UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Yevhen Tyshchenko

Depression and anxiety detection from blog posts data

Master's Thesis (30 ECTS)

Supervisor: Kairit Sirts, PhD

Tartu 2018

Depression and anxiety detection from blog posts data

Abstract:

Depression and anxiety affect the life of many individuals and if the diagnosis is not stated in time it could lead to considerable health decline and even suicide. Nowadays, mental health specialists, as well as data scientists, work towards analyzing social media sources and, in particular, publicly available text messages and blogs to identify depressed people and provide them with necessary treatment and support. In this work, we adopt an experimental data collection approach to gather a corpus of blog posts from clinical and control subjects. Ill people are considered as clinical subjects while control subjects refer to healthy individuals. We inspect the latent topics found in collected data to analyze the blog' content according to themes covered by blog authors. We experiment with various text encoding techniques such as Bag-of-Words (BOW), Term Frequency-Inverse Document Frequency (TFIDF) and topic model's features. We apply Support Vector Machines (SVM) and Convolutional Neural Network (CNN) classifiers to discriminate between clinical and control subjects. Additionally, we explore the classification performance of CNNs trained on blog post texts of different size. The best accuracy and recall scores of 78% and 0.72 respectively were obtained with a Convolutional Neural Network (CNN) classifier initialised with pretrained GloVe word vectors.

Keywords: Text classification, natural language processing, machine learning, neural networks, depression, anxiety

CERCS: P170 Computer science, numerical analysis, systems, control

Depressiooni ja ärevuse tuvastamine blogipostituste andmete baasil

Lühikokkuvõte:
Depressioon ja ärevus mõjutavad paljude inimeste elu ja kui diagnoos ei ole õigeaegselt määratud, võib see kaasa tuua märkimisväärseid terviseprobleeme ja isegi suitsiidi. Tänapäeval uurivad vaimse tervise spetsialistid ja andmeteadlased meetodeid, kuidas sotsiaalmeedia ja eriti avalikult kättesaadavate tekstisõnumite ja blogitekstide analüüsimise abil depressioonis inimesi tuvstada ja pakkuda neile vajalikku ravi ja toetust. Selles töös kogume eksperimentaalse andmestiku avalikult kättesaadavatest blogipostitustsest, mis koosneb nii kliinilisest kui ka kontrollgrupi postitustest. Kliiniline grupp koosneb autoritest, kes kannatavad depressiooni ja/või ärevuse all, kontrollgrupp koosneb tervetest isikutest, kes oma blogis kirjutavad depressiooni ja ärevuse teemadel. Töös leiame kogutud andmetes sisalduvad latentsed teemad ja analüüsime blogipostituste sisu vastavalt blogi autorite poolt kajastatud teemadele. Katsetame mitmete teksti kodeerimismeetoditega nagu sõnahulk (BOW), TFIDF ja teemamudelist tuletatud tunnused. Treenime

tugivektormasinatel (SVM) ning konvolutsioonilistel närvivõrkudel (CNN) põhinevaid klassifikaatoreid kliinilisse ja kontrollgruppi kuuluvate autorite eristamiseks. Lisaks uurime, kuidas mõjutavad erineva pikkusega blogipostitused CNN'i klassifitseerimistäpsust. Parimad täpsuse ja saagise skoorid vastavalt 78% ja 0,72 saadi konvolutsioonilise närvivõrgu (CNN) klassifikaatoriga, mis oli initsialiseeritud eeltreenitud GloVe sõnavektoritega.

Võtmesõnad: Tekstide klassifitseerimine, loomuliku keele töötlus, masinõpe, neurovõrgud, depressioon, ärevus

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction

According to World Health Organization[1] (WHO) depression is common worldwide mental disorder that affects more than 300 million people regardless their age. The long-lasting depression could lead to suicide if the depressed individuals are not provided with specialized help in time. In addition to this, individuals also suffer from anxiety (i.e. anxious distress or generalized anxiety disorder) which is often confused with depression but has slightly different symptoms.

According to Diagnostic Criteria for Major Depressive Disorder (MDD) and Depressive Episodes[2] the symptoms of a clinical depression are depressed mood or a loss of interest in daily activities that lasts more than two weeks, weight loss or gain, change in sleep cycle, loss of energy, loss of concentration and activity, indecisiveness and suicidal thoughts. If these symptoms also cause considerable distress or harm any of the important life areas (social, occupational, studying etc.) this can be an important clue for making a clinical diagnosis. The symptoms of Generalized Anxiety Disorder (GAD) are excessive worry, restlessness, being easily fatigued, having troubles with concentrating, irritability, sleep disturbance and muscle tension. Despite the fact that MDD and GAD have some common symptoms such as sleep disorders, fatigue and trouble concentrating depressed people tend to move slowly and have dulled reactions. Alternatively, anxious people are more keyed up and afraid of their future[3]. According to Anxiety and Depression Association of America (ADAA) [4] roughly half of the people diagnosed with clinical depression are likely to be diagnosed with anxious distress and these two disorders can cause each other. According to WHO, there is a number of factors that interfere the depression detection and treatment such as lack of professional specialists, social shame factor, inaccurate diagnosis and so on[1]. Although, apart from the obstacles mentioned above, people also do not get required treatment if the depression detection accuracy is low or their diagnosis remains covered for some reason. Thus, the depression detection is an important issue that can help people who suffer and literally save lives.

The previous research on depression detection and its indicators is wide and complex in terms of the applied approaches, data sources, evaluation methods and even ethical perspective. The most widely used assessment methods are questionnaires, clinical interviews, and self-assessment tests including The Hamilton Rating Scale for Depression [2] (HAM-D), The Beck Depression Inventory [3](BDI-II) and The Patient Health Questionnaire [4]. Their idea is to ask an individual a list of questions or run an interview, score each answer or statement, count the final score and provide a conclusion based on it. However, they all require a personal appointment with a specialist thus putting

[1] http://www.who.int/en/news-room/fact-sheets/detail/depression

[2] http://www.psnpaloalto.com/wp/wp-content/uploads/2010/12/Depression-Diagnostic-Criteria-and-Severity-Rating.pdf

[3] https://www.psycom.net/anxiety-depression-difference

[4] https://adaa.org/understanding-anxiety/depression

much responsibility on a depressed individual and his or her self state understanding. These methods work well on individual's level detection and cannot be scaled to those who got used to low mood and who is ashamed of going to specialist and telling their private thoughts and feelings. Thus, it is important to use other approaches and data sources to identify more ill people and provide them required help and treatment. The diagnosis alone would not not cure anyone but it might at least convince some people to attend the mental health specialist. In other words, they will have proofs that the feelings they experience are not normal and gain more courage to explain them. Finally, the extensive research on depression and anxiety prediction would lead to the introduction of internet-based programs i.e. online mental health checkers where individuals will just provide their messages and receive an initial diagnosis.

Nowadays, people spend much time on the Internet and tend to share their thoughts, stories and quite personal feelings on the web resources such as social network insights (i.e Facebook, Twitter and Instagram), blog platforms, forums etc. It has inspired mental health specialists to take advantage of the available textual data and analyze it to develop novel mental health detection approaches and determine if it is possible to distinguish healthy and ill individuals. Such research is interdisciplinary and incorporates psychiatrists, linguists and data scientists, and is closely related to Natural Language Processing (NLP). In particular, the depression detection task can be stated in a variety of ways depending on the aim: binary text classification on depressed and healthy people, onset depression detection, depression severity prediction, multiple mental problems' diagnosis, and multi-modal depression detection. Each problem is approached differently and uses not only textual information but often includes features from multiple modalities such as audio, video, and each particular subjects' meta data.

The research conducted in this work obviously requires the text data so we briefly describe the data collection approach as follows. The chosen data source is blog provider platform Blogspot [5] where people post their text messages describing their life or covering some dedicated topics. These messages are usually retrieved by means of scraping scripts in an automated way. This process is aimed to gather the dataset containing text messages of healthy and ill individuals. Furthermore, the analysis of scraped blogs often includes the discovery of topics within these text documents focused and their manual evaluation. The topic analysis provides some basic intuition about the obtained data and reveal hidden text similarities and patterns.

Topic modeling has been also applied for depression identification and mental health monitoring. In addition to being an effective tool in computational linguistics, topic model reduces the input textual data feature space to a fixed number of topics people discuss in their narratives. This makes topic models applicable for the text classification tasks as their output has a considerably lower dimensionality than the initial input text. The topics obtained with topic model can be interpret well by humans and contain

[5] www.blogspot.com

valuable information about the language use of their authors [5]. Several papers have attempted to detect depressed individuals using the information extracted from topic model [6], [7] or applied topic modeling to remove the unwanted topic bias of input data [8].

This work is focused on the detection of depression and anxiety using the textual data extracted from blog posts. We will propose a clinical and control data collection method, analyze the topics covered within the extracted text data by means of topic modeling and experiment with classification methods. As soon as the collected data vary in length considerably which might affect the performance of CNN model, our research will try to estimate the quality of extracted data and assess the classification performance depending on blog post length. In other words, we will treat the post length as a hyperparameter to see what impact doest it have on the overall performance.

The goals of our research are:

- collect the publicly available media messages of healthy and self-diagnosed individuals
- evaluate the extracted data
- analyze the topics covered within the collected data and experiment with multiple feature extraction methods to see what results can we achieve treating them as features
- apply machine learning methods such as SVM, RF and CNN to predict depressed and anxious individuals by their blog posts and perform subject- and post-level evaluation results
- determine how post length affects the classification performance

The work is structured as follows. In section 2 we cover the related work on how depression affects the language people use to express their thoughts and provide a short overview of articles on depression detection as well as machine learning methods that have been applied there. Section 3 provides a detailed technical overview of data collection, text representation and classification methods we experiment with in this work. Here we also cover the technical background of topic modeling and list the evaluation metrics. Section 4 describes the data this work is based on and covers the data collection method as well as the preprocessing step. In section 5 we provide the topic model description, discuss the parameter search procedure and results. Section 6 provides and discusses the experimental setup and classification results for chosen classifiers. Finally, in section 7 we provide the conclusions and propose some possible further improvements.

2 Related Work

This section describes the previous research of language aspects of depression and anxiety. We shortly cover the word count based approaches and tools, and describe the text classification methods applied in mental health diagnosis prediction. Furthermore, this section provides an overview of machine learning approaches as well as data sources used in these previous works.

2.1 Language aspect

The previous research of depression has shown that depression affects the language people use and, in particular, how it differs from the healthy individuals. It has been observed that depressed people often use first person singular pronouns, word "I", verbs in past tense and absolutist words [9], [10], [11], [12]. Pennebaker et al. (2003) performed a comprehensive work [13] on word use as a sign of psychological and physical health change which was aimed to find a connection between words people use and their mental and physical health. This research lead to the development of software and tools for language feature based analysis and prediction such as Linguistic Inquiry and Word Count [14] (LIWC) and Differential Language Analysis Toolkit (DLATK) [15].

LIWC program reads the input text and counts the words that reflect various emotions, thinking styles and social concerns. It has a predefined number of word categories developed by cross-domain researchers and specialists but also allows adding new custom categories. LIWC categorization has been also widely used to analyze social media posts and detect suicidal and self-harm ideation, and other social risk factors [16]. Although, LIWC based features have proved their efficiency the LIWC software is a commercial language-dependent product which limits its usage.

2.2 Machine learning

This work belongs to Natural Language Processing (NLP) field and text classification as a particular task. Text classification task is one of the most researched tasks in NLP. It is aimed on predicting the dependent target variable (class label) using the features extracted from text messages which are treated as independent variables. In general, the previous research in text classification has been related to various domains where machine learning and deep learning methods showed amazing results mainly because of computational power available these days. Moreover, text classification task has been one of the most competed task in competition platforms such as Kaggle[6][7] and lead to the introduction of new text analysis methods and models. One of example

[6]https://www.kaggle.com/

[7]https://bicepjai.github.io/machine-learning/2017/11/10/text-class-part1.html

applications is movie review sentiment classification [17] where authors applied Naive Bayes and Support Vector Machines (SVM) to categorize the movie reviews as positive and negative.

The research then moved towards the extraction of population-based health information from social network insights — an ever-growing source of data. Particularly, M. De Choudhury et al. (2013) used Twitter messages to estimate the depression on a population level in US also applying SVM as a machine learning method [18]. In general, Twitter social network has become a popular source of data for similar analysis. Additionally, another scientific research experimented with N-gram language models to predict post-traumatic stress disorder, depression, bipolar disorder, and seasonal affective disorder [19]. Similarly to the previous paper, they analyzed Twitter messages and showed that language models outperformed LIWC and claimed that they model the language better than count-based approaches.

On the other hand, the alternative feature engineering approaches are aimed on building classifiers on top of the topic-based features that compress input texts to a fixed number of non-overlapping topics. The other research group integrated the hidden topic features obtained from Latent Dirichlet Allocation (LDA) topic model to classify short and sparse texts [6]. Moreover, Blei et al. (2003) in his paper [5] argued that SVM classifier with LDA-based document features performed better than simple bag-of-words features. They also pointed out that LDA can be considered as a dimensionality reduction technique that provides meaningful results which correlate with the underlying text data structure and is often well interpretable. The other researchers have widely used LIWC categories in natural language analysis and, for instance, augmented LIWC features with LDA in [20] to predict neuroticism and depression in students and showed promising results. Resnik et al. (2013) has claimed that topic features has improved the precision and preserved recall to decrease. Despite the fact that we have both long and short text pieces we also applied topic modeling to see how various classifiers will work on this compressed data representation and analyze whether the topics discussed by clinical and control subjects are similar or not.

Another interesting shared task was proposed on CLPsych 2016 workshop [8] where participants had to predict the urgency of posts of a youth mental health forum between 4 severity categories. The dataset contained posts from the Australian website ReachOut.com[9] which were labeled by specialists. The winning system by Kim et al. (2016) experiments with two types of text feature representations: TFIDF and post embedding vectors [21]. The best reported accuracy was obtained with an ensemble classifier constructed of multiple maximum entropy models with post- and sentence-level TFIDF features and post-level embeddings.

This research was mostly inspired by the work "Predicting depression for Japanese

[8] http://clpsych.org/

[9] https://au.reachout.com/

blog text" by Hiraga [8]. Her aim was to predict clinical depression for Japanese bloggers using various machine learning approaches such as Naive Bayes, Logistic Regression and Support Vector Machines. She also performed text feature engineering and stated an impressive accuracy of Naive Bayes classifier with the selected lemmas of 95.5%. The data was scraped from blog provider websites that has a "depression" category where people provided their self-stated diagnosis as well as the life experience of living with it. Particularly, she also made an attempt to extract a control group such that the healthy individuals were of the same age as their ill counterparts. Moreover, she removed the bias towards the "depression" topic to ensure the prediction is made regardless the shift towards this topic. Alternatively, in our work we followed another blogs' selection strategy and did not remove topic bias and, instead, kept them as they are. Our main data collection goal was to retrieve blogs such their authors write about depression regardless the class they belong to. The applied data scraping method is briefly explained in 4.1.

3 Technical background

This section incorporates technical description of methods and approaches applied for this natural language processing task including web data retrieval, numerical text representation, probabilistic topic modeling, and text classification methods.

3.1 Web data retrieval

Web data information retrieval field has developed considerably since the rapid growth of Word Wide Web pages and websites. The existing approaches could be grouped into two categories by their underlying work principle ([22], [23]): *tree-based approaches* and *web wrappers*. The third category is hybrid systems that incorporate benefits of the aforementioned approaches.

The first category is *tree-based approaches* that consider the DOM web page representation which is basically a labeled rooted tree hierarchy over a mixture of text and HTML tags. This particular web page representation motivates the usage of tree-based algorithms and mechanisms for addressing specific page tree nodes containing desired data. This is usually performed by XPath queries to a single page element or a group of similar page elements enclosed between HTML tags.

The second category is *web wrappers* that in terms of web data extraction is often described as a process aimed to find, extract, and transform an unstructured target data for further analysis by computer program in automated or semi-automated way. This approach can be decomposed into the following three steps:

1. Initialization — the wrapper is created;
2. Execution — wrapper runs and collects the data;
3. Maintenance — if the data source structure changes then the respective wrapper should be also tuned to handle it.

The script for data collection designed and used in this work falls into the category of web wrappers. We provide the high-level explanation of the applied web data retrieval method in Section 4.1.

3.2 Text representation

Most of the natural language processing tasks require input text in numerical representation rather than raw words meaning that each word must be encoded in order to be processed by machines. This section describes text encoding methods applied in our research.

3.2.1 Bag-of-words

One of the most popular and straightforward approaches to encode text is to use a bag-of-words (BOW) text representation. This representation encodes words with their frequencies and outputs a matrix where rows are documents and columns represent unique words. The word importance is just its frequency which is the main weakness of this representation. In other words, frequently used words such as "the" which are not discriminative across the collection of the documents will get higher weights thus hiding the less frequent but more informative key words. BOW text representation can be also constructed with binary feature vectors that represent only the presence or absence of each particular word from vocabulary in text document.

3.2.2 Term Frequency-Inverse Document Frequency

Term frequency-inverse document frequency (TFIDF) is applied to resolve the issue that less frequent but informative words are outweighed by more frequent words. TFIDF is basically a numeric statistic that describes the importance of a word with respect to each document in a corpora. The term frequency (TF) is be calculated as follows:

$$tf(w, d) = \frac{n_w}{\sum_k n_k},$$

where n_w is the number of times the word appeared in a document d and $\sum_k n_k$ is the total number of words in a document.

The inverse document frequency (IDF) reduces the weights of frequent words and strengthens the weight of rare words if they do not appear often in other documents except the current one, and can be calculated as follows:

$$idf(w, D) = log\frac{|D|}{|\{c \in C : w \in c\}|},$$

where $|D|$ is the corpus size i.e. the number of documents and $|\{d \in D : w \in d\}|$ is the number of documents where the word w appears. Finally, TFIDF itself is a product of the two aforementioned frequencies:

$$tfidf(w, d, D) = tf(w, d) \cdot idf(w, D).$$

The word will be ranked with greater TFIDF value if it is present in a particular document and absent in other documents.

3.2.3 Word vectors: GloVe

The alternative approach is to use distributed word representations i.e. word embeddings. These representations can be obtained by various methods including neural networks [24].

On of the most popular text encoding scheme is Global Vectors for Word Representation (GloVe[10]). The approach was proposed by Pennington et al. [25] and their method is based on word co-occurrence matrix which holds the information of how often words appear together in a corpus. The authors also prevent words that rarely co-occur together to be underestimated and overshadowed by frequently co-occurring words, thus weighting infrequent words more heavily than frequent ones.

The relationship between each pair of word vectors in GloVe model is defined as follows:

$$w_i^T w_j + b_i + b_j = logX_{ij},$$

where $w_i^T w_j$ is a scalar product of i-th and j-th word, X_{ij} is their co-occurrence frequency, and b_i and b_j are the bias terms that prevent initial value to be zero. The final GloVe model is defined as follows:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T w_j + b_i + b_j - logX_{ij})^2.$$

Here f is the proposed weighting function:

$$f(x) = \begin{cases} (\frac{X_{ij}}{x_{max}})^\alpha, & X_{ij} < x_{max} \\ 1, & otherwise \end{cases}$$

According to Kim [26]: *"Initializing word vectors with those obtained from an unsupervised neural language model is a popular method to improve performance in the absence of a large supervised training set"*. Since we want to experiment with a neural text classification model and our labelled training set is relatively small we have experimented with GloVe word vectors to improve the performance of the neural models.

3.3 Topic modeling

Nowadays, the amounts of written text information often do not fit the computational capacity and make scientists apply text mining methods to discover useful hidden text structures or similarities. One important task in natural language processing field is detecting the between document similarities based on ideas and topics covered within corpus of texts. Topic modeling is an unsupervised text mining approach that discovers these similarities in a text or a corpus. Topic modeling itself refers to a set of statistical algorithms that explore the topics over the collection of documents and helps to annotate these documents according to detected topics.

[10]https://nlp.stanford.edu/projects/glove/

3.3.1 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a generative probabilistic model that models a text corpus and is helpful in discovering its underlying topic structures [5]. The main idea behind LDA is that each document is represented as a probability distribution over latent topics. Every topic in this model is treated as a distribution over words existing in the collection of documents. The LDA model could be decomposed into two steps: *model definition* and *model inference*.

Model generation The model definition explains how the model works from a probabilistic perspective and how document-to-topic and topic-to-word distributions are drawn from two Dirichlet distributions with α and β parameters. The *Dirichlet distribution* is a multivariate probability distribution that represents the probabilities x_i of $K > 2$ distinct categories such that

$$\sum_{n=1}^{K} x_i = 1, x_i \in (0, 1)$$

Then, for each word position in document choose topic and, finally, given the topic distribution pick the word for this position. LDA model is often represented using plate diagram (Figure 1) with the following notations:

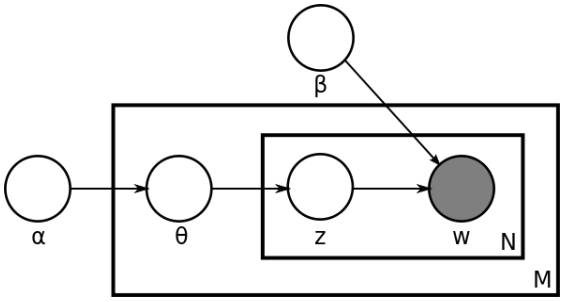

Figure 1. Plate notation of the LDA model.

- The number of topics K is fixed and defined.
- The number of all documents in a corpus D is M.
- ϕ_k is the k-th topic-to-word distribution over a fixed vocabulary where $1 \leq k \leq K$.
- Θ_m is the m-th document topic distribution where $1 \leq m \leq M$.
- z_m is the topic assignment for the m-th document, and $z_{m,n}$ is the topic assignment for the n-th word in the m-th document.

- $w_{m,n}$ is the n-th word in the m-th document.
- α, β are hyperparameters of a Dirichlet distribution.

Then, the generative process can be described as follows:

1. $\forall$ document $d_i \in D$ draw $\Theta_{d_i} \sim Dirichlet(\alpha)$
2. $\forall$ topic $k \in K$ draw $\phi_k \sim Dirichlet(\beta)$
3. $\forall i,\, j : 1 \leq i \leq M, 1 \leq j \leq |d_i|$ where $|d_i|$ is the number of selected words in document d_i draw:
    - topic $z_{i,j} \sim Multinomial(\Theta_i)$
    - word $w_{i,j} \sim Multinomial(\phi_{z_{i,j}})$

The *multinomial distribution* is often referred to as *categorical distribution* in natural language processing context. It is a discrete probability distribution that represents the outcome of a random variable that can take one of N categories.

Model inference Now the task is to infer word-to-topic assignments $z_{i,j}$, document-to-topic distributions Θ_i and corpus topic distributions ϕ_k. Blei et al. [5] describes a *variational inference* approach to estimate the posterior distribution of the hidden variable w — gray circle on figure 1.

3.3.2 Model evaluation

There exist various methods and measures for automated topic model evaluation such as model perplexity [5], likelihood score [27] or topic coherence on held-out documents [28]. In practice, however, the best model in terms of these measures could be semantically poor and contain senseless topics[11]. Thus, we decided to evaluate the topic models manually according to the most frequent words of each topic. In addition to this we has visualized the transformed documents using *t-SNE* algorithm[12].

3.4 Classification

Classification task is usually described as the task aimed on predicting the label i.e. making a group assignment for each given element based on some classification rule [29]. In our work the task is binary text classification meaning that we have two classes –

[11] https://www.quora.com/What-are-good-ways-of-evaluating-the-topics-generated-by-running-LDA-on-a-corpus

[12] https://lvdmaaten.github.io/tsne/

ill and healthy people – and we want to train a classifier to predict either one of these labels for each input text. In this section we provide a list of applied classifiers along with their technical descriptions.

3.4.1 Support Vector Machines

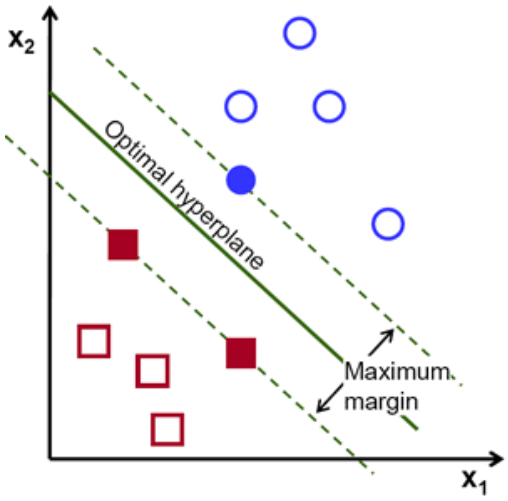

Figure 2. SVM illustration for two-dimensional case.

Support Vector Machine (SVM) [30] is a supervised machine learning method that can be used for both classification and regression tasks. The main idea behind it is to divide a linearly separable data into two classes with maximum between-class distance. If we consider two-dimensional example illustrated on Figure[13] 2, SVM finds an optimal line that lie as far from the nearest class data points as possible. The dashed lines are called support vectors.

In general, SVM finds an optimal hyperplane in high-dimensional space that separate the input data with the maximum margin between this hyperplane and nearest training data points of any class. If the input data is not linearly separable a kernel function can be applied to map the data into higher-dimensional space to make it linearly separable. The most popular non-linear kernels are: polynomial kernel, Gaussian radial basis function, and sigmoid kernel.

3.4.2 Random Forest

Random Forest (RF) is an ensemble learning method applied in both classification and regression tasks. It is based on the concept of *decision tree* — a supervised rule-based

[13]Image source: https://www.quora.com/What-does-support-vector-machine-SVM-mean-in-laymans-terms

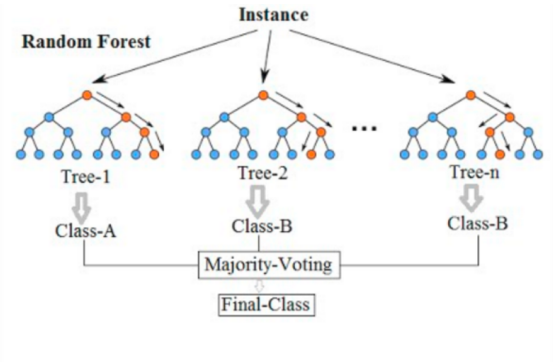

Figure 3. Random forest illustration for the classification task.

model for decisions and their possible outcomes represented as a tree, a graph or a flowchart. For instance, given the training data for a classification task, the decision tree algorithm will create a set of rules that discriminate the data. In other words, it constructs a tree with the most discriminative attributes selected for each level of the tree. These rules are then used to make predictions on unseen data.

A random forest is basically build of multiple decision trees. The algorithm consists of two general stages: random forest creation and actually making predictions. The creation of a random forest is an iterative process that starts with randomly selecting k features out of total n features. Next, the current tree construction begins. The best feature is selected to be the root node and remaining $k - 1$ features as child nodes for this tree ending up with leaf nodes that represent the target classes. This procedure is repeated to build m random trees which will finally form a random forest.

The predictions in random forest are usually obtained using the procedure known as the *majority voting* method which is one of the most common prediction aggregation methods in ensemble learning. The prediction derivation process and an example random forest is provided on Figure 3[14] and is as follows:

1. Get the target leaf node for a test data point using the rules of each random decision tree and store them;

2. Calculate the total number of votes for each predicted target class;

3. Pick the most frequent target class as final prediction.

The orange nodes represent the flow from most to least discriminative features on the prediction derivation step.

[14]Source: https://cdn-images-1.medium.com/max/800/1*i0o8mjFfCn-uD79-F1Cqkw.png

3.4.3 Convolutional Neural Networks

Convolutional Neural Networks (CNN) showed state-of-the-art performance in image recognition and classification tasks. Additionally, they proved their efficiency in text and sentence classification tasks. CNNs are very similar to simple Feed-forward Neural Networks: they consist of neurons with learnable weights and have the same training procedure. A single neuron in CNN represents a region within an input sample i.e. a piece of image or text. CNN is also different in terms of its layers and consists of the input layer, hidden layers (*convolutional*, *pooling*, *normalization*, *fully-connected*) and the output layer. A neural network is considered a convolutional if it has at least one convolutional layer (Figure 4)[15].

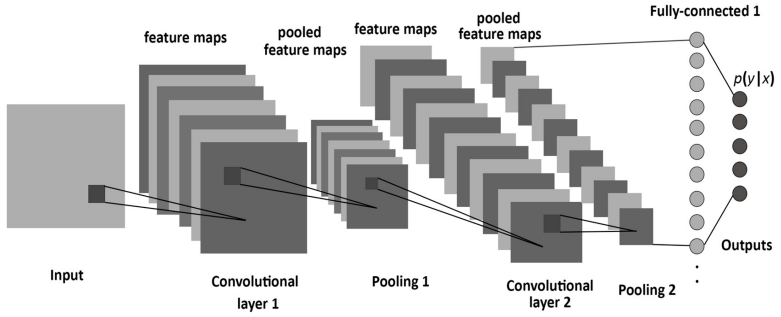

Figure 4. Convolution neural network architecture example with one input layer, two pairs of convolutional and pooling layers and one fully-connected layer.

Convolutional layer Convolutional layer consists of filters (matrices) with weights that are learned during the training phase. During the convolution operation, the filter slides along the encoded input data with some predefined step and performs a dot product operation on each *"window"* [31]. In other words, it convolves the input data, reduces its dimensionality and provides convolved features as the output. These features from different filters are then stacked into one activation matrix. The detailed illustration of the convolution operation is provided on Figure 5[16]. The aforementioned filters' weights are trained to activate if they capture some patterns in data. Intuitively, the more convolutional layers we stack the more specific features and patterns can be detected.

Pooling layer The pooling layers in CNNs are applied just after the convolutional layers to prevent overfitting, reduce the spatial dimensions of obtained data representations, and as a result reduce the computational resources needed for training. The input

[15]Source: http://www.mdpi.com/entropy/entropy-19-00242/article_deploy/html/images/entropy-19-00242-g001-550.jpg

[16]Source: https://www.safaribooksonline.com/library/view/deep-learning/9781491924570/ch04.html

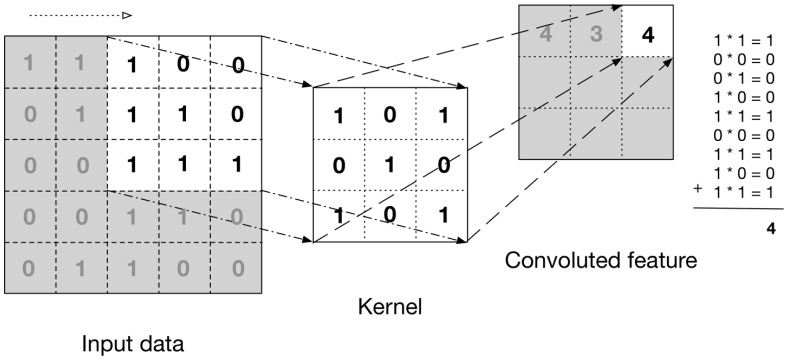

Figure 5. Convolution operation example.

matrix (feature map) obtained from the convolutional layer is split into non-overlapping sub-matrices. The pooling operation keeps one value which could be maximum, average or any known norm of the values in this matrix. The output of pooling layer is a matrix built of these values for each input sub-matrix. In practice, max-pooling operation is often used because it keeps the strongest feature data representations throwing the weaker ones away (Figure 6[17]).

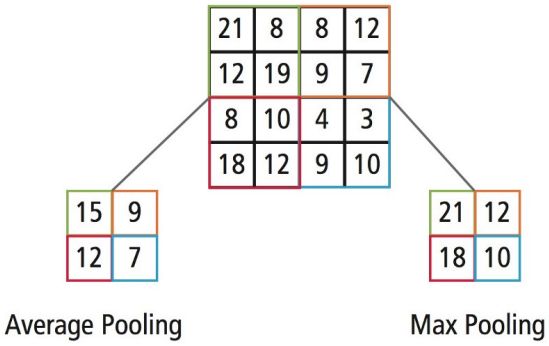

Figure 6. Average and max-pooling operations example. Max-pooling keeps stronger features that are represented with bigger values.

Fully-connected layer Having the extracted features from the previous convolutional and pooling layers we intend to get the final predictions. This is usually performed by a fully-connected (FC) layer that takes as input the output from the preceding layers and outputs an N-dimensional vector where N is the number of target classes. FC layer tries to find the strongest correlations between these high level features and target classes. Neurons in FC layer are connected to all activations from the previous layer.

[17]Source: https://medium.com/@Aj.Cheng/convolutional-neural-network-d9f69e473feb

Activation functions Activation functions play a significant role in artificial neural networks learning. Activation function converts the sum of products of inputs multiplied by weights into output values. These output values are then fed into subsequent layers as inputs. The reason to use activation functions is to make the neural network able to learn complex non-linear relationships from data [18]. The most popular types of activation functions are *Sigmoid*, *Rectified Linear Unit (ReLu)*, and *Hyperbolic tangent*. In our research, we used ReLu on convolutional layers and Sigmoid activations on fully connected layer. Their illustrations are provided on Figure 7[19].

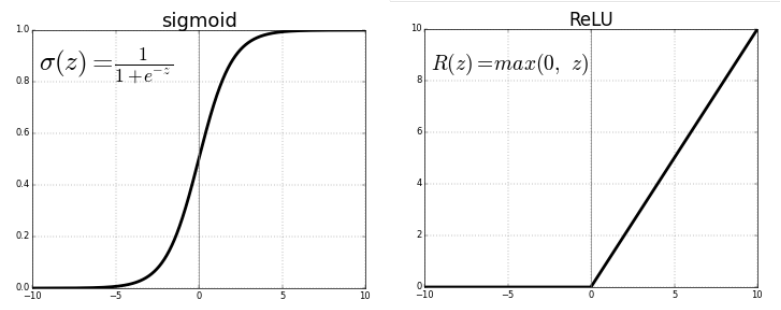

Figure 7. Sigmoid and ReLu activation functions and their formulas.

CNN architecture In our work we experimented with CNN architecture based on network proposed by Yoon et al. (2014)[26]. 8. It consists of embedding layer as input followed by one-dimensional convolutional layer, max-pooling layer, and a prediction output layer. The model is a multi-channel convolutional network which means that it consists of multiple similar networks and where each channel has different kernel size. This ensures that the model will process text considering not only single words but also their combinations of different predefined size i.e. *n-grams* and will learn their best combinations and interpretations that lead to better prediction. In our implementation, we also added dropout layer between the convolutional and the max-pooling layer in order to prevent fast *overfitting* — the issue when the model just memorizes training data and loses the generalization power on unseen data. The concern about overfitting is warranted as the collected dataset described in 4 is relatively small which means that there is a high risk of overfitting.

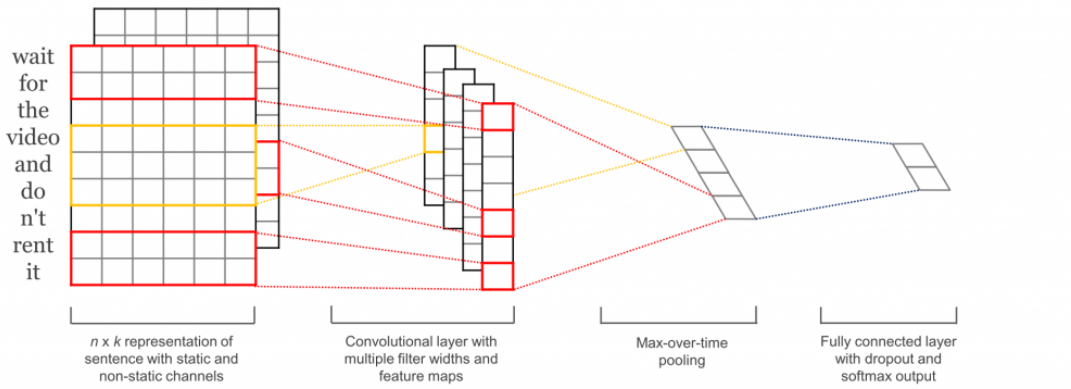

Figure 8. Kim's CNN architecture for sentence classification. Source: [26]

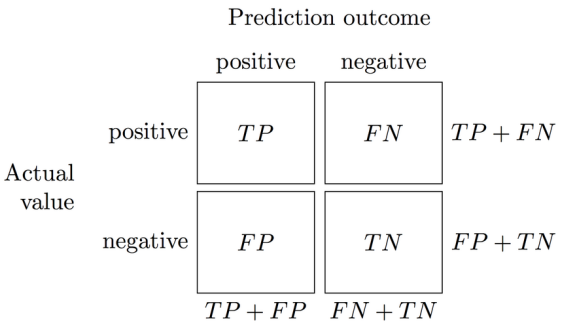

Figure 9. Confusion matrix for binary classification task.

3.5 Evaluation metrics

This section describes the evaluation metrics applied in this work. These metrics rely on a *confusion matrix* which incorporates the information about each test sample prediction outcome (Figure 9). TP stands for the number of true positive predictions, TN – true negative predictions, FP – for false positive predictions, and FN – for false negative predictions. The most straightforward classifier evaluation score is *accuracy* which is defined as:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN},$$

Accuracy provides reasonable results when the data has equal number of class samples but it loses its representation power when classes are imbalanced. In this case, this metric will be biased towards the majority class. Thus, if the aim is to assess class related classification performance, typically *precision*, *recall* and *F1-score* are used. *Precision* estimates how many positively identified samples were correct, while *recall* estimates what proportion of positive samples was correctly identified.

$$Precision = \frac{TP}{TP + FP}, \; Recall = \frac{TP}{TP + FN}$$

F1-score incorporates both precision and recall and is actually a harmonic average of these measures.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

[18]https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

[19]Source: https://cdn-images-1.medium.com/max/800/1*XxxiA0jJvPrHEJHD4z893g.png

4 Data collection

This section describes the data collection and preprocessing approach applied in our work. To recap, our aim is to retrieve the blog posts of self-reported ill people and a control group – healthy individuals – from a blog provider platform. We assume our collected data will be similar in a context of discussed topics so that non-neural classifiers will not biased in their predictions i.e. will not predict the disorder based on the topic.

4.1 Collection method

Web data retrieval technique applied in this work falls into the category of web wrappers although the XPath queries were used to get specific DOM elements. The goal of this step was to extract two corpora of blog posts – control and clinical – such that they would be similar in terms of the discussed topics.

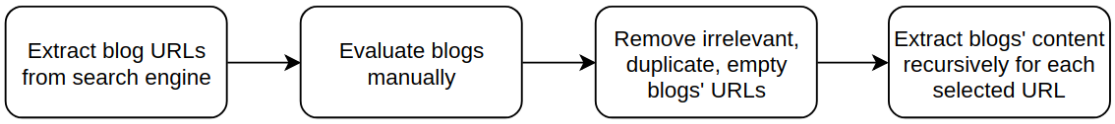

Figure 10. Scraping procedure flowchart applied in this work.

The data for this work was extracted from the Blogger platform in an semi-automated way using a Python script and Selenium package for browser automation. Initially, the idea was to collect both control and clinical blogs by keyword interest "*depression*" using the platform built-in filter, then manually verify scraped blog URLs, and finally get desired blog posts by these URLs using the scrapper. The drawback of using the built-in depression filter was that it mostly retrieved blogs that would fall into control group, such as blogs maintained by psychiatrists, volunteers, psychologists and blogs about religion. Therefore, another query was constructed with the help of Google Advanced Search tool and can be interpreted as follows:

- Find pages that contain all these words: *my, life, clinical, depression, anxiety*
- Any of the words: *anxiety, depression*
- Within the website: *www.blogspot.com*.

This detailed search query brought much more blogs where people mention their clinical or self-stated diagnosis.

The scraping method introduced more than 100 blog links that were then manually evaluated. We ended up with 47 clinical and 36 control subject blogs which were then

Table 1. List of applied regular expressions for string cleaning.

Target string	Substitute
^https?:\/\/.*[\r\n]*	empty
[^A-Za-z0-9(),!?\']	whitespace
([0-9])	empty
\\\\	empty
\'s	whitespace + 's
\'ve	whitespace + 've
n\'t	whitespace + n\'t
\'re	whitespace + \'re
\'d	whitespace + \'d
\'ll	whitespace + \'ll
,	whitespace + , + whitespace
!	whitespace + ! + whitespace
\?	whitespace + ? + whitespace
\(	empty
\)	empty
\s{2,}	whitespace

scraped entirely. In total, there are 10799 and 6176 control and clinical blog posts respectively, and later cropped to the same size to avoid data imbalance which means that the resulting classes would have similar number of posts for each of the two classes.

4.2 Data preprocessing

This section describes data preprocessing — a vital step in any natural language processing task. Firstly, all input data was cleaned using regular expressions which are actually the sequences of characters that define the search string in a text. In particular, the initial string cleaning pipeline is based on Yoon Kim's preprocessing approach[20], modified and extended with additional regular expressions matching and replacing the URL addresses, contractions and redundant whitespaces with a single whitespace. Additionally, brackets, colons, dashes, punctuation marks, contractions and all newline symbols were removed. Finally, the posts were lowercased and saved as separate text files in order to perform the preprocessing step only once thus considerably reducing the overall execution time. The list of applied regular expressions as well as their target substitutes is provided in Table 1.

The *lemmatization* were chosen because unlike *stemming* it aims to remove word endings based on vocabulary and morphological word analysis. This ensures that the

[20]https://github.com/yoonkim/

Table 2. Resulting data partitions summary.

	Train	Dev	Test
Number of posts	9196	2094	1205
Clinical posts	4546	1010	634
Control posts	4650	1084	571

word endings will not be roughly dropped often resulting in senseless word pieces (*stemming*) but rather transformed into word lemmas conformant with dictionary. For example, the lemmatization output for words *"good", "better", "best"* is just the common lemma *"good"* while stemming will result in *"good", "bet", "best"*. Furthermore, all words were enriched with *part-of-speech tags* (POS-tags) to represent their word classes and improve lemmatization results.

Then, the data was split into three parts with 70%, 15% and 15% partitions sizes for training, development and test set respectively. To ensure that each author belongs to only one partition splitting was done based on the blog sizes i.e. the number of posts per author. Moreover, it was attempted to keep the distribution of blog sizes similar across these train, dev and test parts so as to prevent data imbalance. The resulting data split is summarized in Table 2.

5 Topic modeling

The purpose of topic modeling within the scope of this work is to extract and evaluate the observed topics. In addition to this, topic model will help to check the assumption we made that our control and clinical group discuss similar topics. Topic modeling experiments have been performed using the *scikit-learn* [32] Python LDA implementation.

5.1 Topic number parameter search

The number of latent topics is chosen right before the topic model training and considerably affects the interpretability of the final mode. In other words, the obtained topics have to be evaluated at least manually according to their most representative words. This analysis helps to determine the redundant (duplicate) and "rubbish" topics and adjust the topic number parameters to reach better performance.

In this work, the LDA model was evaluated based on particular text classification task. We firstly tuned the SVM hyperparameters on 15 LDA topics. Subsequently, we trained multiple topic models on train set, trained the SVM classifier on document-to-topic features extracted from these models and then evaluated the performance of the classifier on dev set. The search grid started from 5 and ended up with 150 topics with step 5. The obtained best number of topics was fixed for all subsequent classification methods, and the SVM was fine-tuned again. It was also observed that SVM's accuracy fluctuated during the topic number parameter search as illustrated on Figure 11.

5.2 Topic model analysis

The resulting LDA model requires further analysis to reach better understanding of the scraped data. The topic model parameter search procedure described in 5.1 shows the best classification score of 76% with 30 topics and hereinafter we will analyze this best topic model.

These topics were manually investigated and labeled according to their 15 most representative words (Table 7). They topics often share the same words and lead to the same (*"family", "psychotherapy"*) or very similar (*"mental health"*, *"mental illness"*, *"physical treatment"*, *"treatment"*) topic labels.

The resulting 30-dimensional document representations i.e. document-to-topic vectors were visualized in two-dimensional space. We used t-SNE to perform dimensionality reduction and get a two-dimensional coordinates for our data [33]. The t-SNE algorithm has two main hyperparameters that has an impact on final visualization – perplexity and number of iterations. We tuned them and produced the data representation provided on Figure 12 using the *perplexity=80* and 1100 iterations. The figure shows quite logical topic arrangement putting related topic clusters close to each other and nicely arranges documents with different major together.

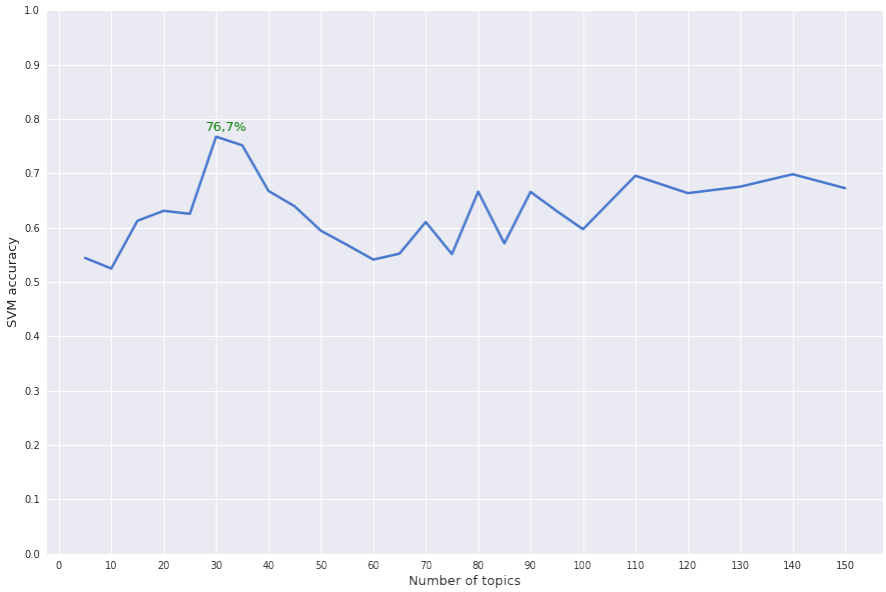

Figure 11. LDA topic number search.

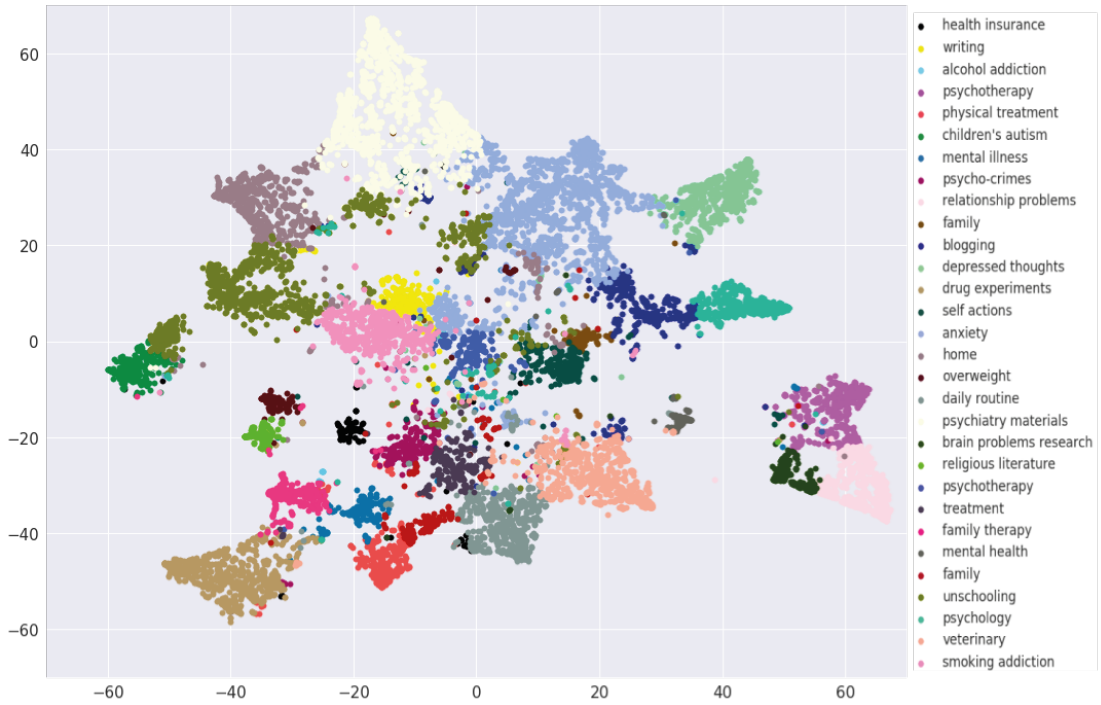

Figure 12. Two-dimensional t-SNE topic data representation.

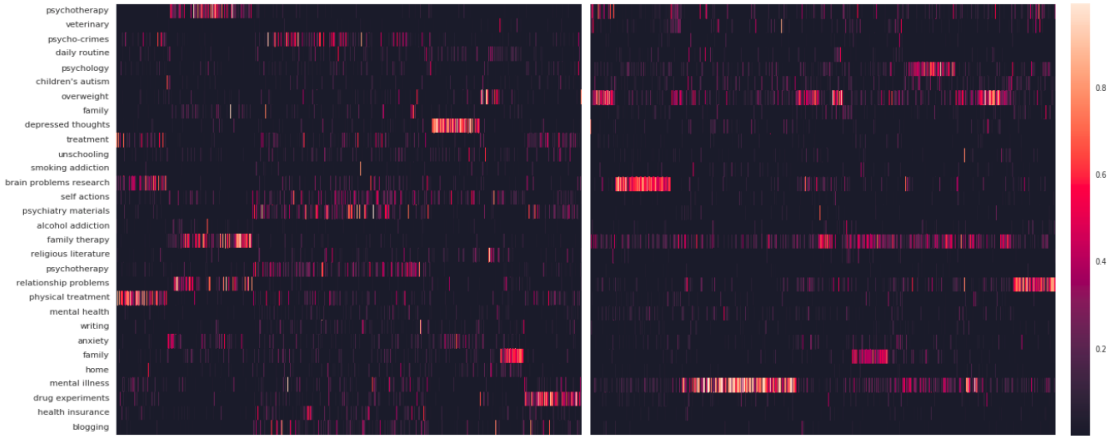

Figure 13. Heatmap visualizations of document-to-topic probabilities for control (*right*) and clinical (*left*) text corpus.

Next, we want to compare the topic distribution in clinical and control datasets to understand whether there are major topical differences between these datasets. This is because our data collection approach was aimed to avoid these differences. In order to estimate it, we present the document-to-topic probability distributions for both datasets as heat maps so the topics can be visually inspected (Figure 13). The visualizations outline the shift towards the following topics: *"daily routine"*, *"anxiety"*, *"self actions"* and *"religious literature"* for the clinical group. Moreover, the control group discuss more *"depressed thoughts"*, *"family"* and *"family therapy"*, and *"physical treatment"*. Potentially, this imbalance could make classifier predict based on topic assignments.

We also estimated the statistical significance of the clinical and control groups in terms of topic distributions. To test the independence of two categorical variables, the *Chi-square test of independence* and *G-test of independence* were applied. They both require a contingency table as input which contains: 30 columns (LDA topics) and two rows — *control* and *clinical* with document counts. The topic with the highest probability was assigned to each particular document.

Our null (H_0) and alternative (H_1) hypothesis are as follows:

- H_0 — topic assignment is independent upon the mental health condition.
- H_1 — topic assignment depends on the mental health condition.

After the construction of contingency table and having the hypothesis stated we can perform the hypothesis testing. The SciPy Python's package has both these methods implemented so we applied them and received the following results:

- Chi-square: *p-value<0.0001*, *chi2=7070.15*, *degrees-of-freedom=29*

- G-test: *p-value<0.0001*, *g=8575.64*, *degrees-of-freedom=29*

According to the tables for these statistics both methods resulted in values that are considerably higher than the critical values listed for 29 degrees of freedom. Moreover, the p values are close to zeros in both cases. Consequently, the H_0 hypothesis is rejected which means that the H_1 is true and there are significant differences in topic distributions between clinical and control groups.

6 Document Classification

The goal of document classification is to predict the document's target class label by its features. In this section we experiment with the feature extraction approaches, described in 3.2, combined with the selected classifiers described in 3.4.

The measure of success is the proportion of correct predictions with respect to all predictions. We also measure precision and recall but consider recall more important because if the depressed or anxious person is not identified correctly then there are fewer chances that this person gets necessary treatment.

6.1 Experimental Setup

This section provides the optimal configurations applied to the classification methods and describes their technical details. In this work we use *scikit-learn* [32] implementations of all non-neural classifiers, text transformation methods and automated tuning procedures. The neural network model is constructed with Keras [34] package for Python.

6.1.1 Non-neural setup details

The BOW text representation is constructed with 100000 features i.e. word types. The BOW representation of a corpus is a sparse matrix $n \times m$ where n stands for the number of documents in corpora and m is the number of features. Subsequently, we also apply TFIDF transformation on the obtained BOW model and experimented with these two text representation strategies. Additionally, we transform the input text data using an LDA topic model, extract the document-to-topic representation matrices and use them as features for chosen classifiers.

The specifications of the non-neural classification setups are as follows:

BOW+SVM SVM classifier with BOW features. We use the default SVM parameters for this setup.

BOW+RF The next classification setup includes the same BOW features but RF classifier. The optimal hyperparameters are: $numer\,of\,trees = 500$, $max\,number\,of\,features = 30$, $max\,tree\,depth = 2$.

TFIDF+SVM SVM classifier with TFIDF features. The SVM parameters have been tuned and fixed: regularization term $C = 3$, $gamma = 0.5$ and radial basis kernel.

TFIDF+RF RF classifier with TFIDF features. The optimal parameters are: $number\,of\,trees = 500$, $max\,number\,of\,features = 30$, $max\,tree\,depth = 2$.

LDA+SVM SVM classifier on top of LDA features. The SVM parameters have been tuned and fixed: regularization term $C = 0.01$, $gamma = 10$ and radial basis kernel.

LDA+RF In this experiment we apply the RF classifier on top of LDA features. The optimal parameters are: $number\,of\,trees = 500$, $max\,number\,of\,features = 30$, $max\,tree\,depth = 2$.

6.1.2 Neural networks setup details

In the initial scenario with neural model, CNN takes as inputs the sequences of vectors containing words encoded by their unique respective numbers. Thus, the input is a matrix of size $m \times n$ where m is the length of the document and n is the size of the vocabulary. This representation requires these vectors to be padded with zeros to ensure the input size is the same across all posts. This is performed with Keras's built-in text Tokenizer and method for sequence padding. We experiment with two scenarios:

1. the input word embedding layer is initialized randomly and learned during the training
2. the input word embeddings are initialized with pretrained GloVe embeddings [25] and fine-tuned during training.

The specifications of neural the classification setups are as follows:

CNN-rand The CNN model described in 3.4.3 and randomly initialized embedding layer. The parameters are:

- embedding layer shape: (*vocabulary size*, 100)
- 1D-convolutional layer: *kernel sizes* – 2, 3, 4; *number of filters* – 128; *activation function* – relu
- dropout: 0.6
- max-pooling layer: *pool size* – 30
- fully connected layer: 10 units; *activation function* – sigmoid

CNN-GloVe CNN model with the embedding layer initialized with the pretrained GloVe word vectors. The parameters are:

- embedding layer shape: (*vocabulary size*, 100)
- 1D-convolutional layer: *kernel sizes* – 2 (not trainable channel), 3 (trainable channel), 4 (trainable channel); *number of filters* – 128; *activation function* – relu
- dropout: 0.6
- max-pooling layer: *pool size* – 30
- fully connected layer: 10 units; *activation function* – sigmoid

CNN with varying post size In this scenario we train multiple models with various post sizes. The post cropping have been performed in the next way – we just kept first n words from the beginning of each document. All in all, we evaluate 17 models from 100 words per post and up to 3300 words per post with step 200. The models' parameters are the same as used in CNN-rand scenario.

6.2 CNN training

Since the task is binary classification it was reasonable to use *binary crossentropy* as loss function and *accuracy* as the metric. The chosen optimizer is *adam* with learning rate tuned and set to 0.0001. It is lower than the default value because the dataset is small and a high learning rate leads to quick overfitting and badly affects model learning.

The *dropout* is usually applied to prevent the overfitting of a neural network by randomly removing the nodes as well as the connections between them. For instance, the 0.5 dropout means that each individual neuron will be dropped with probability $1 - 0.5 = 0.5$ in the training stage and then will be added to the network with their initial weights on the prediction step. In this work, the dropout parameter was tuned for both of the aforementioned CNNs. Initially, we set the number of training *epochs* to 20 and applied the Keras's *EarlyStopping* callback which breaks the training process if the specified measure – in our case validation accuracy – stops decreasing. We determined that in CNN-rand and CNN-GloVe scenarios models start overfitting while being trained on full data around third and fourth epoch as the callback was executed.

There are two different training environments used in this work. The non-neural classifiers were trained on Intel® Core™ i5-6200U CPU 2.30GHz $\times$ 4 with 16 GB of RAM. The neural models were executed and tuned on 2 x Intel® Xeon® CPU 2.20GHz with 256 GB RAM.

Table 3. Document-level classification performance on development set.

Methods	Accuracy	Precision	Recall	F1-Score
BOW+SVM	0.8108	0.7479	0.9168	0.8238
BOW+RF	0.7975	0.7495	0.8712	0.8058
TFIDF+SVM	0.7994	0.7875	0.8000	0.7937
TFIDF+RF	0.7893	0.8399	0.6960	0.7612
LDA+SVM	0.7673	0.7376	0.7712	0.7541
LDA+RF	0.6840	0.6215	0.8099	0.7033
CNN-rand	0.7531	0.8833	0.5623	0.6872
CNN-GloVe	0.7769	0.8081	0.7049	0.7530

Table 4. Author-level classification performance on development set.

Methods	Accuracy	Precision	Recall	F1-Score
BOW+SVM	0.5000	1.0000	0.3571	0.5263
BOW+RF	0.6000	1.0000	0.4666	0.6363
TFIDF+SVM	0.6000	1.0000	0.5294	0.6923
TFIDF+RF	0.5789	1.0000	0.4285	0.6000
LDA+SVM	0.6000	1.0000	0.4666	0.6363
LDA+RF	0.6500	1.0000	0.5000	0.6666
CNN-rand	0.7500	1.0000	0.6428	0.7826
CNN-GloVe	0.8000	1.0000	0.7142	0.8333

6.3 Classification results

This section provides the performance comparison tables for applied methods. The accuracy, precision, recall and F1-scores of the applied classifiers have been summarized in tables 3 and 4 for document- and subject-level evaluation on development set and tables 5 and 6. The subject-based evaluation is grounded on the aforementioned majority voting principle. If the number of correct per-subject predictions is bigger than half of the subjects' blog posts the subject prediction is considered to be *true* and *false* otherwise.

6.3.1 Non-neural classifiers

According to measures provided in the aforementioned tables, we see that simpler models perform better on the development set. In particular, the BOW+SVM scenario outperforms neural network and other scenarios on post-level classification reaching the accuracy of 81% which is slightly better than BOW+RF (79.7%) and TFIDF+SVM (79,9%). The highest recall and F1-score is obtained with BOW features. In general, both BOW and TFIDF features have worked better than LDA topic model's features for

non-neural classifiers. Most of the non-neural classifiers reached around 60% author-level accuracy, however, the best recall values were obtained with TFIDF+SVM and LDA+RF scenarios.

However, the test set evaluation results in tables 5 and 6 show that BOW+SVM, TFIDF+RF and LDA+RF work the best in terms of accuracy and recall on post-level classification. Similarly, the top three author-level classifiers are BOW+SVM, TFIDF+SVM and TFIDF+RF. In general, the test results are considerably lower than dev set results for the majority of classifiers except for LDA+RF and TFIDF+RF scenarios. It means that these two models generalize better on unseen data. It is also worth mentioning that on author-level evaluation, when using the majority voting aggregation method, all models obtain precision of 1.0, which is also the reason we focus mostly on recall.

Table 5. Document-level classification performance on test set.

Methods	Accuracy	Precision	Recall	F1-score
BOW+SVM	0.6989	0.5918	0.6409	0.6695
BOW+RF	0.6174	0.6261	0.6217	0.5991
TFIDF+SVM	0.6632	0.8343	0.4045	0.5448
TFIDF+RF	0.7038	0.8023	0.5383	0.6443
LDA+SVM	0.6671	0.8033	0.4397	0.5683
LDA+RF	0.7165	0.7571	0.6350	0.6907
CNN-rand	0.6474	0.8649	0.3467	0.4950
CNN-GloVe	0.6989	0.8114	0.5158	0.6307

Table 6. Author-level classification performance on test set.

Methods	Accuracy	Precision	Recall	F1-score
BOW+SVM	0.6250	1.0000	0.5200	0.6842
BOW+RF	0.5588	1.0000	0.4444	0.6153
TFIDF+SVM	0.7272	1.0000	0.6400	0.7804
TFIDF+RF	0.6363	1.0000	0.5200	0.6842
LDA+SVM	0.4000	1.0000	0.4000	0.5714
LDA+RF	0.6060	1.0000	0.5000	0.6666
CNN-rand	0.7452	1.0000	0.6810	0.8102
CNN-GloVe	0.7857	1.0000	0.7272	0.8421

6.3.2 Neural classifiers

The neural classifiers performed worse on development set than non-neural models which can be seen from the resulting tables. CNN-GloVe model outperformed CNN-rand

scenario reaching 77,7% and increasing the accuracy by 2.4% as well as considerably improving the recall score. It was empirically identified that model fails to learn when the bi-gram channel's weights are non-static which means they are being adjusted during the training. Thus, we fix them and allowed the model to train only three- and four-gram channels. The issue with bi-gram channel should be studied more in further work because it remains unclear – either it is not being updated on the training phase or it introduces too much noise which causes the training failure.

CNN-glove model obtains one of the best document-level accuracies comparing to the other models and reaches 69.8% on test set even though it makes many false negative predictions (low recall). Similarly, this scenario shows almost 6% better accuracy result on test set than CNN-rand approach. Finally, both convolutional classifiers reached the best accuracies and recall on the test set for author-level prediction among all experimental setups.

6.3.3 Post length experiments

To recap, we have trained multiple CNN models keeping first n words for each post in training data. The post- and author-level dev accuracies and recalls for the trained neural models are provided in Figure 14. Three best post-level accuracies are 84%, 83% and 82% obtained with 1100, 1700 and 700 words per post respectively while CNNs with 1100 and 1700 post size restrictions result in two best recall values. The average of the post-level accuracies for these 17 models is higher than the one we have discovered on dev set with CNN-rand or CNN-GloVe scenario.

The author-level evaluation shows that classifiers predict depressed and anxious authors better with 700, 1700 and 2300 words scoring 85%, 85% and 80% accuracy on the dev set. Although the performance scores are higher than those obtained with CNN model trained on full data, the judgment remains unclear because the curves for subject-level results fluctuate considerably and appear to be unstable.

Figure 14. The history of accuracy and recall for post-level ($left$) and author-level ($right$) performance on development set. The x and y axis refers to post length and obtained measure's value respectively.

7 Conclusions and discussion

In this work, we proposed and implemented the data collection approach to collect the blog posts from the blog provide platform. The main point of it was to retrieve the blogs such that the topics described there are not dependent for the control and clinical corpora. Secondly, we experimented with BOW, TFIDF and topic modeling features as input representations to SVM and Random Forest Classifier. Thirdly, we trained Convolutional Neural Networks with both randomly initialized and pretrained word vectors. Finally, it was explored if the length of the blog posts affect the performance of the CNN model and how much.

7.1 Inference

The proposed data retrieval method succeeded and the dataset was successfully collected from the blog provider resource. However, the further analysis of the collected data showed that our assumptions about the data were not confirmed. In particular, the topic model showed that the clinical and control subjects discussed different topics. Also, despite the fact that we avoided the extracted data to be skewed towards the depression topic using the proposed data collection approach, classes appeared to be dependent in terms of topics covered by their representatives. Furthermore, both Chi-square and G-tests confirmed that the topic distributions assigned to documents are statistically significantly different between clinical and control groups. This dependency highly likely affects the classification results and explains the relatively good performance of LDA-based classifiers on the test set.

We experimented with non-neural classifiers (Support vector machines and Random forest) and three types of features (Bag-of-words, Term frequency-inverse document frequency and LDA-based document features). The majority of simple non-neural scenarios with frequency-based features performed the best on post level while neural networks completely outperformed the other scenarios on subject level prediction reaching greater scores for both accuracy and recall. Additionally, we compared the Convolutional neural network classifier with randomly initialized embedding layer weights and using pre-trained word embedding weights. As we expected, the experiments showed an improvement in the model with pre-trained word vectors comparing to one with randomly initialized weights.

The experiment with multiple Convolutional neural network classifiers trained on cropped posts showed better accuracies on the test set for shorter text inputs than the identically-defined network trained on full data. The second reason for this is might be that the reference model architecture was originally developed for short sentences thus was able to learn from cropped text pieces better. However, the results of the experiment with cropped posts require further analysis because the obtained accuracy and recall do not draw up trends and thus, the effect of the blog post length to the classification performance should be studied more.

7.2 Future work

In this section, we propose some possible improvement strategies for future work. The further improvements can be roughly divided into two categories depending on their direction: data-oriented improvements, experimental setup improvements.

7.2.1 Data oriented improvements

The data collection part takes the considerable time required for manual evaluation of candidate blogs and the label assignments are error prone without the support of mental health domain experts. The inaccurate labels obviously affect the classification performance as they introduce noise to training data and cause machine learning methods to learn "wrong" features. This issue can be resolved by introducing the external assistance of mental health domain experts aimed on improving the accuracy of assigned labels. Additionally, the data source could be replaced by more popular blogging platform to collect more data.

Another possible improvement is to experiment with smarter text preprocessing and introduce different substitute words depending on the nature of the original string. For instance, categorize stopwords, add LIWC-like word categories and define the respective target word substitutions. This enhancement would reduce the number of features thus, help to provide clues for model interpretation.

7.2.2 Experimental setup improvements

The experimental scenarios used in this work do not cover the whole range of possible machine learning methods that can be applied in such document classification task. In particular, better results could be obtained with Hierarchical Attention Networks (HAN) introduced by Yang et al. (2016) [35]. Their main motivation is to capture the document's hierarchical structure and use this information in model construction. In addition to this, the HAN's architecture proposed in the study ensures high interpretability through attention visualization of learned text features.

Similarly, there also exists *eli5*[21] – a Python library that allows to debug and visualize machine learning methods applied for text classification tasks. The library could be used to help to interpret features and estimate their contribution to the final prediction.

[21] https://eli5.readthedocs.io/en/latest/index.html

References

[1] Depression: Overview. http://www.who.int/news-room/fact-sheets/detail/depression.

[2] Max Hamilton. A rating scale for depression. *Journal of neurology, neurosurgery, and psychiatry*, 1960.

[3] M Mendelson Aaron T Beck, C Ward. Beck depression inventory (bdi). 1961.

[4] Robert L Spitzer Kurt Kroenke and Janet BW Williams. The phq-9. journal of general internal medicine. 2001.

[5] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[6] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, pages 91–100, New York, NY, USA, 2008. ACM.

[7] Yuan Gong and Christian Poellabauer. Topic modeling based multi-modal depression detection. 2018.

[8] Hiraga Misato. Predicting depression for japanese blog text. In *Proceedings of ACL 2017, Student Research Workshop*. Association for Computational Linguistics, 2017.

[9] W. Bucci and N. Freedman. *The language of depression*. Bulletin of the Menninger Clinic, 1981.

[10] E.-M. Gortner S. Rude and J. Pennebaker. *Language use of depressed and depression-vulnerable college students*, volume 18. Cognition Emotion, 2004.

[11] N. Kuvshinova A. Krasnov D. Romanov D. Smirnova, E. Sloeva and G. Nosachev. Language changes as an important psychopathological phenomenon of mild depression. In *Proceedings of the 21st European Congress of Psychiatry*, volume 28. Elsevier, 2013.

[12] 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. *Clinical Psychological Science*, 2018.

[13] JW Pennebaker, KG Niederhoffer, and MR Mehl. Psychological aspects of natural language use: Our words, our selves. *Annual Review of Psychology*, 54:547–577, January 2003.

[14] Boyd RL Pennebaker JW, Booth RJ and Francis ME. Linguistic inquiry and word count: Liwc2015. 2015.

[15] H. Andrew Schwartz, Salvatore Giorgi, Maarten Sap, Patrick Crutchley, Johannes Eichstaedt, and Lyle Ungar. Dlatk: Differential language analysis toolkit. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 55–60. Association for Computational Linguistics, 2017.

[16] Reilly Grant, David Kucher, Ana M. Leon, Jonathan Gemmell, Daniela Stan Raicu, and Samah J. Fodeh. Automatic extraction of informal topics from online suicidal ideation. *BMC Bioinformatics*, 19:57–66, 2018.

[17] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 79–86. Association for Computational Linguistics, 2002.

[18] Munmun De Choudhury, Scott Counts, and Eric Horvitz. Social media as a measurement tool of depression in populations. In *Proceedings of the 5th Annual ACM Web Science Conference*, pages 47–56, New York, NY, USA, 2013. ACM.

[19] Glen Coppersmith, Mark Dredze, and Craig Harman. Quantifying mental health signals in twitter. 2014.

[20] P Resnik, A Garron, and R Resnik. Using topic modeling to improve prediction of neuroticism and depression in college students. pages 1348–1353, 01 2013.

[21] Sunghwan Mac Kim, Yufei Wang, Stephen Wan, and Cecile Paris. Data61-csiro systems at the clpsych 2016 shared task. In *Proceedings of the Third Workshop on Computational Lingusitics and Clinical Psychology*, pages 128–132. Association for Computational Linguistics, 2016.

[22] Emilio Ferrara, Pasquale De Meo, Giacomo Fiumara, and Robert Baumgartner. Web data extraction, applications and techniques: A survey. *Knowledge-Based Systems*, 70:301 – 323, 2014.

[23] Alberto H. F. Laender, Berthier A. Ribeiro-Neto, Altigran S. da Silva, and Juliana S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Rec.*, pages 84–93, 2002.

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013.

[25] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[26] Yoon Kim. Convolutional neural networks for sentence classification. 2014.

[27] Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, New York, NY, USA, 2009. ACM.

[28] David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[29] Binary classification. `https://en.wikipedia.org/wiki/Binary_classification`.

[30] Corinna Cortes and Vapnik Vladimir. Support-vector networks. *Machine Learning*, 1995.

[31] Cs231n convolutional neural networks for visual recognition. `http://cs231n.github.io/convolutional-networks/`.

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[33] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[34] François Chollet et al. Keras. `https://keras.io`, 2015.

[35] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. Hierarchical attention networks for document classification. In *HLT-NAACL*, 2016.

Appendix

I. 30 LDA topics with the most representative words

Table 7. 30 labeled LDA topics according to their 15 most frequent words.

psychotherapy	therapy therapist clients psychotherapist client psychotherapy help often trauma work self somatic article find emotional experience
veterinary	dog cat pet gus animal food timmy sweet vet kitten two rescue hammie therapy love
psycho-crimes	psychiatric hospital mental state treatment patient violence care law person gun police ill court illness system violent
daily routine	today work week days run back going got home feeling little long morning hours weeks
psychology	health mental clinical psychologist university social services community trust psychology care foundation group treatment service
children's autism	kids children autism school boys parents year family right needs first say little every parent
overweight	food eating body weight diet blood health doctor disease exercise fat healthy heart cancer brain
family	mother father daughter son parents grief death wife told week child loss home mothers never
depressed thoughts	depression brain fear thoughts pain mind depressed book negative always better something feeling work myself
treatment	medication drug effects side pain treatment taking dose patient prescription prescribing take opioid addiction
unschooling	unschooling really school learning want children something learn world different education find important need say
smoking addiction	sugar smoking duck candy smoke halloween cocaine nicotine emotional tobacco support addictive channel cigarette stop

Table 7 continued from previous page

brain problems research	brain pubmed pmid research psychiatry risk number point study clinical authors human information associated problem
self actions	said got back never went little told something want thought first things started myself wanted
psychiatry materials	new psychiatry dr article podcast post york medical blog times talk psychiatrist listen clinkshrink interview
alcohol addiction	alcohol addiction stress recovery drink abuse behavior problem sober help alcoholism drugs alcoholic anonymous meetings
family therapy	therapy emotional help family therapist mother article child parents feelings trauma work problems told feeling
religious literature	god though poetry must world love say never today better without myself self still last
psychotherapy	patient psychiatrist treatment therapy psychotherapy doctor therapist say sometimes someone want talk ask session often
relationship problems	relationship couples might person spouse marriage love often together find help want psychotherapist sex problems
physical treatment	care medical health psychiatrists managed physician psychiatry patient treatment problem medicine interest business years point
mental health	mental illness health suicide women depression stigma help media may support story men someone cancer tv
writing	book read write story written film writer stories first author novel movie amazon find chapter
anxiety	anxiety help myself want make need something feeling better work really try might find keep self hard
family	years family love year friend home lives old young may never children together times school
home	love around save room house fun water might baby great look play pictures art favorite old
mental illness	disorder symptoms bipolar depression diagnosis treatment illness mental psychiatric anxiety mood personality person diagnostic criterias

Table 7 continued from previous page

drug experiments	study data drug depression placebo patients treatment research article results trials clinical journal effect antidepressants
health insurance	money insurance pay company work call job phone financial need cost number plan care medicare
blogging	blog post facebook please email share write readers want comment blogging twitter blogs page site

License

Non-exclusive licence to reproduce thesis and make thesis public

I, **Yevhen Tyshchenko**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to:

   1.1 reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2 make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

   of my thesis

   Depression and anxiety prediction from blog posts data

   supervised PhD Kairit Sirts

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive licence does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, 09.08.2018