Detect Depression Symptoms using Methods of Affective Computing

Patrick Schinke[1] , **Anastasiia Sedova**[2]
[12] Ludwig Maximilian University of Munich
patrick.schinke@campus.lmu.de, anastasiia.sedova@campus.lmu.de

Abstract

Depression is a very serious medical illness that may lead to severe outcomes, such as mental breakdown and even suicide. However, it is not only very serious, but also quite common today. It is even often the case that people are depressed without ever realising it and, therefore, do not seek help and treatment from a professional. This study aims at helping to discover symptoms of depression based on what a person says and what a person writes. The main focus lies on text analysis with different methods, such as analysis of the vocabulary a person uses as well as sentiment analysis. The proposed algorithm was tested on the data sets manually collected from Reddit and Shakespeare's tragedy *Hamlet*. Also we realised our algorithm as a web application in order to give an opportunity to all the people to test themselves, which could be a first step on the path to recovery.

Our project is uploaded to Github repository, please follow the link to find it: https://github.com/agsedova/depression_detection

1 Introduction

In Germany about 10,000 people take their own lives every year. Most of them due to untreated depression [9]. This fact alone shows how dangerous this common mental disease can be. Apart from going to a therapist or a doctor there are many depression self test tools in the internet to get a rough diagnosis and to find out whether medical treatment is necessary. However, those tests usually are single-choice based and ask very direct questions such as "Are you in a depressed mood?"[1][8]. The goal of this study is now to build a depression diagnose tool like that as well. However, instead of using the commonly used single-choice questionnaire we built a questionnaire that asks indirect questions and requires a written out answer. This text input is then processed by methods of affective computing to find out whether depressive symptoms are present or not. We hope that this indirect and more open approach leads to more objective and accurate results[2]. Apart from that, there is evidence that people with certain psychological disorders have a different approach to writing about their experiences then people without mental diseases even when they aren't writing about the diseases themselves[15]. That's why it is likely that we will be able to classify the mentally healthy and mentally unhealthy people by just looking at the texts they're writing. Generally, this paper covers our entire working process throughout this project.

2 Theoretical Background

Since the basic idea of our application is to look for depressive symptoms in a user input, we have to think about how this mechanism can work theoretically and how depression can be measured.

Voice Analysis

There is an empirical evidence that the analysis of the human voice can be a useful tool for analysing the psychological and emotional state of someone.

Stephen Silverman, a psychiatrist from the Yale University, observed that the voice of his patients was a good predictor for suicide. He noticed a subtle but distinctive difference in the vocalization of suicidal and non-suicidal but depressed people and between depressed and mentally healthy people. He discovered that suicidal people usually speak at a higher pitch than merely depressed ones which is a byproduct of inner tension in their bodies. So one clue about the psychological state could be the pitch of the speaking voice [1]

Apart from that John Pestian also had a lot of success with detecting depression via voice analysis. He used Machine Learning algorithms to classify people as suicidal, mentally ill but not suicidal and mentally healthy with an astoundingly high accuracy of 85%. He analysed voice characteristics such as fundamental frequency, vocal dynamics and also the length of pauses in the speech itself [20].

In conclusion, it is evident that voice characteristics can be an incredible accurate predictor of depression and suicide intention and can therefore be useful for our application.

[1] Freely translated

[2] We also tried to use audio input. More about that in section 3

Text Analysis

In the previous chapter we presented a way to diagnose depression by using audio input. Now we'll show that text input analysing can also be a method for getting an accurate diagnosis.

Having obtained text input, we can analyse the style of writing and the content of the text which can, according to research, give us clues about the mental state of the writer [14]. One of the symptoms of depression is a continuous low mood and constant sadness [17]. Our hypothesis would be that the writing of depressed people contains more measurable sadness than the writing of mentally healthy people, even if both are writing about the same topic. This is supported by the research of a psychologist Mohammed Al-Mosaiwi, who found out that depressed people use more sad adjectives like *lonely*, or *sad* [15]. Therefore, we believe that the constant sad feelings depressed people have could show themselves in the text, which, in turn, can be measured with the help of sentiment analysis.

The sentiment analysis is a computational method for analysing a data collection in order to get the understanding of emotions, attitudes or opinions expressed by the author of these data. In particular, it allows us to understand, what feelings the author had when saying or writing the corresponding piece of text [12]. The most popular application of sentiment analysis is the analysing of a review on some product for measuring the customers' satisfaction. This task is also sometimes called **polarity detection**, since in most cases the model is designed as a binary classifier with only two outputs: positive and negative [6]. However, sometimes there are more than two labels which could be assigned to the piece of text, but *n* labels, which mark the degree of positivity [23][5]. Thus, much more accurate and fine-grained result could be obtained.

The second task of sentiment analysis is the **emotion recognition**. For this task there is usually a set of predefined emotion labels which the models looks for and extracts from text collection. The emotion recognition has already proven quite useful in domain of affective computing for capturing the human emotions. For example, Zucco et al [31] applies this method to depression recognition and suggests a model which extracts the emotions using the lambda architecture. They used the following list of emotions: *acceptance, anger, anticipation, aversion, courage, dejection, desire, despair, disgust, fear, joy, hate, hope, love, sadness, surprise*. However, no results have been shown in [31], so, we cannot judge how successful this method was.

To the best of our knowledge, there are no experiments where the polarity detection model would be used in affective computing domain. Therefore we decided for this type of sentiment analysis model and realised in in two ways: firstly, using AFINN score and secondly, building and training a neural network. Both of the models are described in details in Section 4.3.

Another way to measuring depression is measuring of absolutist thinking. Absolutist thinking refers to a cognitive distortion which makes us to see our experiences as either perfect or terrible. A study by Mohammed Al-Mosaiwi and Tom Johnstone concluded that absolutist thinking is indeed a good predictor for depression and suicidal intention. They measured this type of thinking by counting specific words such as *always* or *completely* in forum comments and found out that in depression or suidical ideation forums those words were much more common than in other ones. This leads to the idea that this kind of black and white thinking is more common for depressed people [15].

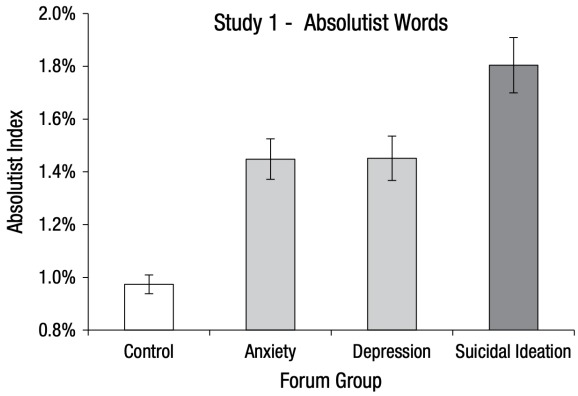


Figure 1: Table taken from Al-Mosaiwi 2018

According to Al-Mosaiwi the above explained absolutist thinking can be reinforced by swear word usage. He states that swear words can be used to reinforce absolute words. For example, the expression "fucking sick of this" is a reinforced and more extreme version of "completely sick of this". Therefore the swear words are also worth looking at when trying to find depressive symptoms [15].

Lastly the usage of pronouns can also be a good tell. Depressed people are using the first person pronoun a lot more often than people from the control group. This apparently comes from the fact that they are more often isolated, lack contact from other people and are very focused on themselves. They are also using the third person pronoun less often. This also can come from the self-centrism mentioned above[14].

Additionally we decided to extract the topics covered in the analysed texts. The main hypothesis is the following: The texts produced by the mentally unhealthy people should presumably contain a number of negative ideas, which could form the negative topics. In order to reveal the topics in text collections we used a method called *probabilistic topic modeling*. Probabilistic topic modeling describes a probabilistic representation of the latent structure of text collections through implicit factors, namely topics – sets of clusters containing words revealing similarity in meaning which is based on distributional similarity within a corpus [13]. Thus, the thematic model acts as a means of generalising and systematising information from large text collections and provides an opportunity to identify hidden structures and implicit dependencies in the data.

So all in all we have found five properties on the style

and content (self-centrism, use of absolute words, use of swear words, sentiment, topic modeling) which may help us to distinguish depressed people's writing from the one of non-depressed people.

3 Speech and Voice Analysis

Now that we've gone over the theory, let's dive into the more practical part.

3.1 Calculation of the average fundamental frequency

In this chapter we'll discuss the our idea to use audio input for depression detection and explain why it hasn't made it into the application:

First we tried to analyse and plot the frequency spectrum of audio files in the .wav-Format using PyAudio and matplotlib. We used the Fast Fourier Transformation to manipulate the input signal to make it easier to analyse.

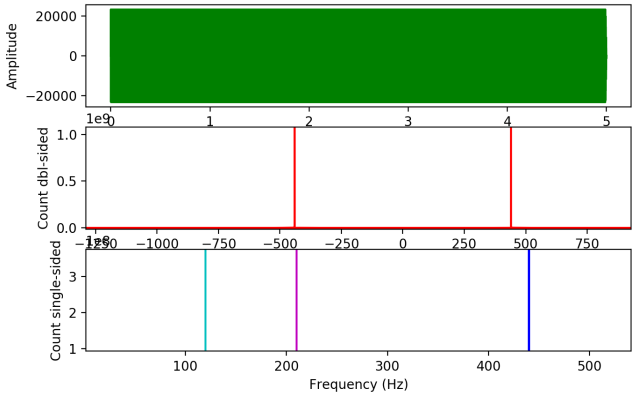


Figure 2: Example: How a single note at 440 Hertz looks like

Our first goal was to be able to identify the different frequencies in the audio input so we can calculate the average frequency and compare it to certain values to identify whether someones voice is higher pitched than normal. The figure above shows the transformation from the raw audio signal before and after applying the fft and in the third plot there's also a comparison of the frequency of the signal with the average fundamental frequency of men (120Hz) and women (210Hz)[27].

3.2 Noise reduction

We then tried to make own voice recordings using the build-in microphones of laptops and stumbled upon the problem that those recordings needed a lot of preprocessing to be analysable. First we needed to filter the noise out of the recordings.

The algorithm we used for this requires two inputs. First we need the audio clip itself which obviously contains the signal we want to keep and the noise we want to cut out. Second we also need an audio clip which only contains the noise we want to remove. The algorithm which is based on Tim Sainburgs NoiseReduce[3] calculates a threshold and compares each frequency to it. The threshold is computed by statistically analysing the the noise clip mentioned earlier. Once again we do the Fast Fourier Transformation on both clips to get the frequencies out of the time based signal.

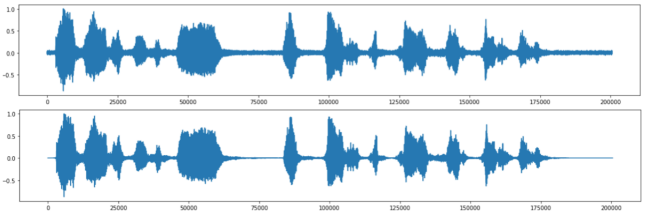


Figure 3: Example: An audio signal before and after the noise reduction. The noise is visible as the baseline between the peaks of both plots. Since there's more noise in the first signal the first plot has a thicker line

However, after losing two team members we had to stop working on audio analysis since, apart from noise reduction, this type of analysis required additional preprocessing such as considering the different voice types[22] which made things more difficult. Therefore, we focused on making the text analysis which required less preprocessing and generally seemed more doable within the scope of this project.

4 Text Analysis

As we have already discussed in Section 2 we have defined five properties which could be measured in order to get an idea about a person's state of mind. Roughly they can be united into 3 groups: lexicon analysis (where we would calculate the self-awareness score and the use of swear and absolute words), sentiment analysis and topic modeling.

4.1 Data Collection and Preprocessing

For testing our algorithm we have collected 3 datasets[4]. Firstly, as mentioned before, we formed four questions which should help us to better understand the state of mind of someone answering these. Later we have used the same questions in our web application. The questions are: *Tell me about your day. How has it been? Tell me about the day before that. Has it been worse or better? If you could turn back time, what would you change? What would you expect from the future?* To get comparable data, we searched for threads on Reddit platform[5] which had those exact questions as a topic and anonymously collected data from the comments (i.e. answers) from the users. For example, on subreddits specialized on basic small talk we found a thread where the users talked about their day. In addition, we found another

[3] https://github.com/timsainb/noisereduce

[4] We couldn't get access to already extracted datasets so we had to make our own

[5] https://www.reddit.com

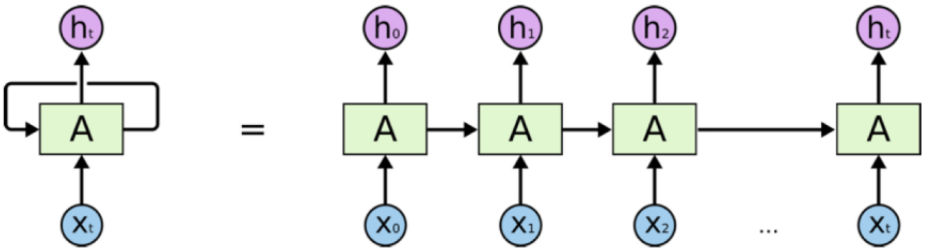


Figure 4: Recurrent Neural Network

similar thread where the question was directly aimed at depressed people. The fact that the second thread was used by more people with diagnosed or undiagnosed depression was not certain but definitely much more likely. So, as a result, we received two datasets: a negative dataset with data collected from potentially depressed people (2777 tokens after pre-processing) and a positive to neutral dataset (1822 tokens after pre-processing). In Appendix 1 you can see the examples as well as the list of the links to threads we have used.

Additionally we have taken the William Shakespeare's tragedy *Hamlet* and collected all words of the main character as a third dataset, in order to analyse him as a person who potentially may be depressed (12285 tokens after preprocessing). Above all, we needed this set to try topic modeling, since this method works only on relatively big amount of data.

As a preprocessing step we eliminated all punctuation symbols and numbers, slit the input text into sentences using the *nltk.tokenize* package from NLTK 3.4.5[6] and performes tokenisation with GenSim Tokenizer[7]

4.2 Lexicon Analysis

As we have already discussed, the depressed people tend to reference themselves more often than the people who do not suffer from any mental diseases. Basically, that means that the depressed people more often use the first-person pronouns (*I, me, my, mine, myself*) and their own name. In order to measure it, first of all, we performed the text tokenisation and part-of-speech tagging using NLTK 3.4.5[6] and after that directly calculated the number of corresponding pronouns a person used. Regarding the counting of speaker's proper name, it turned out to be impossible on Reddit data, since it is not always possible to detect the author's proper name. In contrast, it can be done on Hamlet data and in web application, where the first question was explicitly formulated as *What's you name?* So, the self-reference parameter is then calculated as a sum of first-person pronouns' number and a speaker's name number.

4.3 Sentiment Analysis

We decided to realise the polarity detection sentiments analysis using two methods: AFINN lexicon and neural network. In this section we will go into details of these two methods.

Sentiment Analysis using Afinn Score

AFINN sentiment analysis is a wordlist-based approach for sentiment analysis. Methods of this kind usually have some lexicon, namely a list of words, where each word has some polarity score. The sentiment score of a whole sentence is then calculated basing on the scores of its words. The lexicon for AFINN sentiment analysis, AFINN Lexicon, was first introduced in [18] as a list of words with the corresponding scores in range between -5 and 5, where -5 mean that the word is very negative (e.g. *bastard, bitches*) and +5, correspondingly, that it is very positive (e.g. *hurrah, outstanding, thrilled*). Today there are more than 3300 words in the lexicon, the whole list are upload to the project's Github repository[8]. Using of AFINN Lexicon for sentiment analysis has proven to be a very effective and accurate yet simple technique [26][11][16]. We realised the AFINN Sentiment analysis using *afinn* package for Python. The results are shown in Section 5.

Sentiment Analysis using Long Short-Term Memory network (LSTM)

Another very popular approach for sentiment analysis is sentiment analysis with neural networks. There is a great number of experiments with different types of neural networks, for example, with Convolutional Neural Networks [21]. In our experiment we decided to use Recurrent Neural Networks since they have already become very popular for text analysing and being created exactly for processing sequential data. More precisely, we have used Long Short-Term Memory network (LSTM). However, before we go into details of LSTM, it is worth explaining what Recurrent Neural Networks (RNNs) are in general.

The main idea behind the creation of RNNs was the necessity of working with sequential data, for example, text or movie. Imagine we want our network to predict the next word in a sentence or what will happen in the next movie frame. This could be obviously done only basing on the previous words in the text or previous events in the film. So, in order to perform this task the model has to remember the information about already processed data and build connection to the next units to be predicted. That is exactly what RNNs do.

[6]https://www.nltk.org

[7]https://radimrehurek.com/gensim/

[8]https://github.com/fnielsen/afinn

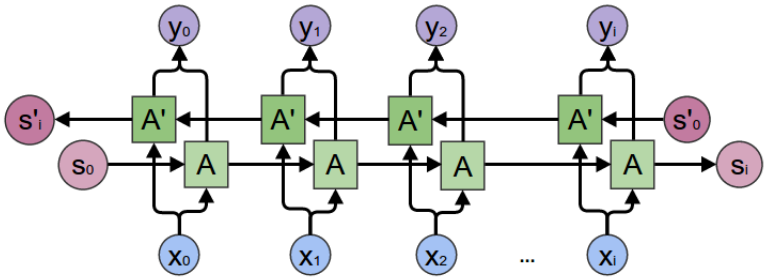


Figure 5: Bidirectional Recurrent Neural Network.
Source: http://www.easy-tensorflow.com/tf-tutorials/recurrent-neural-networks/bidirectional-rnn-for-classification

On Figure 4 you can see a rolled and an unrolled representation of Recurrent Neural Network. It takes as an input a sequence that contains vectors x_t from different time stamps. In our case the sequence would be a sentence, and *t* - different positions in this sentence from 0 to the length of the sentence, vector x_t - an encoded corresponding word which is in this position in the sentence. Each chunk of neural network takes two inputs: the new vector *x* from the input sequence and the output of the previous chunk. Concerning output, there are different ways of building an RNN; for example:

- RNNs that produce outputs at each time step
- RNNs that produce only one single output after processing of the entire sequence. This is how usually machine translation models are built, since the goal is to receive only one output at the end - the translation of the input sentence.
- RNNs that produce no output at all. Such model is used, for example, in encoder-decoder architecture, when the RNN is used only as a first stage of the model for input sentence encoding, while the real output produces only the second part of the model, decoder.

An important property of RNNs is parameter sharing: it offers the possibility to extend the model in order to apply it to examples (here: sentences) of different lengths and generalise across them [10]. In most cases the pieces of information we want our model to extract can be found in different parts of sentence. Compare: *I have birthday on the 18th of November* and *On the 18th of November I have birthday*. If we want our model to answer the question, when the author of the sentence has birthday, it should be able to recognise the relevant piece of information, namely *the 18th of November* regardless of the sentence's part this information appears in. So, RNN can accomplish the task, in contrast to a simple feed-forward network, that has individual parameters for each input feature and, therefore, learns the rules of the language separately at each position in the sentence having no generalising ability. That is the reason why sometimes RNNs are described as multiple copies of the same network, since each member of the output is a function of the members of the previous output produced using the same update rule and same parameters.

However, sometimes it is not enough to take into consideration only the preceding context. Imagine the sentence *I was born in ... . My native language is French.* We want our model to predict the word *France*. Unfortunately, if our model processes the words only in the way they appear in the sentence, it will not be able to fill the gap with correct word. In order to avoid this problem the Bidirectional Recurrent Neural Networks have been created. Their main difference from the simple RNNs is the number of hidden layers: there are two hidden layers of opposite directions which are connected to each other. BiRNN reads the input data, firstly, from the beginning to the end and, secondly, from the end to the beginning. Such strategy is often claimed to be very useful for sequential data and especially text processing, since it provides not only preceding context, but also the following one, what helps the model to learn faster and deliver more accurate results. The architecture of BiRNN is presented in Figure 5.

Unfortunately, there are cases where simple RNNs are not as useful as they might be. Consider the task we have already mentioned before: The prediction of the next word in a sentence. On the one hand, if we have a sentence like *People in France speak ...*, where the word to be predicted (*French*) is placed quite near to the word that contains the helpful information (*France*), simple RNN can deliver quite good results. On the other hand, if we consider a sentence *I grew up in Germany. It is a very beautiful country. I love it very much. I also can speak fluent ...*, we cannot expect our model to easily predict the correct word *German* basing only on the previous sentence, since the necessary information is contained in the sentence further back (*I grew up in Germany*). The further the sentence with relevant information and the sentence with the word to be predicted are, the less probably is that RNN makes a correct prediction. This drawback is successfully overcome by Long Short Memory networks.

Long Short Memory networks (LSTMs) are a special kind of Recurrent Neural networks, which have been designed to avoid the long-term dependency problem [28]. The key difference between simple RNN and LSTM is a structure of the repeating module. In contrast to the RNN, it does not have a single neural network layer, but four; please refer to Figure 6.

The main concept of LSTMs is a cell gate. This is basically the memory of the network, since through the whole processing of the sequence it contains all relevant informa-

tion. The processes of adding and removing the new information is controlled by gates. As output they give out the value in [0,1] reflecting what part of information should be let through. Gates can make the unimportant data be forgotten (if the output value of the gate is close to 0), while the important data be kept (if the output value of the gate is close to 1).

In total, there are tree gates:

- Forget Gate Layer
  A sigmoid layer which decides what information from the cell we will not take into account and will just throw away (namely, what information from the previous steps are relevant for us now). As input, the gate takes the information from the previous hidden state and information from the current input x_t
- Input Gate Layer
  A sigmoid layer which decides which values will be updated (namely, what information should be added from the current step). As input, the gate takes the information from the previous hidden state and information from the current input x_t.
- Output Gate Layer
  A sigmoid layer which decides what part of the cell state we are going to output, namely what the next hidden state would be. As input, it takes the previous hidden state and the current input.

Also there is one more layer, which is called tanh layer. As input it also takes the information from the previous hidden state and information from the current input and passes it through the tanh function. As an output we get a vector of new candidate values squished between -1 and 1.

The overall process can be described as follows: the output of tanh layer is multiplied with the output of input gate, that decides which information should be kept. As a result of this operation we get a candidate value to update the state. The old cell state is meanwhile multiplied with the output vector from forget gate layer, which helps to forget the things to be forgotten. The final cell state is calculated then by pointwise addition of the candidate value got in the first step and the transformed old cell state got in the second step. The output is then the updated cell state with new values that the Neural Network finds relevant. Then the newly modified cell state is passed to the tanh function and multiplied with the output of the gate layer. As a final output we get the new hidden state. Both the updated cell state and the new hidden state are passed to the next time step.

LSTMs have already been widely used for sentiment analysis tasks [7][19]. Sometimes LSTM is used in combination with other neural networks, such as CNN ([2], [29]) or Attention ([30]).

For our experiment we build a model with 2 bidirectional LSTMs and 2 linear layers with ReLU activation function between them. As a regularisation we used dropout and batch normalisation. For the word vector initialisation we used pretrained Glove vectors[9] of 200 dimensions. This data contains 27B tokens trained on twitter data and is often used as initialisation in natural language processing models. The best results have been obtained with stochastic gradient descent as optimizer and cross-entropy loss function as a loss function. The model was implemented on Python using the PyTorch library[10]. For training we have taken the Stanford sentiment analysis dataset SST-5 [24] with 11 334 sentences for training and 1101 sentences for testing extracted from movie reviews. Each sentence has a label from 0 to 4 depending on sentiments class it is assigned to, where 0 means that the sentence is very negative and 4 - very positive. When testing on our data set, we normalised the obtained result in order to get the value between 0 and 1. We called this part of our algorithm LSTMSent.

After 150 epochs we got the accuracy of our model around 72%, which is considered to be quite good for the fine-grained classification models. The detailed results are presented in Section 5.

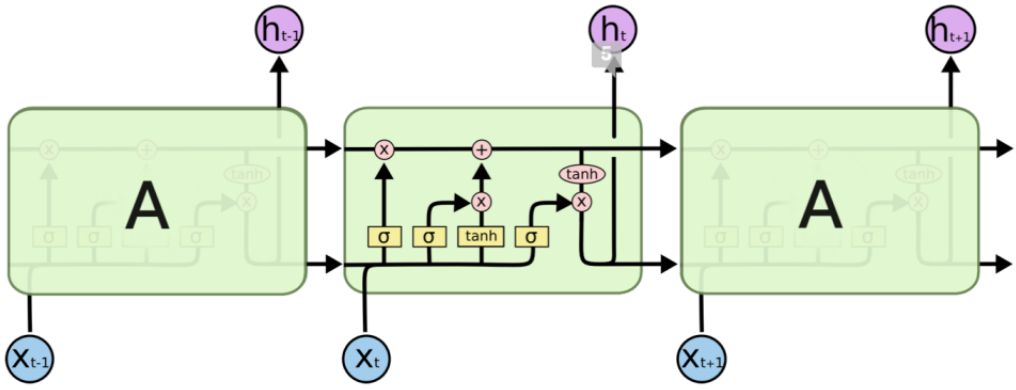


Figure 6: Repeating module of LSTM.

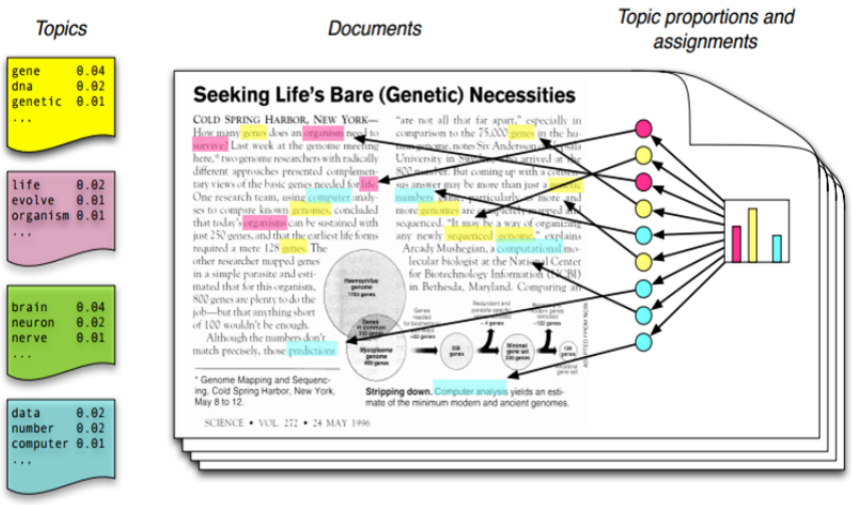


Figure 7: Latent Dirichlet Allocation Model. On the left side of the picture there are topics covered in this text, which are represented by the words they are distributed over. On the right side there is a text distribution over these topics and further, the distribution of the topics over the words. [3]

4.4 Topic Modeling

For topic modeling in our experiment we have chosen Latent Dirichlet Allocation generative probabilistic model (LDA) [4], as it is considered the most popular one. The LDA model, based on Dirichlet prior distribution, provides a soft clustering of the documents, which results in every word and every document of a corpus being allocated to several topics at the same time with particular probabilities. So, there are some topics (unknown in advance), which reflect the meaning that some part of document may contain. Each topic is distributed over the words and each document in corpus is distributed over these topics (i.e. each document exhibits the topics with different proportions). The main goal is to derived both unknown distributions and thus receive the topics presented as a list of words, which are related to these topics with some probabilities. Document can be any piece of text; normally, the length of document is tuned during the algorithm testing. We have taken the most popular approach and decided to define the document as one sentence from text collection.

A very important part of topic modeling is a proper data preprocessing. Our data is already tokenised and clear of non-linguistic symbols; the further preprocessing is lemmatisation and stemming. For these tasks we used WordNet Lemmatizer[11] and PorterStemmer[12] correspondingly using their practical realisation in NLTK 3.4.5[6] package. Also we eliminated all words with length less than 3 letters and all stop-words using the stop-words list proposed in [25] and integrated into GenSim[13] package. Additionally we filtered context-specific stop words (41 in total). Here is an example of 3 random sentences before and after preprocessing:

```
Original sentences:

['tis', 'not', 'alone', 'my', 'inky', 'cloak', 'good', 'mother', 'nor', 'customary', 'suits', 'of', 'solemn',
'black', 'nor', 'windy', 'suspiration', 'of', 'forcd', 'breath', 'no', 'nor', 'the', 'fruitful', 'river', 'in
', 'the', 'eye', 'nor', 'the', 'dejected', 'havior', 'of', 'the', 'visage', 'together', 'with', 'all', 'forms
', 'moods', 'shapes', 'of', 'grief', 'that', 'can', 'denote', 'me', 'truly']

['these', 'indeed', 'seem', 'for', 'they', 'are', 'actions', 'that', 'a', 'man', 'might', 'play', 'but', 'i',
'have', 'that', 'within', 'which', 'passeth', 'show', 'these', 'but', 'the', 'trappings', 'and', 'the', 'suit
s', 'of', 'woe']

['i', 'shall', 'in', 'all', 'my', 'best', 'obey', 'you', 'madam']


 Tokenized and lemmatized sentences:

['inki', 'cloak', 'customari', 'suit', 'solemn', 'black', 'windi', 'suspir', 'forcd', 'breath', 'fruit', 'riv
er', 'deject', 'havior', 'visag', 'form', 'mood', 'shape', 'grief', 'denot', 'truli']

['action', 'play', 'passeth', 'trap', 'suit']

['best', 'obey', 'madam']
```

Figure 8: An example of text sample before and after preprocessing for topic modeling.

The next step is the creation of a bag-of-word model on the data set. In this model, a text is represented as the "bag" of its words. More precisely, each sentence is turned into a vector with the length of vocabulary size by counting how many times each word appears. For example, if we have a text collection of N sentences constructed from a vocabulary of V words, the bag-of-word model represents it as N V-dimensional vectors. During data transformation into bag-of-word model we also filter the extremes - the tokens which appear either too often or too rare. After the number of experiments with different thresholds was decided to filter the tokens, that appear in less than 4 documents and in more than 50% of documents. After that we keep only the first 200 most frequent tokens. The next step was data transformation to a matrix of TF-IDF features in order to evaluate how important each word is to a sentence and topic modeling itself. We decided to extract 5 topics each of which is represented by 10 words.

The specificity of LDA and topic modeling in general is so that this method could be applied only on relatively big

[9] https://nlp.stanford.edu/projects/glove/
[10] https://pytorch.org
[11] https://www.nltk.org/_modules/nltk/stem/wordnet.html
[12] https://www.nltk.org/_modules/nltk/stem/porter.html
[13] https://radimrehurek.com/gensim/

text collections, since otherwise no reliable probabilistic representation would be build. So, for experiment we have used the Hamlet dataset. The results are presented and discussed in Section 5.

5 Results and Discussion

In this section we are going to have a look at the results of our algorithm applied to the data we have collected (please refer to Section 4.1 where we talked about data collection). We have treated them as if all sentences had been written by one person, and now our aim is to define if this person is depressed or not.

To begin with, we defined the thresholds. As thresholds we used the results from [15]: if out of all speaker's words more than 1.2% are absolute and/or more than 0.2% are swear words and/or more than 5% are reference to oneself (first-person pronouns and proper name), it can be a marker that speaker has depression.

Firstly, we tried our algorithm on Reddit negative datatset. The results are the following:

- Self-reference: 213 out of 2776 words (7.6%)
- Cursing words: 6 out of 2776 words (0.2%)
- Absolute words: 37 out of 2776 words (1.3%)
- Afinn score: 0.3
- LSTMSent score: 0.3

So, compared to the thresholds we decided to use, self-reference, cursing words and absolute words values signify that the speaker may have a depression. Afinn score value is close to 0, what is somehow on the boarder between positive and negative value. LSTMSent score is definitely negative. So, to sum it up, the "speaker" has 4 scores out of total 5 "depression scores"[14], what can definitely be a marker of potentially depressive state of mind.

Secondly, we tried our algorithm on Reddit positive dataset. The results are the following:

- Self-reference: 67 out of 1821 words (3.6%)
- Cursing words: 0 out of 1821 words (0%)
- Absolute words: 23 out of 1821 words (1.2%)
- Afinn score: 0.6
- LSTMSent score: 0.45

Here we can conclude that the "speaker" is unlikely to have a depressive state of mind, since 2 of 5 parameters are strictly positive (self-reference and cursing words) and other 3 (afinn score, LSTMSent and absolute words) are close to the edge. This can by easily explained by the type of the data we collected: we have no goal to collect only positive answers, but rather positive *and* neutral. So, this result is exactly what we have expected: the "speaker" is neutral and tends to be a positive person, but he or she definitely has no or very few depressive symptoms.

If we compare the results of both groups we see that indeed the values, which indicate depression, are higher in every category of the negative dataset. There is a much higher percentage of self-references, more curse and absolute words being used while the afinn score and the sentiment analysis are also more in the negative direction. We would need much more data to prove a correlation but still there was a clear tendency.

However, these data sets are not big enough in order for topic modeling. This method we tried exclusively on the third data set, the words of Hamlet in Shakespeare's tragedy *Hamlet*. Here are the results:

```
Topic: 0
Word: 0.114*"sleep" + 0.066*"queen" + 0.039*"poor" + 0.035*"poison" + 0.035*"head" + 0.033*"friends" +
0.029*"foil" + 0.026*"pale" + 0.026*"night" + 0.026*"farewell"
Topic: 1
Word: 0.064*"dead" + 0.064*"bear" + 0.058*"villain" + 0.034*"grave" + 0.033*"knave" + 0.031*"madness" +
0.030*"dust" + 0.030*"elsinore" + 0.029*"hamlet" + 0.024*"gentlemen"
Topic: 2
Word: 0.075*"lord" + 0.070*"fellow" + 0.059*"leave" + 0.043*"answer" + 0.039*"seek" + 0.037*"head" +
0.037*"eye" + 0.037*"month" + 0.036*"form" + 0.031*"woot"
Topic: 3
Word: 0.071*"players" + 0.055*"hold" + 0.052*"right" + 0.051*"pray" + 0.044*"reason" + 0.032*"honest" +
0.031*"dear" + 0.027*"beggar" + 0.026*"fee" + 0.026*"fool"
Topic: 4
Word: 0.108*"follow" + 0.039*"murther" + 0.032*"dust" + 0.030*"play" + 0.030*"revenge" + 0.029*"hear" +
0.025*"need" + 0.025*"noble" + 0.025*"better" + 0.023*"lord"
```

Figure 9: Result of Topic Modeling of Hamlet words from Shakespeare's tragedy *Hamlet*.

The main conclusion is the following: among the extracted topics there is no one, which could be defined is strictly positive. Indeed, one can see that in each of 5 topics there are one or more negative words (i.e. *poor, poison, seek, beggar, dust* etc). Moreover, there is a topic, where the speaker's name (*Hamlet* appears, what means, that the speaker mentions his own name quite often and not in really positive context (among other words in this topic there are *dead, villain, grave, madness, dust*). These both observations can be interpreted as signals of a generally negative point of view on life and self-obsession and thereby of possible depression.

For future work, it would be interesting to test the algorithm on bigger data set in order to tune the thresholds. Also, maybe it would be reasonable to assign some weights to parameters when calculating the final "depression scores": for example, the LSTMSent score seems to be more important for making a resolution, if the person tends to be depressive, than number of absolute words. However, in order to clarify this quesiton, we need to receive the data of true depressed people, which is quite complicated matter because of privacy issues. So, this could be also a line of future work.

6 Proposed Application

The following chapter covers the features of the application that we have developed with the above explained knowledge.

6.1 Graphical User Interface

The main goal of our UI was to present the application and particularly the questionnaire in a non-distracting and

[14]For every category where the result was higher than the threshold we would add a 1 to the overall depression score. Therefore, the highest result would have been 5, the lowest 0

transparent way. We started using PyQt which we weren't satisfied with. We then used Flask for the overall layout and ChartJS for the radar chart in the comparison option. The whole application is structured as follows: First the start screen appears which links to a short explanation of the application itself and to the questionnaire. The latter is build like a classical questionnaire with text input fields and the questions and links to the result screen which offers not only the results of the text analysis but also a comparison of those results to the characters Hamlet, Walter White, Eric Cartman and Donald Trump.

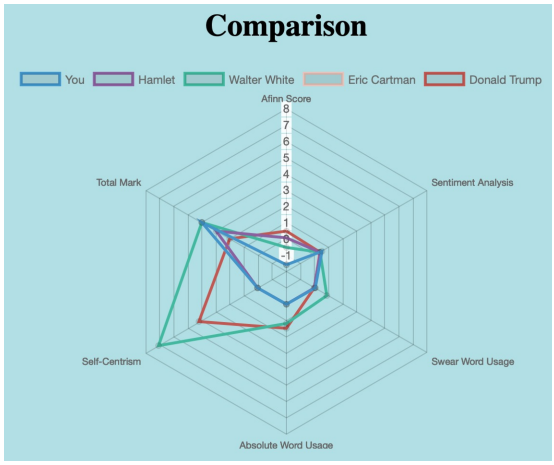


Figure 10: Comparison of the results shown as a radar chart.

6.2 The Questionnaire

The first two questions asked the user about his day and the day before that. The idea behind that was to trigger some basic small talk. The second question was about turning back time and changing the past of ones life. This was supposed to filter out regrets of the user. The fourth question asked what the user would expect from his future. Meant to get his thoughts about his future into a text file for us to analyse. All in all the plan was to use questions which weren't depression related but could trigger some responses that could give a good tell about the emotional state of the user.

6.3 Bonus Feature: Result comparison

As stated earlier, after filling out the questionnaire and analysing it there is an option to compare the results to the ones of specific characters from fiction or reality. To generate those results for the characters we searched for quotes, interviews, monologues or speeches. In the end we used the famous to live or to die-monologue for Hamlet, picked quotes for Walter White and Cartman and used an Interview from Donald Trump where we deleted the interviewers speaking parts. For the topic modeling part we used all the monologues from the entire piece of Hamlet to achieve better results since, as stated earlier, our method of topic modeling requires a lot of data to work well. Its also important to add that we haven't used the original Hamlet version but a translation to more

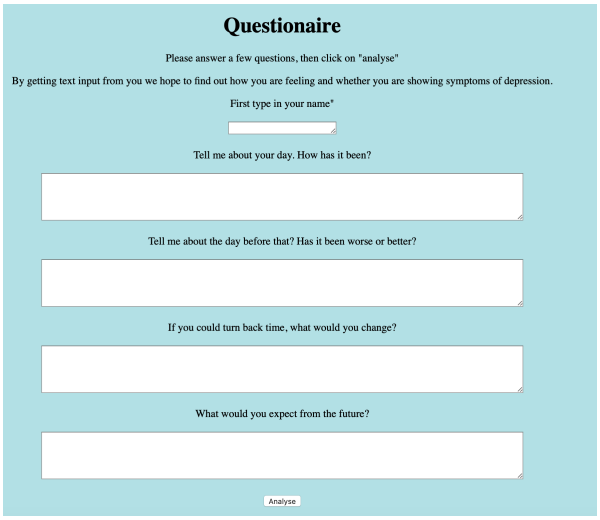


Figure 11: Our questionnaire (with slightly smaller text fields).

modern English to have more linguistic similarity in our data. All in all, the results of the analysis of the characters were pretty unsurprising (f.e. the fact that Cartman uses a lot of swear words)

6.4 (Anonymous) User Study

After finishing the application we tested it on three different people. Our intention was to see whether the application is well usable and transparent and whether the diagnosis is accurate. Since no one of the subjects is diagnosed with depression we expected our application to classify all of them as mentally healthy which it did. In every category all of the subjects had average (equals healthy) results except for the self-centrism category. However, this was expected since the questionnaire asks them a lot about themselves. So a high use of first person pronouns was expected and equally present in all of the subjects[15]. We also asked for feedback and they told us that they indeed experienced the application as being more objective and therefore expect it to be more accurate than a single choice test. In addition they rated the GUI as non-distracting and pleasant.

References

[1] Stephen Silverman et al. "Acoustical properties of speech as indicators of depression and suicidal risk". In: *IEEE Transactions on Biomedical Engineering 47(7) pp.829-37* (2000).

[2] Abdulaziz M. Alayba et al. "A Combined CNN and LSTM Model for Arabic Sentiment Analysis". In: *Machine Learning and Knowledge Extraction*. Springer International Publishing, 2018.

[3] D.M. Blei. "Probabilistic topic models". In: *Communication of the ACM, 55(4)* (2012), pp. 77–84.

[15] For privacy reasons we didn't include the text input of the subjects in the appendix

[4] D.M. Blei, A.Y. Ng, and M.I. Jordan. “Latent Dirichlet Allocation”. In: *Journal of Machine Learning Research* (2003), pp. 993–102.

[5] Siddhartha Brahma. “Suffix Bidirectional Long Short-Term Memory”. In: *CoRR* (2018).

[6] Erik Cambria. “Affective Computing and Sentiment Analysis”. In: *IEEE Intelligent Systems* 31.2 (Mar. 2016), pp. 102–107.

[7] Huiling Chen et al. “Fine-grained Sentiment Analysis of Chinese Reviews Using LSTM Network”. In: *Journal of Engineering Science and Technology Review* 11 (2018), pp. 174–179.

[8] Stiftung Deutsche Depressionshilfe. *Selbsttest*. URL: https : / / www . deutsche - depressionshilfe . de / depression - infos - und - hilfe / selbsttest - offline. (accessed: 01.2.2020).

[9] Heike Friedewald. *Zahlen und Fakten über Depression*. URL: https://www.aok-bv.de/imperia/md/aokbv/presse/pressemitteilungen/archiv/2018/07_faktenblatt_depressionen.pdf. (accessed: 01.2.2020).

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http : // www . deeplearningbook . org. MIT Press, 2016.

[11] Dini Handayani et al. “Sentiment Analysis for Malay Language: Systematic Literature Review”. In: *Preceedings of the Conference: 2018 International Conference on Information and Communication Technology for the Muslim World (ICT4M)” (ITCL-2017)*. 2018.

[12] B. Liu. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge University Press, 2015.

[13] Olga Mitrofanova and Anastasiia Sedova. “Topic Fiction Modelling in Parallel and Comparable Texts (the case study of English and Russian prose)”. In: *Proceedings of the International symposium Internet and Modern Society (IMS-2017): International Academic Workshop “Information Technology and Computational Linguistics” (ITCL-2017)*. 22-23 June 2017.

[14] Mohammed Al-Mosaiwi. *People with depression use language differently – here’s how to spot it*. URL: https://theconversation.com/people-with-depression-use-language-differently-heres-how-to-spot-it-90877. (accessed: 01.2.2020).

[15] Mohammed Al-Mosaiwi and Tom Johnstone. “In an absolute state: Elevated use of absolutist words is a marker specific to anxiety, depression, and suicidal ideation”. In: *Clinical Psychological Science* (2018).

[16] Cataldo Musto, Giovanni Semeraro, and Marco Polignano. “A comparison of lexicon-based approaches for sentiment analysis of microblog postsy”. In: *In DART@ AI* IA* (2014).

[17] nhs. *Symptoms - Clinical depression*. URL: https : / / www . nhs . uk / conditions / clinical - depression / symptoms/. (accessed: 01.2.2020).

[18] Finn Årup Nielsen. “A new ANEW: Evaluation of a word list for sentiment analysis in microblogs”. In: *CoRR* (2011).

[19] Jakub Nowak, Ahmet Taspinar, and Rafał Scherer. “LSTM Recurrent Neural Networks for Short Text and Sentiment Classification”. In: *Artificial Intelligence and Soft Computing*. Ed. by Leszek Rutkowski et al. 2017.

[20] John P. et al. Pestian. “A Machine Learning Approach to Identifying the Thought Markers of Suicidal Subjects: A Prospective Multicenter Trial”. In: *Suicide and Life-Threatening Behavior 47(1)* (2016).

[21] Aliaksei Severyn and Alessandro Moschitti. “Twitter Sentiment Analysis with Deep Convolutional Neural Networks”. In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Association for Computing Machinery, 2015.

[22] Robert Shewan. “Voice Classification: An Examination of Methodology”. In: *The NATS Bulletin 35 pp. 17–27* (1979).

[23] Richard Socher et al. “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. URL: https://www.aclweb.org/anthology/D13-1170.

[24] Richard Socher et al. “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank”. In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642.

[25] Benjamin Stone, Simon Dennis, and Peter J. Kwantes. “Comparing methods for single paragraph similarity analysis.” In: *Topics in Cognitive Science, 3(1)* (2010), pp. 92–122.

[26] Y. F. Tan et al. “Sentiment analysis for telco popularity on twitter big data using a novel Malaysian dictionary”. In: *Front. Artif. Intell. Appl* (2016).

[27] Hartmut Traunmueller and Anders Eriksson. *The frequency range of the voice fundamental in the speech of male and female adults*. URL: https://www2.ling.su.se/staff/hartmut/f0_m&f.pdf. (accessed: 01.2.2020).

[28] Tutorial. *Understanding LSTM Networks*. 2015.

[29] Jin Wang et al. “Dimensional Sentiment Analysis Using a Regional CNN-LSTM Model”. In: Jan. 2016, pp. 225–230.

[30] Yequan Wang et al. “Attention-based LSTM for Aspect-level Sentiment Classification”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2016, pp. 606–615.

[31] Chiara Zucco, Barbara Calabrese, and Mario Cannataro. “Sentiment analysis and affective computing for depression monitoring”. In: *Proceedings of the 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. Nov. 2017, pp. 1988–1995.

Appendix 1. Data Collected

Extract of Negative Dataset:

I went to a family dinner and i was silence the whole time without even realizing it. It took longer than I thought but I survived. We got back home, I changed back to my pajamas and I've been laying in bed ever since. But it was nice! I ate some good food which made the whole thing worth it. Felt pretty good about myself because I think I stopped some little kid from getting beat up. For a moment at least. Actually left my house for the first time in 2 WEEKS! My husband made a lovely meal. I'm pleased to end my weekend on a high note after being a guinea pig for the pat month to find a medication that doesn't make me suicidal. Edit: thank you for asking. Little things like this, really make my day! not great. i pretty much wasted it, then proceeded waste another few hours here on reddit and reading that super depressing thread on the front page. Woke at 5.30pm after meaning to stay awake all night to fix my sleeping patterns, no food in house so ordered pizza at midnight but appetite gone by the time it arrived. I've been half watching really quite boring documentaries, half dootling about on the Internet, and I might get up and make a cup of tea in the next hour. Starting back uni tomorrow after missing a good 60 per cent of last term's classes, so I'm trying to psyche myself up to get some momentum going again Took a depression nap but woke up with cat I'm watching for a few weeks in my arms... So that's nice. I tried. I got up, ate breakfast, made plans for dinner with the fam, and laid down. But I still managed to shave, brush my teeth, shower, eat lunch, and even went on a jog! I'm still a kissless virgin at 25, I work a boring office job, and I still struggle with depression, ADHD, and social anxiety issues. But I manage to work for my dad 40 hours a week, I have medication, a psychiatrist, a counselor, and no debt. And that's after making four suicide attempts in the span of a year. I learn to celebrate the little victories, which then turn into major accomplishments. I'm numb as I'm writing this, but I am doing the best I can. Well, I'm still here. So bad. My grandma died Already drunk and it's 11am YEW Average and bleak.

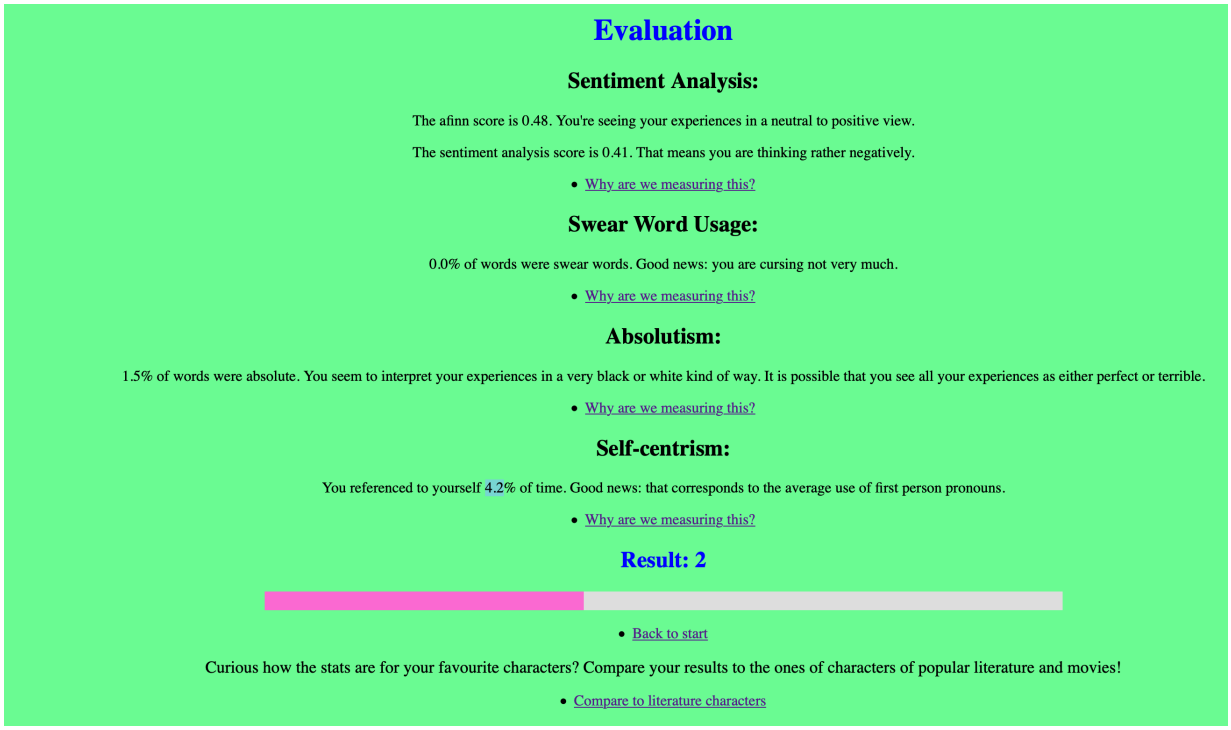
Extract of Positive Dataset:

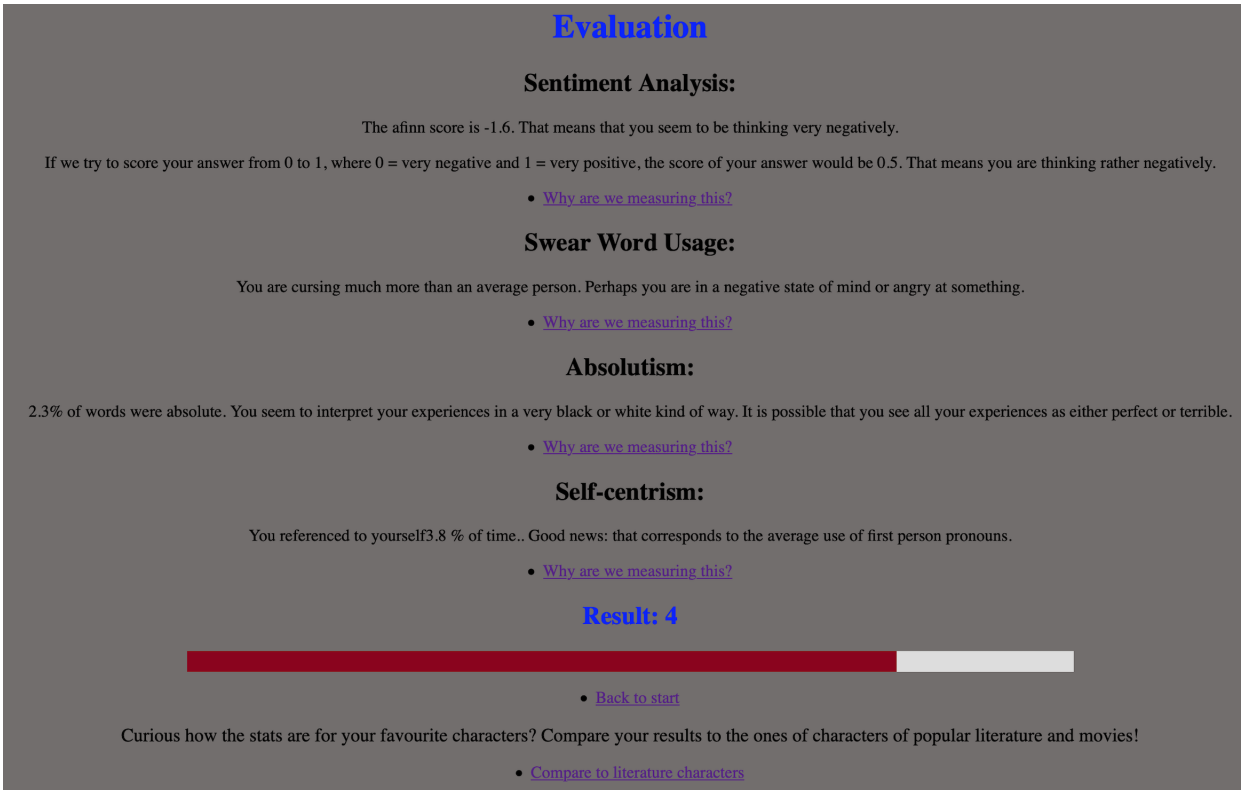
Gym: Bad at first, but got a lot better. Like I always say, an advantage of having bad hearing is a bigger likelihood of weird out-of-context things. Oh, also I was worried about math. Math: I had my midterm test. I didn't know what I was doing, but I feel kind of good about it. Later, it was boring. Finally, it was kind of fun once everyone finished. My day has been mundane, but fine. I'm watching my brother until my mom gets home from work, which is nothing too out of the ordinary for me. Aside from that, I'm trying to work on watching everything on my clogged-up-beyond-repair DVR. It's frankly beyond ridiculous how many movies I've recorded and then promptly didn't watch. After I finish that I'll get some reading done. Already had dinner, which was soup with lots of hot sauce added in. As for my week overall, I've been having a highly eventful one. We're searching for a rental to move into in order to fix our house up to sell. This is my childhood home and so it'll be surreal to live in a different house, finally. Say, what music are you listening to? I could go for something new. Good. I went to counseling today and talked about enjoying activities and received good advice to make myself grow and be happy so I don't need to depend on (often one sided) romantic relationships. Also cleaned off mud from new pink shoes I put on to feel good about myself. All in all a pretty good day. Nice job on the essay! I overslept today and went to the pictures to see Star Wars VIII. Then found out we have an assignment due tomorrow evening. So a pretty eventful day, even if it started way too late for me. My day's pretty good. Hows the book so far? I did band instead of choir growing up and never tried choir, is it fun? I was never a singing kid. Pretty good, how bout yours Pretty good! How has yours been My 3 week old cough is still here but otherwise ok Long but not bad typical Wednesday shockingly good

Links to source threads:

- How's your day been?
  https://www.reddit.com/r/CasualConversation/comments/7pivyg/hows_your_day_been/
  https://www.reddit.com/r/AskReddit/comments/b96ylm/people_of_reddit_hows_your_day_been/
  https://www.reddit.com/r/AskReddit/comments/7qgb56/depressed_people_of_reddit_how_was_your_day/
- If you could turn back time, what would you change?
  https://www.reddit.com/r/AskReddit/comments/afhlve/if_you_could_turn_back_time_a_year_back_what/
  https://www.reddit.com/r/AskReddit/comments/br3gdi/reddit_users_if_you_could_turn_back_time_in_your/
  https://www.reddit.com/r/depression/comments/b4vt7j/ever_wish_you_could_turn_back_time/
  https://www.reddit.com/r/depression/comments/8fykan/i_wish_i_could_go_back_in_time_and_redo_my_life/
- What would you expect from the future?
  https://www.reddit.com/r/AskReddit/comments/8b5kut/what_would_you_expect_by_the_year_2030/
  https://www.reddit.com/r/depression/comments/784ojm/i_cant_picture_my_future_anyone_else/

Appendix 2. The result screen of the proposed application. Providing a precise evaluation with different subcategories and linking theoretical explanations for each of those.





Appendix 3. Work Distribution

Patrick Schinke:

My work was based around the research and the frontend-development of the app. I did most of the research regarding the question how to measure depression and how to distinguish between text or audio input from depressed and from mentally healthy people. In addition to that I was fully responsible for designing the Graphical User Interface of the app. I also was involved in the development of the audio analysis and the word counting algorithms.

Anastasiia Sedova:

I was fully responsible for the backend-development of the application, namely lexicon analysis and sentiment analysis, as well as topic modeling. I also did research for the application of these methods in affective computing domain. In addition I was involved in general research of depression recognition questions and in the development of the audio analysis and noise reduction, which finally was not included in final version.