'violate'),
7), (('post', 'sub', 'violate', 'role'), 7), (('sub', 'violate', 'role', 'model'), 7), (('violate', 'role', 'model', 'rul
e'), 7), (('role', 'model', 'rule', 'welcome'), 7)]
```

Figure 1. *- r/depression: frequency dist. of words, concordances of "want", and quadgram collocation output.*

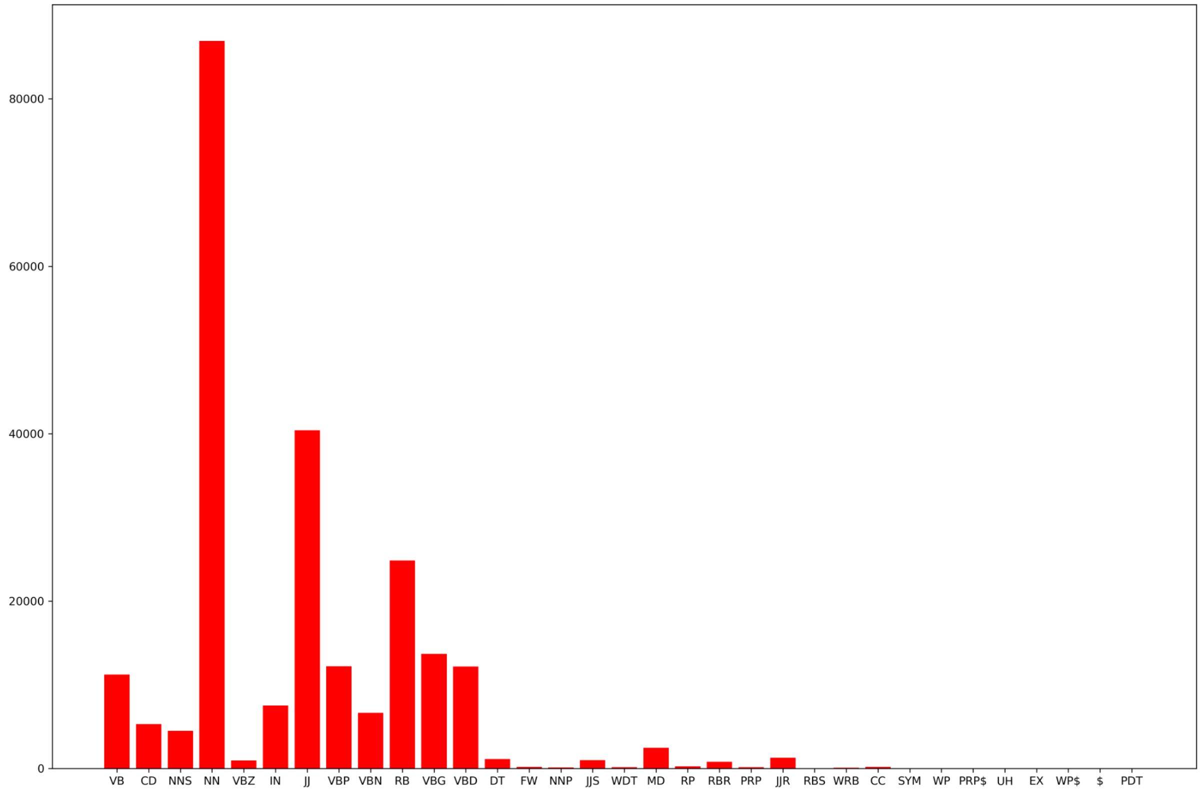


Figure 2. *- r/depression: distribution of POS tag frequency.*

Machine Learning

Introducing BERT

BERT stands for Bidirectional Encoder Representations from Transformers. It is a state-of-the-art NLP model developed by Google, and it is based on the attention mechanism for pre-training, and uses a stack of 12 transformers to learn and predict from a dataset. Since its debut, many other transformer models have been developed and completely changed the world of NLP from using traditional RNN, CNN, and LSTM to using transformer based models instead.

BERT offers an out-of-the-box code for text classification tasks called run_classifier.py. I selected the bert-based-uncased model which was pre-trained with the COLA dataset. This model contains 12 transformer-encoder layers, 768 hidden size, and 12 attention heads. In part II of the capstone, I explore both non-deep learning methods for classifications, and BERT after it has been fine tuned.

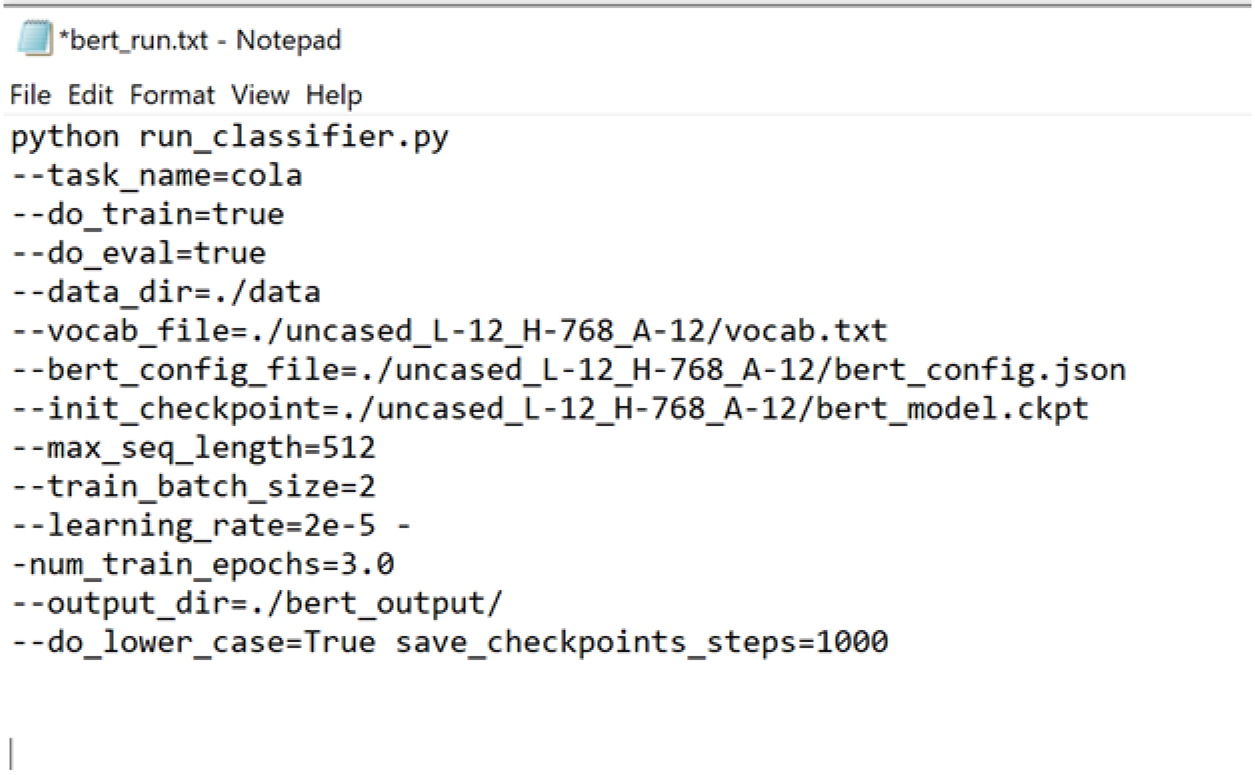
Requirements

BERT requires three input .tsv files to run its classifier: train, test, and dev.tsv stored in a data repository created beforehand. The train and validation (dev) datasets can be created by utilizing scikit-learn's train_test_split() method, and must be configured to the correct format. The **train.tsv** file is formatted with the following four columns:

- **Column 0:** A unique id for each row.
- **Column 1:** A label (target) for the data we want to classify, as an integer.
- **Column 2:** A 'throwaway' column with a random alphabet.
- **Column 3:** A column with *untokenized* text we want to classify.

The **test.tsv** file simply requires a unique id column, and a corresponding text we wish to test the trained classifier on. However, it is important to keep a record of the true labels for the test dataset so they can be mapped back to the text and compared with the predicted label to calculate the accuracy of the model.

Once the files have been prepared, they must be stored in a data repository, which are later referenced when calling run_classifier.py. Some hyperparameters can be adjusted at this stage. As seen in the figure below, I set the max sequence length at the maximum, at 512, and chose to sacrifice training speeds by setting the batch size at 2. The tradeoff was worth it because the length of the documents were mostly quite large.

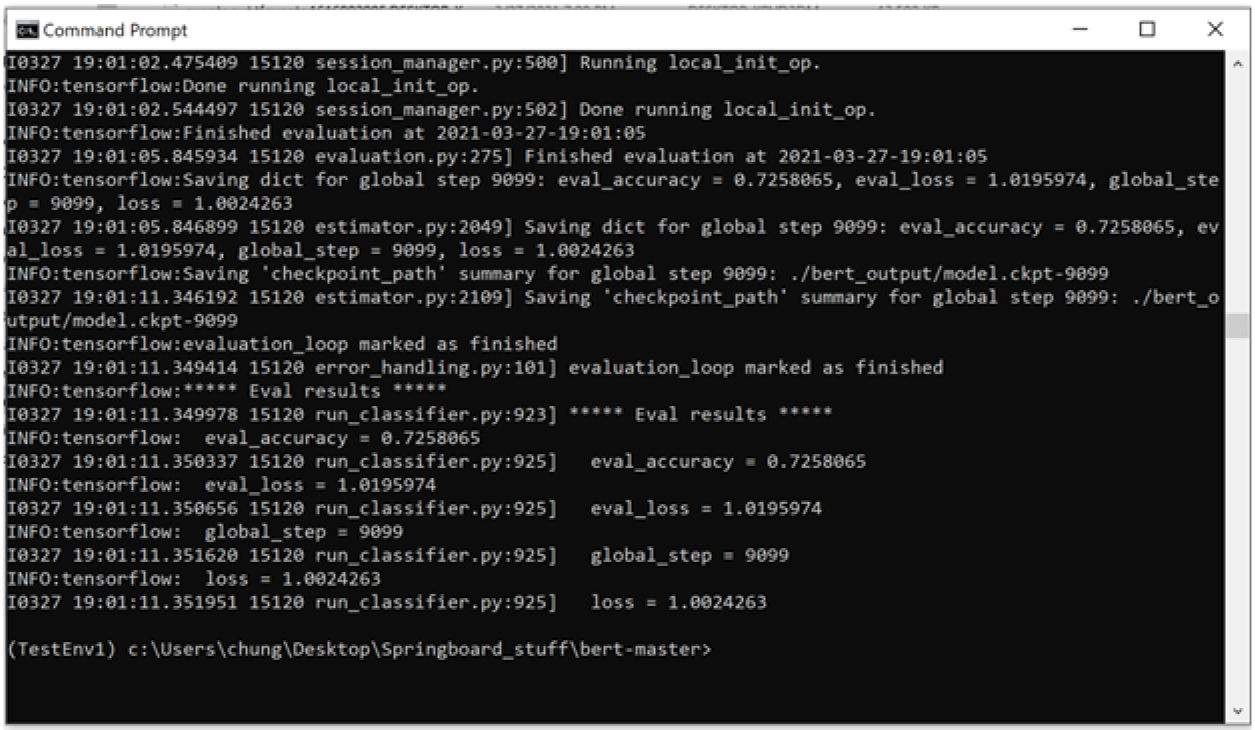


```
python run_classifier.py
--task_name=cola
--do_train=true
--do_eval=true
--data_dir=./data
--vocab_file=./uncased_L-12_H-768_A-12/vocab.txt
--bert_config_file=./uncased_L-12_H-768_A-12/bert_config.json
--init_checkpoint=./uncased_L-12_H-768_A-12/bert_model.ckpt
--max_seq_length=512
--train_batch_size=2
--learning_rate=2e-5 -
-num_train_epochs=3.0
--output_dir=./bert_output/
--do_lower_case=True save_checkpoints_steps=1000
```

Figure 3. - hyperparameters used to train the model.

Results

The accuracy scored at roughly 72.5%. The result isn’t state-of-the-art quite yet, but it is a great start. The validation loss and the training loss values were 1.095974, and 1.0024263 respectively. The fact that training and validation loss values are so similar informs me that at just 3 epochs, the model may have been under-trained and could potentially perform better at higher epochs.

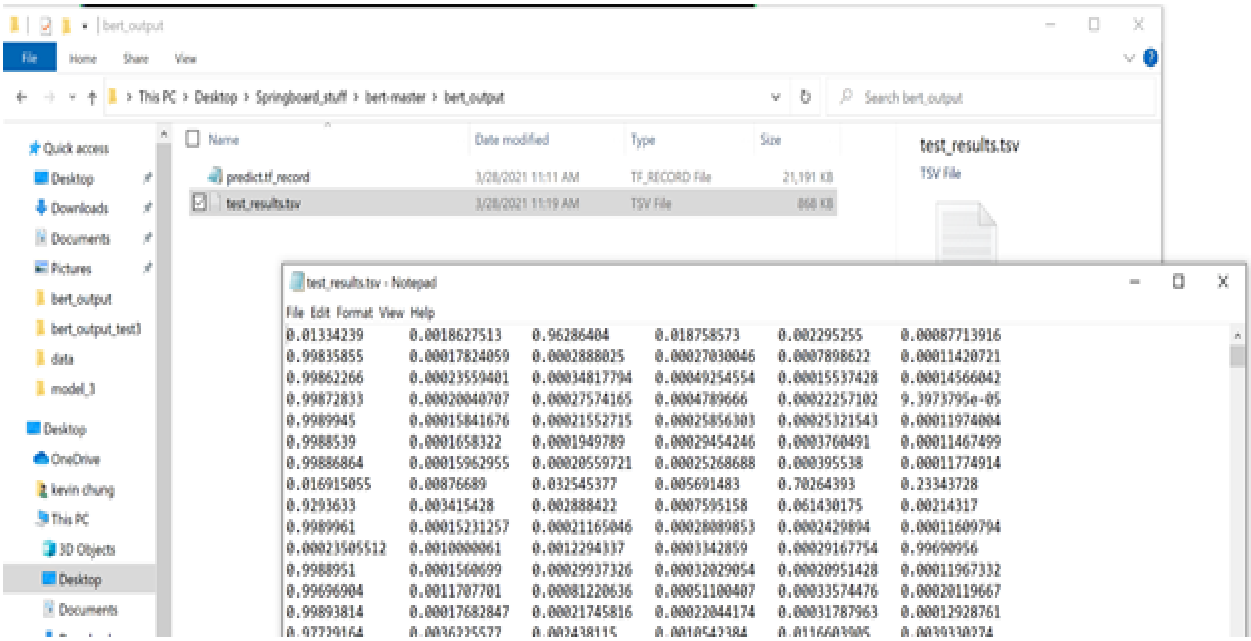


```
I0327 19:01:02.475409 15120 session_manager.py:500] Running local_init_op.
INFO:tensorflow:Done running local_init_op.
I0327 19:01:02.544497 15120 session_manager.py:502] Done running local_init_op.
INFO:tensorflow:Finished evaluation at 2021-03-27-19:01:05
I0327 19:01:05.845934 15120 evaluation.py:275] Finished evaluation at 2021-03-27-19:01:05
INFO:tensorflow:Saving dict for global step 9099: eval_accuracy = 0.7258065, eval_loss = 1.0195974, global_ste
p = 9099, loss = 1.0024263
I0327 19:01:05.846899 15120 estimator.py:2049] Saving dict for global step 9099: eval_accuracy = 0.7258065, ev
al_loss = 1.0195974, global_step = 9099, loss = 1.0024263
INFO:tensorflow:Saving 'checkpoint_path' summary for global step 9099: ./bert_output/model.ckpt-9099
I0327 19:01:11.346192 15120 estimator.py:2109] Saving 'checkpoint_path' summary for global step 9099: ./bert_o
utput/model.ckpt-9099
INFO:tensorflow:evaluation_loop marked as finished
I0327 19:01:11.349414 15120 error_handling.py:101] evaluation_loop marked as finished
INFO:tensorflow:***** Eval results *****
I0327 19:01:11.349978 15120 run_classifier.py:923] ***** Eval results *****
INFO:tensorflow:  eval_accuracy = 0.7258065
I0327 19:01:11.350337 15120 run_classifier.py:925]   eval_accuracy = 0.7258065
INFO:tensorflow:  eval_loss = 1.0195974
I0327 19:01:11.350656 15120 run_classifier.py:925]   eval_loss = 1.0195974
INFO:tensorflow:  global_step = 9099
I0327 19:01:11.351620 15120 run_classifier.py:925]   global_step = 9099
INFO:tensorflow:  loss = 1.0024263
I0327 19:01:11.351951 15120 run_classifier.py:925]   loss = 1.0024263

(TestEnv1) c:\Users\chung\Desktop\Springboard_stuff\bert-master>
```

Figure 4. - *BERT train/eval results*

To evaluate the model on an unseen dataset, I simply formatted the data I reserved for training into a BERT test.tsv input and loaded it into the data repository. After running the prediction loop (without training), the results came back in a .tsv file with six columns of percentile likelihood of an example belonging to each class.



```
Name               Date modified        Type            Size
predict.tf_record  3/28/2021 11:11 AM   TF_RECORD File  21,191 KB
test_results.tsv   3/28/2021 11:19 AM   TSV File        868 KB
```

```
0.01334239      0.0018627513    0.96286404      0.018758573     0.002295255     0.00087713916
0.99835855      0.00017824059   0.0002888025    0.00027030046   0.0007898622    0.00011420721
0.99862266      0.00023559401   0.00034817794   0.00049254554   0.00015537428   0.00014566042
0.99872833      0.00020040707   0.00027574165   0.0004789666    0.00022257102   9.3973795e-05
0.9989945       0.00015841676   0.00021552715   0.00025856303   0.00025321543   0.00011974004
0.9988539       0.0001658322    0.0001949789    0.00029454246   0.0003760491    0.00011467499
0.99886864      0.00015962955   0.00020559721   0.00025268688   0.000395538     0.00011774914
0.016915055     0.00876689      0.032545377     0.005691483     0.70264393      0.23343728
0.9293633       0.003415428     0.002888422     0.0007595158    0.061430175     0.00214317
0.9989961       0.00015231257   0.00021165046   0.00028089853   0.0002429894    0.00011609794
0.00023505512   0.0010000061    0.0012294337    0.0003342859    0.00029167754   0.99690956
0.9988951       0.0001560699    0.00029937326   0.00032029054   0.00020951428   0.00011967332
0.99696904      0.0011707701    0.00081220636   0.00051100407   0.00033574476   0.00020119667
0.99893814      0.00017682847   0.00021745816   0.00022044174   0.00031787963   0.00012928761
0.97729164      0.0036225577    0.002438115     0.0010542384    0.0116603905    0.0039330274
```

Visualizations

In order to achieve a better visual assessment on the model's performance I employed seaborn and matplotlib to chart the predictions vs true labels in a few different ways. First, I took a direct look at the predictions on a heatmap:

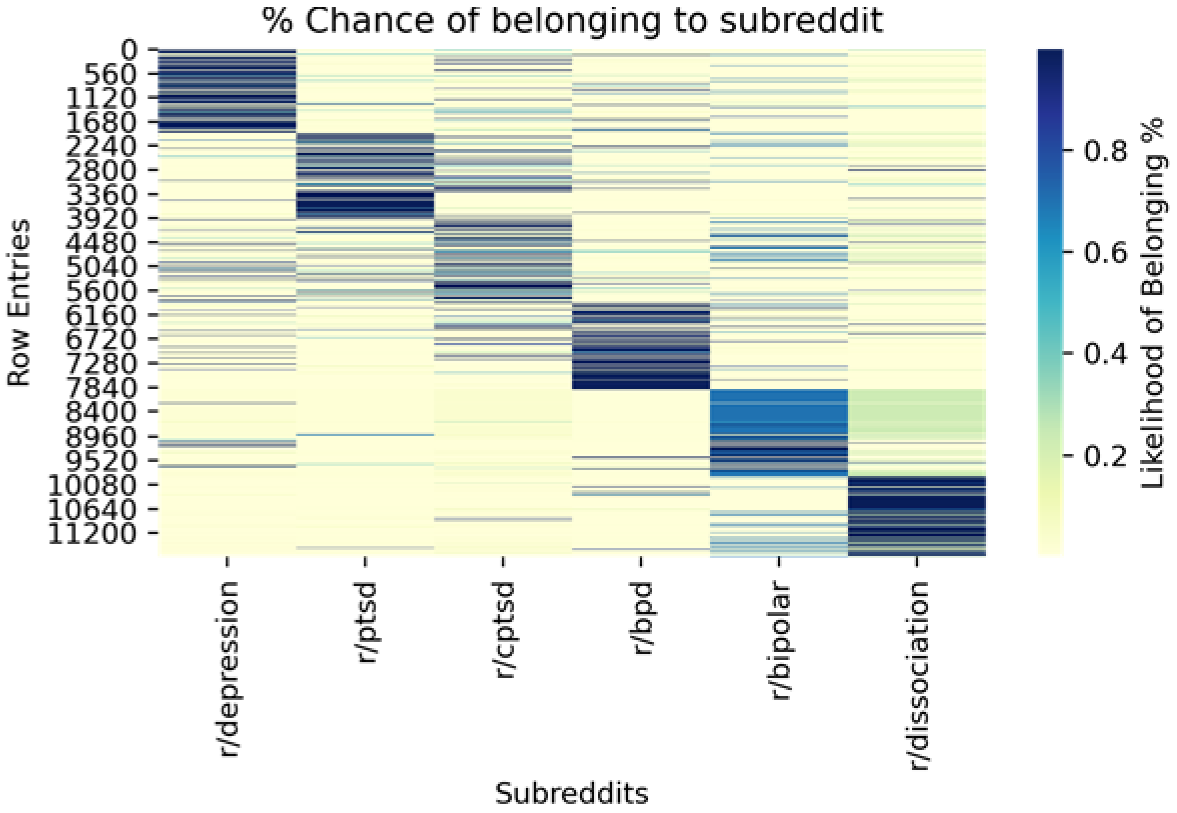
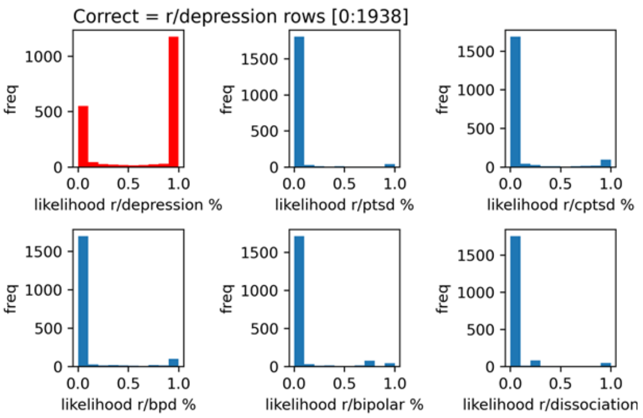
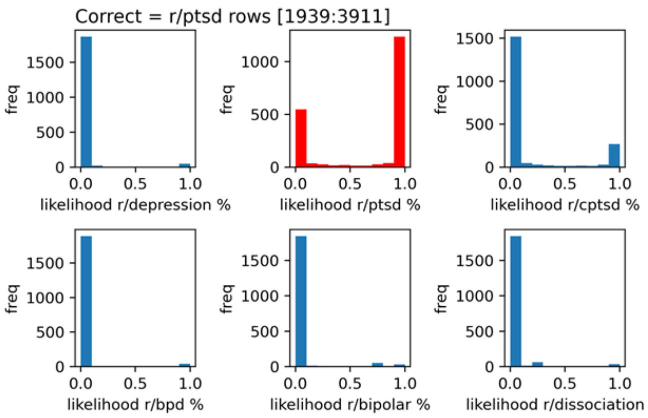
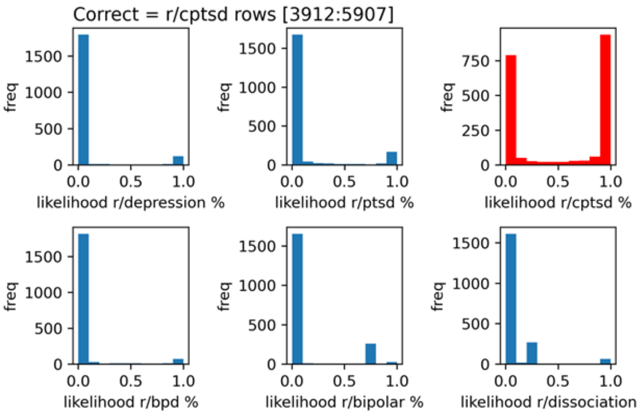


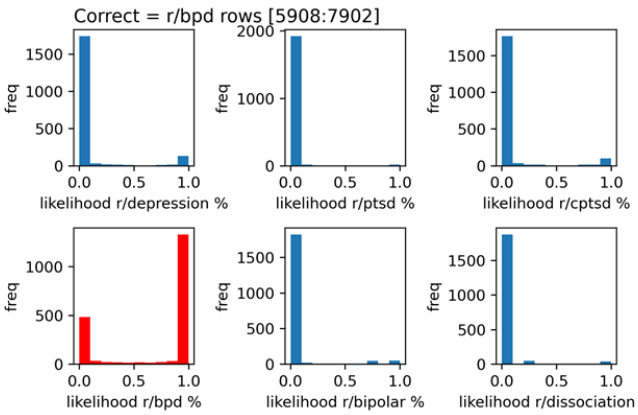
Figure 5. - Heatmap of results per class. Blue areas signify pockets of strong predictions that the model made.

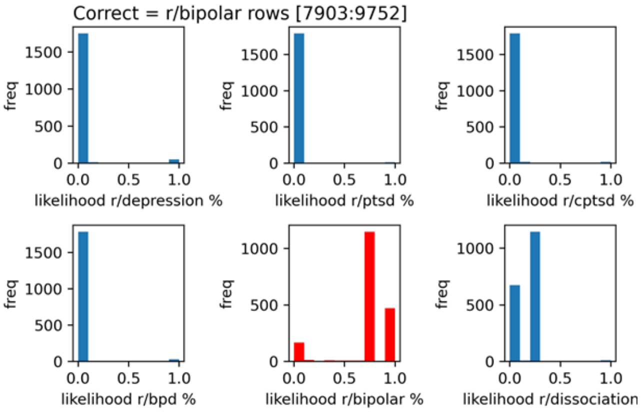
From the above visual it is clear that the model is successfully distinguishing each class to a decent degree of accuracy. A quick look also shows that there is a decent amount of bad predictions. To take a closer look, I plotted the results comparing each class:

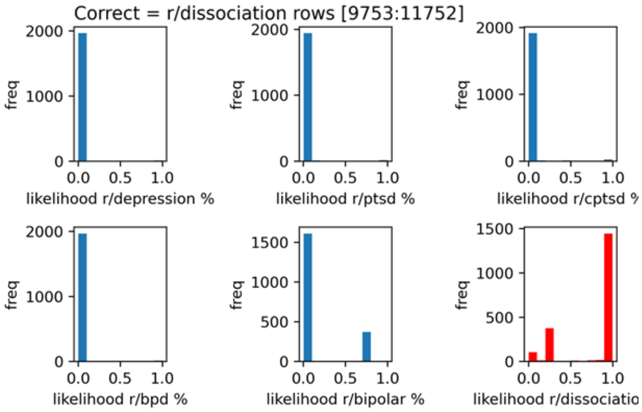








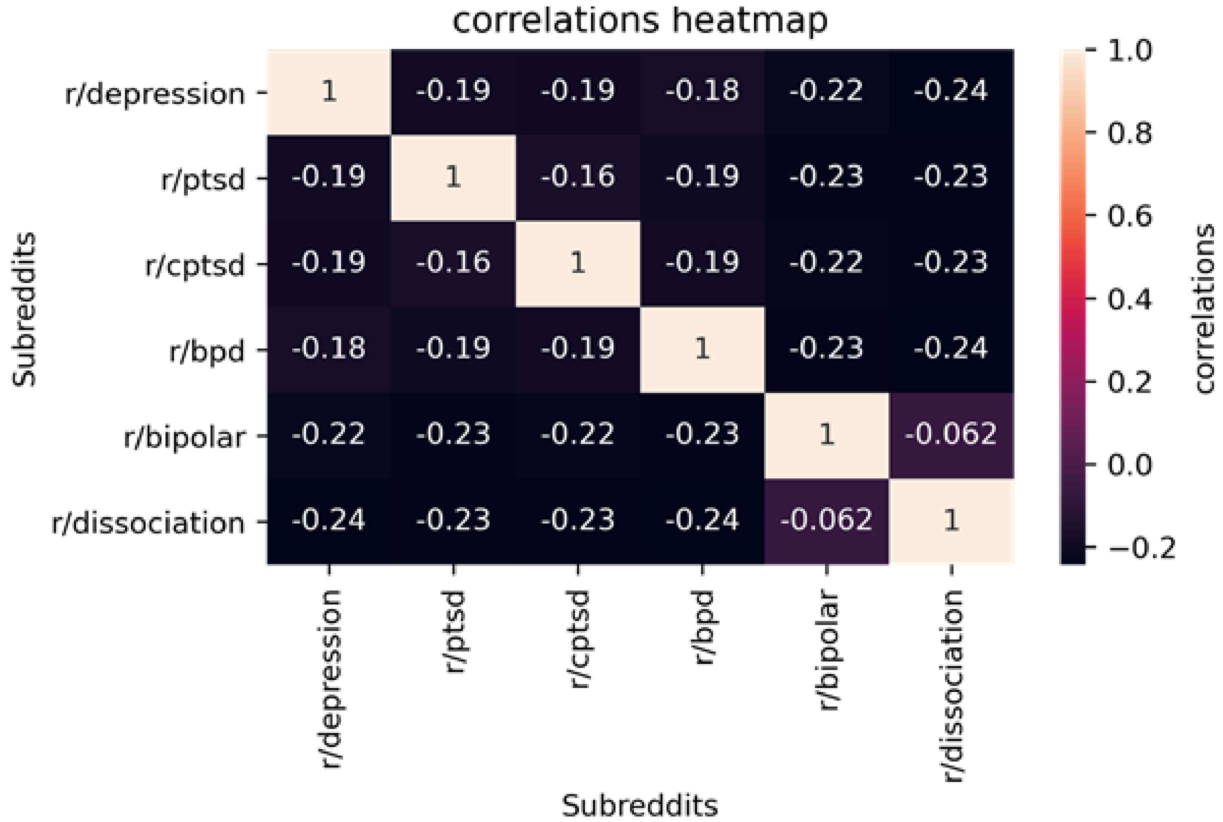




The above subplots show clear spikes where predictions were strongest in areas where the true labels matched the model's classifications. An interesting byproduct of this study is that BERT was actually capable of distinguishing PTSD and CPTSD, in spite of their pathological and symptomatic similarities. Furthermore, CPTSD is currently not considered to be its own diagnostic criteria, according to the DSM-V.

However, the ICD-11, published by the World Health Organization, *does* classify CPTSD as its own diagnostic criteria. This distinction is particularly interesting because while the ICD-11 is an internationally researched and validated diagnostic manual, the DSM-V is published by the APA which is an American publishing association for psychological studies. Could this be suggesting a bias in the field of American psychology? There may not be sufficient evidence here to back such a claim; however, mounting evidence in the literature strongly suggests that psychological research in the United States should be thoroughly examined for bias. Interested readers should look into the replication and validation crisis in psychology.

The classification result also reveals some correlationary confusion. Namely, bipolar and dissociative labels are seen to confuse each other, as well as PTSD and CPTSD. This is clearly demonstrated in a correlational matrix shown below.



	r/depression	r/ptsd	r/cptsd	r/bpd	r/bipolar	r/dissociation
r/depression	1	-0.19	-0.19	-0.18	-0.22	-0.24
r/ptsd	-0.19	1	-0.16	-0.19	-0.23	-0.23
r/cptsd	-0.19	-0.16	1	-0.19	-0.22	-0.23
r/bpd	-0.18	-0.19	-0.19	1	-0.23	-0.24
r/bipolar	-0.22	-0.23	-0.22	-0.23	1	-0.062
r/dissociation	-0.24	-0.23	-0.23	-0.24	-0.062	1

It is currently unbeknownst to me why the confusion exists. However, it is a relief to know that the model is not overfitted, and is doing a relatively good job of representing each class.

Discussions

Limitations

The above results are decent, but they are not state-of-the-art decent. Clearly, there is room for improvement. The first and the most obvious limitation to this study is the lack of quality data. I was able to artificially inflate the document count by treating the title and body sections of a post as individual examples; however, the classes remain imbalanced and are underwhelming in context rich content. It became very clear that I need to put more effort towards collecting data. Reddit alone simply isn't cutting it.

The second limitation I found was that experimenting with hyperparameters, and their various combinations, is simply too computationally expensive for my refurbished 2015 Macbook Pro. It was clear that fine tuning and increasing the epoch count was beyond my current computational capacity. I needed to switch the environment to a computer more suited for machine learning tasks.

The third limitation of this study is that non deep learning models haven't been compared. Afterall, as hyped up as BERT may be, it isn't guaranteed that it is the best model to fit datasets relating to mental health disorders. Thus, a comparative analysis of various models seems to be necessary to identify the best model for the classification task.

Next Steps

In the next phase of the capstone, I will address the above limitations by collecting more data, switching to a more powerful computing environment, and by rigorously iterating and evaluating the performances of several different models. I will then return to BERT by pre-training an attention head and fine tuning to predict over a holdout test dataset.