Detecting Signs of Depression from Social Media Text

TzuHwan Seet (tseet), Sarah Branse (sbranse), Andrew Liu (aliu54), Kalvin Lam (klam4)

Github: https://github.com/tzuhwan/dl-final-project

Introduction:

Depression is one of the main causes of disability globablly, and suicide resulted from depression is the second leading cause of death for young adults. Mental illness such as depression often remains undiagnosed because of social stigma, so suicide preventative programs can fail to reach people who need help. With the surge of social media use over the past decade, people are more likely to talk about mental health issues and emotions with online forums. Computational NLP methods can isolate emotions from online discussions to identify mental health cues.

The paper we chose to re-implement, *Detecting Early Onset of Depression from Social Media Text using Learned Confidence Scores* [1], seeks to develop a structural prediction model that can detect early onset of depression from users' posts on Reddit. We chose to implement this paper for several reasons. All of us were interested in pursuing an application of NLP within our model, and this paper in particular seemed very relevant with the rise of social media. The model we've implemented utilizes many topics we've learned about in class including word embeddings and natural language processing techniques.

Data

We utilized the eRisk (Early Detection on the Internet) dataset, which is developed from an annual workshop held by CLEF (Conference and Labs of the Evaluation Forum). We've received access to the 2018 and 2020 eRisk dataset from the CLEF organizers. The eRisk dataset has two tasks, detecting early depression and anorexia. The paper we are replicating focuses on early detection of depression in the 2018 data. We replicated the paper's methodology on the 2020 dataset. The 2020 dataset has 104 depressed users and 319 non-depressed users as training data, and 40 depressed and 30 non-depressed users as testing data. There are a total of 70446 submissions, only 11691 of which are submitted from users with depression. The submissions are over a 2-3 year range.

Before beginning the model, we had a significant amount of preprocessing to develop the NLP pipeline. The initial challenge was determining how to process and stem the XML data files. We had to decide as a group the best way to store the data and what class structure would fit best for the model. The class hierarchy we design is detailed below.

The User class contains the user ID and posts of a given user. The Post class represents a single user post, containing the date, text, title, and info fields. We've created a method get_data that extracts all the data for each user and their posts, and creates a User instance for each subject XML file. Our preprocess method sets each User with the appropriate label, depressed or non-depressed.

For each post, we tokenized and stemmed the words in the texts and titles. The users' texts are cleaned, by transforming the text into lowercase and removing punctuation and stopwords. The numbers and URLs in texts were replaced with specific tokens and then stemming was done with Porter Stemmer. Since the number of submissions from non-depressed users is so much higher than depressed users, we downsampled the majority class to a ratio of 2:1. Since there were 104 depressed users, we removed about a third of the non-depressed users to ensure there was a 2:1 ratio.

Methodology

Our methodology focuses on our base goal which aimed to train the model to classify if the user is depressed or not, without early detection. With the Class hierarchy detailed previously, we converted users into topic model embeddings to feed into our neural network architecture.

At a high level, we defined our topic modelling pipeline in a way such that our topic modelling pipeline takes in a user as input and outputs a topic embedding that looks like [weight of topic 1, ..., weight of topic n]. In essence, our topic modelling pipeline determines, for each user, what topics they talk about most.

The way we accomplished this was by first representing each user as a bag of words that represented all posts that the user has written. We did this by first iterating through all users and accumulating all posts that each user has ever written. Then, once we had all posts for each user collected in one spot, we created a dictionary using gensim's corpora. Then, once we had this dictionary created, we converted each user (represented by all posts that the user has ever written) into a bag of words representation by using our dictionary.

Once we had each user represented as a bag of words, we moved onto the training phase of our Latent Semantic Indexing (LSI) Model. We trained our LSI Model on all of our users, passing in all of the users (represented as bags of words), our dictionary we created, as well as the number of topics we chose, 128 (which we decided upon by researching our topic).

Once we had our LSI model trained, we printed the 128 topics for inspection. This enabled us to see what topics the model found significant as well as what words were important to each topic. With our trained LSI model in hand, we used it to generate topic embeddings for each user. One-by-one, we passed in users (represented as bags of words) into our trained LSI model, and our LSI model outputted a topic embedding for the user. These outputted topic embeddings enabled us to see, for each user, what topics they gravitated towards.

These topic embeddings were then fed into the linear layer of the neural network. We used a similar architecture defined in the paper: 3 hidden layers of sizes 512, 256, and 256 respectively, using a Leaky ReLu and Dropout of 0.2. The last linear layer applied a sigmoid activation function. The hyperparameters of the model are detailed in Table 1 below.

Table 1: Hyperparameters

Number of topics	128
Activation	LeakyRelu in between layers, Sigmoid at last layer

Learning Rate	1e-3
Number of Epochs	10

Results

Since this a binary classification test, we evaluated many statistical measures of performance. We developed the model to return 5 metrics, as detailed in Table 2. We did not solely rely on accuracy because that metric could be misleading if the classifier tends to predict the label of the majority class (non-depressed users). We trained on 231 users (75% of the available data set and tested on 80 users (25%). The training was done with a batch size of 50. The testing was done with a batch size of 20.

The results of the testing data are detailed in Table 2. In particular, the F1 score, precision, and recall values help us compare our model with the study's implementation. The values in the table were the average of 10 training and testing runs with the model. There is not a direct comparison, however, since the study analyzed these metrics based on a latency cost function which adds a scaled time delay for predictions. Our model trains on the data without acknowledging the time of each user post. The paper details the F1 score, precision, and recall for 7 different confidence levels. For comparison with our model, we included a range of the paper's values within our table. The results of the paper did not indicate values for loss, or accuracy.

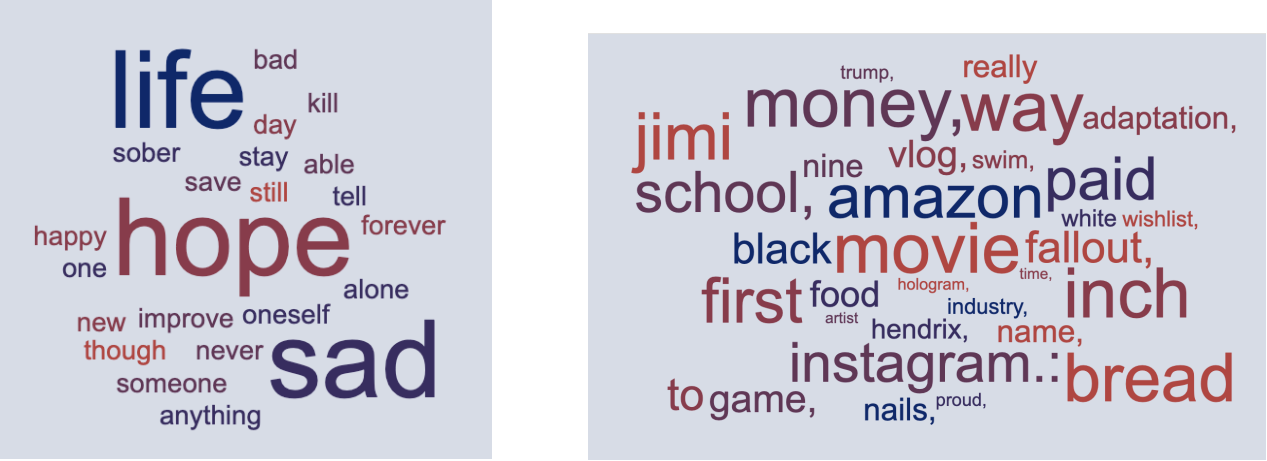
Table 2: Model Results

Metric	**Meaning**	**Value** (Average of 10 runs)	**Comparable Value from Study**
Loss	Sigmoid cross-entropy loss for binary classification	2.0515	N/A
Accuracy	% Correct predictions	0.75	N/A
Precision /	(# True positives) / (predicted	0.76	0.15-**0.25**

Positive Predictive Value	#depressed users)		
Recall	(# True positives) / (total # depressed users)	0.73	0.29-**0.71**
F1 Score	2* (Precision * Recall) / (Precision + Recall)	0.74	0.25-**0.30**

Analysis:

As indicated by Table 2, our model yielded very positive results. It scores higher on precision, recall and F1 score than the paper's best implementation. This is likely because we train the model with all of the users' posts, instead of providing the data in chunks as the original authors did to implement early detection. In order to validate these results, we also tested the model with various labels to ensure that the rate of false positives/false negatives was sensible given the inputs. In addition, we printed out topics to identify if our model successfully learned features of depressed/non-depressed users. When we explore the topics returned by the LSI model, we discover similar patterns to the results in the paper. The posts of depressed users in our dataset tend to relate to topics/themes such as *alone, depression, hugs, hopeful, happy/sad* and profanity. On the other hand, the posts of non-depressed users tend to relate to hobbies or current affairs such as *swim, school, trump, instagram*. Figure 1 illustrates common words seen by depressed and non-depressed users.



(a) **(b)**

Figure 1: Common topics for (a) depressed users and (b) non-depressed users

Challenges

I. Preprocessing

A large challenge with beginning this project was understanding how to preprocess the data. The data we collected from the eRisk dataset was stored in XML files, so we had to determine how to process and stem the XML data files. We had to decide as a group the best way to store the data and what class structure would fit best for the model. In addition, the study utilized a preprocessing method called Latent Semantic Indexing (LSI), which is a dimensionality-reduction technique. This procedure extracts embeddings that contain information about the text in users' posts. A challenge for our group was conceptually understanding how LSI is used on data and what the embeddings represent. We utilized an LSI model from the gensim library, and spent time testing the model by printing topic weights for a given topic. In addition, we had initial difficulty understanding the size of the embeddings matrix, since each user's posts have a different corpus length.

Creating correct matrix dimensions was a new challenge for us, as in previous assignments our matrices were always rectangular in size. However, because the

size of topic embeddings depend on the number of words in the user's text, our matrices were not uniform in size. To fix this initially, we decided to add padding of zeros to the end of matrices. However, we were worried this might distort the information contained in the embeddings. We eventually found a better solution, where we create the topic embeddings from a dictionary of words shared by all users instead of individual user word dictionaries, allowing uniform matrices to be created without padding.

II. Metric Evaluation

Once we were able to successfully train our dataset, we had challenges evaluating and interpreting the results of the test data. We developed our model using the Sequential class from Keras. We used built-in methods to add layers, train the data, and evaluate the results. Although the API is convenient and powerful, it was also a real black-box and we initially encountered difficulties trying to get under the hood when we had to test and debug. If we had more time, we would do more thorough testing of the methods to improve transparency.

Reflection

I. Conclusion

Based on the paper *Detecting Early Onset of Depression from Social Media Text using Learned Confidence Scores* [1], we were able to successfully implement a model that analyzes a user's text posted on online forums such as Reddit, and classifies if that user is depressed based on the language of their posts.

Our model first uses Latent Semantic Indexing (LSI) to extract topic modelling embeddings, which contain information about relationships between words in users' posts. Because LSI was a new technique for us, most of the challenges we encountered were in this preprocessing part of the project, and as a result we changed our approach to implementing LSI multiple times.

Our neural network uses three keras.Sequential layers to make predictions and classify if users are depressed given the topic modelling embeddings as input. The model met our target goal of being able to classify if users are depressed, scoring an average accuracy of around 75%. This was better than the paper's results, which was expected as we did not prioritize early detection and were able to use all of a user's posts for training rather than just their earlier ones.

II. Takeaways

Overall it was a rewarding experience to learn about a deep learning paper and to re-implement it. One of our biggest takeaways from the experience is that the preprocessing stage can be more challenging than one initially expects. Although Latent Semantic Index Modelling is initially outside our comfort zone, we are glad that we get to explore it more as part of the project. We really enjoyed learning about NLP applications to healthcare, and how computational methods can make meaningful predictions about user emotions based on text.

III. Extension

Given more time, we would definitely want to implement a model that is able to classify if the user is depressed or not with early detection confidence score. This would require us to rethink preprocessing as the user's posts need to be stored in a way that reflects the date of writing. In particular, the LSI model during preprocessing would need to be trained over time, with users' posts being sequentially added to the model. We would also need to implement the authors' custom loss function so that the model considers the classification output only if the confidence exceeds a certain threshold. With more time we would also be able to experiment more with the hyperparameters of the model to improve our results.

References:

[1] Bucur, A.M., Dinu, L. Detecting Early Onset of Depression from Social Media Text using Learned Confidence Scores. 2020. https://arxiv.org/pdf/2011.01695.pdf