UNIVERSITÄT KOBLENZ · LANDAU
Fachbereich 4: Informatik

WeST People and Knowledge Networks
Institute for Web Science and Technologies

Inferring gender of Reddit users

Masterarbeit

zur Erlangung des Grades einer Master of Science (M.Sc.)
im Studiengang Web Science

vorgelegt von
Evgenij Vasilev

Erstgutachter:	JProf. Dr. Claudia Wagner
	Institute for Web Science and Technologies
	GESIS - Leibniz Institute for the Social Sciences
Zweitgutachter:	Dr. Florian Lemmerich
	RWTH Aachen University
	GESIS - Leibniz Institute for the Social Sciences

Koblenz, im März 2018

Erklärung

Hiermit bestätige ich, dass die vorliegende Arbeit von mir selbstständig verfasst wurde und ich keine anderen als die angegebenen Hilfsmittel – insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe und die Arbeit von mir vorher nicht in einem anderen Prüfungsverfahren eingereicht wurde. Die eingereichte schriftliche Fassung entspricht der auf dem elektronischen Speichermedium (CD-Rom).

	Ja	Nein
Mit der Einstellung dieser Arbeit in die Bibliothek bin ich einverstanden.	☐	☐
Der Veröffentlichung dieser Arbeit im Internet stimme ich zu.	☐	☐
Der Text dieser Arbeit ist unter einer Creative Commons Lizenz (CC BY-SA 4.0) verfügbar.	☐	☐
Der Quellcode ist unter einer GNU General Public License (GPLv3) verfügbar.	☐	☐
Die erhobenen Daten sind unter einer Creative Commons Lizenz (CC BY-SA 4.0) verfügbar.	☐	☐

.....

(Ort, Datum) (Unterschrift)

Zusammenfassung

Abstract

The content aggregator platform Reddit has established itself as one of the most popular websites in the world. However, scientific research on Reddit is hindered as Reddit allows (and even encourages) user anonymity, i.e., user profiles do not contain personal information such as the gender. Inferring the gender of users in large-scale could enable the analysis of gender-specific areas of interest, reactions to events, and behavioral patterns. In this direction, this thesis suggests a machine learning approach of estimating the gender of Reddit users. By exploiting specific conventions in parts of the website, we obtain a ground truth for more than 190 million comments of labeled users. This data is then used to train machine learning classifiers to use them to gain insights about the gender balance of particular subreddits and the platform in general. By comparing a variety of different approaches for classification algorithm, we find that character-level convolutional neural network achieves performance with an 82.3% F1 score on a task of predicting a gender of a user based on his/her comments. The score surpasses 85% mark for frequent users with more than 50 comments. Furthermore, we discover that female users are less active on Reddit platform, they write fewer comments and post in fewer subreddits on average, when compared to male users.

Contents

1 Introduction

1.1 Problem statement

The online discussion forum, Reddit, is a popular content aggregator that self-labels as "the front page of the internet". According to its calculations, it has 274 million unique monthly visitors [1] and is 8th most visited website in the world at the moment [2]. It consists of different communities that called subreddits, and these communities usually have a topic that unites its participants. The subreddits theme can range from a popular computer game to residents of a small town. By the end of November 2017, it had almost 1.2 million subreddits and has been adding approximately 15 thousand each month [3].

The primary goal of users on Reddit is sharing the content among subscribers of a particular subreddit. As mentioned before, each subreddit usually has some specific topic and users can only post content that is related to this topic. The content added by the user is called a post or a submission, and it can consist either of a link to some resource (e.g., news article, image, video) or a text [4]. Users can vote submission up or down, and it will determine in which order posts appear on a homepage of a subreddit. The users can also comment each submission, the comments are arranged in a nested fashion, and a comment that replies to another comment will appear under it. Comments can also be voted up and down by the users, which will influence the ranking of the comment on a submission page. This thesis will mainly focus on comments that are left by users in different subreddits.

Records of user activities on Reddit can provide an immense amount of valuable information for scientists. However unfortunately, amount of studies conducted on Reddit data is notably lower than ones from data of social network platforms like Facebook and Twitter. One of the indications of this is the number results found on Google Scholar (search engine for scholarly literature) when you query the name of the platform. For Facebook, it is 5.85 million, for Twitter 6.48 million and Reddit only 900 thousand. One of the reasons for lack of popularity among scientists is anonymous nature of Reddit. Unlike social networks, where the demographic information is explicitly stated or can be derived from the first name of the user, Reddit requires only a username when a person registers on the website. This username usually does not represents a name of the person, Reddit's itself suggests randomly generated usernames like "MovingPurism" and "CoatedRibs" during the registration process, which emphasizes the anonymity of the platform for the user.

Lack of demographic aspect of the vast amount of data generated by Reddit users makes it less appealing to scientists. This, in turn, constrains the research conducted by them to study Reddit communities.

This thesis proposes a novel approach of producing demographic labels for a subset of the Reddit users, by utilizing internal rules of several active subreddits. After establishing demographic labels for a fraction of the users, all their comments

on the platform are used to train machine learning models. Several experiments with various machine learning algorithms and deep learning architectures are conducted. These models are inferring the demographic labels (e.g., gender) of any users based on their comments. Releasing these pre-trained models should help other researchers navigate research of user behavior on Reddit.

1.2 Research questions

The following research questions will be covered in this thesis:

1. How can we come up with the demographic labels of users to assemble initial dataset? What labels can we extract (e.g., age, gender, ethnicity, spoken language)? Will labeled users be representative of the whole Reddit userbase?

2. How labeling can be extended with Machine Learning algorithms? What algorithm will be more suitable for this task?

3. What insights can we find in collected data? Can we find features that explicitly distinguish user groups with different demographics?

Discovering the ground truth about demographic labels of a subset of users is an important task because it will let us build machine learning models that can be trained on this data. After discovering the label for a given user, we will be able to use all the comments produced by this user. Majority of demographic labels are mostly constant (e.g., ethnicity, gender) and others are changing in predictable pace (e.g., age). After accumulating a large subset of users with assigned labels, we will be able to apply machine learning algorithms and see whether they are feasible in a task of predicting demographics of users based solely on the text that they have produced.

Machine learning algorithms vary in a level of complexity and explainability. By applying more straightforward algorithms like linear models, we can find how each word or word combination influences the decision of the model. Analyzing these models we can find words that are most distinguishing for different demographics.

1.3 Thesis structure

This thesis begins with a chapter dedicated to previous works in related fields, like Text classification problem, Author profiling and Analysis of social media. This section should help to establish the challenges that this problem holds and showcase how they were tackled before.

The Next chapter covers theoretical background of the machine learning and deep learning. It focuses on Natural Language Processing as a subfield of machine learning and classification task which is used in the thesis. This chapter also lays out the format of social media analysis, especially the structure of Reddit as a platform. It goes deeper into the types of subreddit and their governance.

Chapter number 3 describes the methodology that was used. It introduces the theoretical background and terminology that is required to understand the approaches that were used in this thesis. This chapter also explains how and where data was collected and goes into detail of labeling the gathered data. Subchapter after that describes the machine learning algorithms and deep learning architectures that were trained on accumulated data.

Chapter 4 discusses the experimental setup for chosen machine learning models as well as details of the dataset that was collected. Evaluation of selected models, aspects of implementation and hyperparameter choice is also covered. Additionally, analysis of discarded experiments that wound up unfruitful also reported. Demonstration of application of final models on a chosen subreddit and the discussion of limitations concludes this chapter.

The final Chapter 5 contains the summary of the conducted work and describes the work that has been done to achieve the results. Additionally, the ideas for future work and possible extensions of presented research are discussed.

2 Previous work

This chapter will cover previous works that have been conducted in areas that are related to thesis's topic. In the first part, we will cover what research was carried out in a field of demographic inference from the human-generated language. The next subsection will introduce the analysis of social media by the scientific community and particularly research papers on demographics of social media users. The last subchapter will cover existing scientific works on analysis Reddit as a social platform and behavior of its users.

2.1 Inference of demographics from language

Profiling the textual data for its author's demographic background is a common task [5, 6, 7], and it has been widely researched throughout the years [8, 9, 10]. The primary field of the application before the popularity of social media networks was in forensic linguistics [11]. However, with the rise of the internet and increased amounts of data that it produced this field attracted more scientist from different fields and research funding from big companies. Today, author profiling is an actively researched problem, and several conferences [12, 13, 14] and a competition [15] that include this topic are held every year. This section will offer a brief summary of some of the works in this field.

In research, conducted by Shlomo Argamon et al. [16] they focused on the analysis of British National Corpus and the dissimilarity in the usage of specific parts of speech by different genders. This study focused on a more formal type of speech in contrast to previous one, the average document in this corpus has more than 42 000 words and mainly relates to scientific topics. The research found that parts of speech that are useful for distinguishing male-authored texts are determiners (e.g., a, the, that, these) and quantifiers (one, two, more, some). For females, on the other hand, the pronouns (e.g., I, you, she, her, their, myself, yourself, herself) are all reliable indicators. The study also found a strong correlation between characteristics of male (female) writing and documents of nonfiction (fiction) genre. It should be noted that the study was conducted on the examples of British English and these findings may not apply to other dialects.

Another study by Matthew L. Newman et al. [17] was conducted on a manually collected corpus of written and spoken language. The study also showed that female language was more likely to include pronouns and the frequent use of negations was also a good indicator. The male language showed to have more extended words and more articles than female (e.g. "the music", "a journal"). They also looked into the influence of age on a prediction of gender. By controlling for the age of participants, they found that previously found gender-specific patterns remained the same. It means that discovered gender differences are consistent in all age groups.

In the paper by M Corney et al. [18] they apply support vector machine to a corpus of emails to predict the gender of the sender. To represent the text of the emails, they used a number of handcrafted features, like a word count or character count in an email and other, 221 in total. With this methodology, they achieved an F1 score of 70.2%. They found that the use of function words (e.g., "so", "very") is distinctive between genders, but they did not find any other distinctive features from produced feature set.

The study by Pennebaker et al. [19] investigates the relationship between language style and the age of a person. The data for this study was collected by multiple research labs in different countries. They asked participants from different age groups to write a text about a given topic or participate in an interview. The authors confirmed their hypothesis that aging positively correlates with use of positive words (e.g., happy, nice) they also found that with age people use more present-tense and future-tense verbs (e.g., am, goes, will, shall) as well as words longer than six characters. Additionally, with age people use fewer negative words (e.g., angry, ugly), past-tense verbs (e.g., was, went), personal references (e.g., friend, talk) and time-related words (e.g., hour, soon). Authors conclude that the findings are statistically significant and that language style of a person changes with age.

Demographics is not the only aspect of the person that can be derived from the text, the Linguistic Inquiry, and Word Count (LIWC) approach can predict the psychological state of the person. LIWC uses curated set of dictionaries of words that are associated with different emotions or psychological dimensions. In this approach, the text generated by the person is viewed as a set of words and these words are matched to the dictionaries. If a given word is part of categories like Sadness and Negative Emotion, the scores of this categories will be incremented. After matching all the words, the scores across all the categories will represent the psychological background of a person that produced the text. This approach was introduced by James W. Pennebaker et al. [20, 21] and later developed into a stand-alone software.

This section introduced research papers which study the connection between person's language and demographic background. This works showed that the language style carries a significant amount of information about a person and language analysis can be used to establish both age and gender of a given individual.

2.2 Application in social media

The task of predicting demographics of users based on their actions rose in importance with the rising popularity of social media. The more popular platforms like Facebook and Twitter got the more data their users generated, which attracted the researchers that were interested in studying human behavior. Consequently, the majority of studies related to the analysis of textual data in recent years were conducted on data from social media platforms.

Many different features were used for training machine learning models to predict the gender of the author, and the most popular ones are n-grams of words, POS (part-of-speech) tags (e.g., Cardinal number, Personal pronoun), various statistics of the text (e.g., word count, average word length) and word vector representations. In the study "Chat Mining for Gender Prediction" [22] authors are predicting the gender of the Turkish speaking user based on the chat messages from MSN and ICQ. They used stylistic features like average word count, punctuation usage, vocabulary richness and usage of smiley faces. The authors found that the use of smiley faces is a distinguishing characteristic of the female writing style and males tend to use more slang in their informal communication.

Maarten Sap et al. [23] studied textual data collected from the Facebook and Twitter, where they represented text as a collection of uni-grams, but instead of focusing on only increasing the accuracy of their ML models they worked on producing the dictionary of labeled words. They produced gender and age predictive lexica, similar to ones that are commonly used in sentiment analysis tasks. This dictionary was produced from the corpus of textual data produced by social media users on platforms like Facebook and Twitter. The lexica contain 10 797 words, where each word has an assigned weight that determines to which gender or age group the user that uses this word is more likely to belong. These weights were produced by applying linear regression and support vector machine to the corpus with associated demographic labels and extracting the feature weights from the models.

In cases when social media platform does not include gender as a field in user's profile, other fields are used to infer it. In the report "Understanding the Demographics of Twitter Users" [24], authors analyze a large number of Twitter users, to compare the geographical and demographical representation of United States in twitter and real life. They use the first name of the user to infer his/her gender by matching it to a database of common baby names from U.S. Social Security Administration. In this study, they also infer the ethnicity of the user by matching his/her the last name to a database of 2000 U.S. Census. The authors note that this approach has many limitations, the first one would be that it assumes that the name of the user is correct. Secondly, it uses only names that predominantly used by one gender and ignores the names that are popular among both genders.

The way people speak or write can define different aspects of their demographic background. In Guimarães et al. [6] study they argue that it is possible to predict the age group of a social media users by analyzing their language. They applied different machine learning algorithms to a data collected from Twitter users to predict weather user belongs to the teenage age group or an adult one. They managed to achieve an F1 score of 94% with a deep convolutional neural network with word vectors produced by the word2vec algorithm. Authors found that adults post more messages with links to other web pages (e.g., news articles, images) and use fewer slang words than users in a teenage group.

In another study [25] that had a similar task of predicting age group of Twitter users, the performance of machine learning algorithm was compared with human performance. The algorithm outperformed humans by obtaining 75.1% F1 score in an age group prediction problem, whereas humans only got 61.9%. Authors found that adults are using more complex language with longer messages and sophisticated sentence structures. The younger population, on the other hand, uses more capitalized words, first-person and second person pronouns (e.g., I, You) and they tend to have fewer messages with hashtags and links to other websites. Furthermore, authors discovered that both humans and trained algorithms underestimate the age of older users, they suggest that it is because human language changes at the beginning of life, but as a person gets older the language style stays the same.

The task of predicting the gender of an author of a tweet is one of the sub-tasks of the annual PAN International Competition on Author Profiling. The organizers of this competition provide a labeled dataset of tweets in 4 languages: Arabic, English, Portuguese and Spanish, and the task is to provide a model that achieves the best average accuracy over all languages. The overview of the 2017 competition by Francisco Rangel et al. [15] state that the number of contestants that use deep learning and vector representations of words increased in comparison to the previous year and combination of neural network models even achieved the best accuracy in the Portuguese language. However, the winning solution that managed to get an average accuracy of 82.53 % among all four languages were using SVM with character 3- to 5-grams and word 1- to 2-grams with tf-idf weighting. In their report [26], authors of winning solution of gender prediction sub-task describe various approaches that they attempted and note that by using only words that start with an uppercase letter their model showed nearly the same score as the full one while using only lowercase letters showed a decrease in accuracy by 3%. They also note that limiting words to only the ones that were used at least twice in the whole dataset, gave a significant boost to their model and decreased feature space drastically.

This section introduced the works that analyze the connection between the language that social media users use and their demographic group. The main difference between this works with the ones that were covered in the previous section is the formality of the language that they analyze. The data collected from social media is mostly an informal communication between people, before the advent of the internet researchers had limited access to this type of language.

2.3 Research of Reddit platform

The Reddit platform steadily grew in popularity in last 12 years, during that period many Reddit-related studies were conducted. In 2014 study "Evolution of Reddit: From the Front Page of the Internet to a Self-referential Community?" [27], Singer et al. analyzed the growth of the platform and shift of user behavior that occurred over time. They showed that in a period from 2008 to 2012 the number of submissions by users was growing exponentially. Authors reported that Alexa.com ranked

Reddit as 69th most popular website in the world and 27th in the US, and currently (December 2017) it is ranked 8th in the world and 5th in the US [2]. These rankings indicate that the rapid growth that platform experienced until 2014 has continued to this day. They also illustrated the change of type of content that happened over time. In the beginning, when Reddit had only a few subreddits of general topics most of the submissions contained links to outside sources (news outlets, blogs), but as the number of subreddits grew and they were becoming more specific the majority of content started consisting of text and images generated by users. Authors state that their findings suggest that the platform evolved from aggregator of the web to a large community that generates its content.

Reddit consists of subreddits with distinct topics, and many of this subreddits can provide a lot of information to different branches of science. After the 2016 US election, many political scientists and news outlets were studying the influence of social media platforms on the general population. All leading candidates had active subreddits with hundreds of thousands of subscribers and the availability of historical data of this subreddits allowed researchers track the rise and fall of candidates and try to explain the election outcome.

In a modern world, a person usually has multiple accounts on different social networks and Overdorf et al. [28] attempted to match users to their accounts based only on the style of their language. The aggregated data came from Reddit, Twitter and WordPress blogs of same people and analyzed the language that they are using on different platforms. The authors have found that finding content from the same user on the same platform is less challenging than finding it on another. Reddit to Reddit and Twitter to Twitter matching for ten users had an accuracy of 75% and for Reddit to Twitter and Twitter to Reddit matching only 36%. The topics that user discusses and the language style of the user vary considerably depending on a platform. The authors suggested that additional feature selection and usage of ensembles of machine learning models will improve the performance on this task.

Another type of subreddits that are of interest to the medical, scientific community is discussion and support groups for people that suffer from different illnesses. The study by Park et al. [29] compares three subreddits with similar topics r/Anxiety, r/Depression, and r/PTSD. The authors find that given subreddits have similarities in some of the topics like sharing emotions and discussing sleep and work-related issues.

Social scientists used data from Reddit to study the process of new word adoption. Cole et al. [30] analyzed 426 GB of textual data from Reddit to track down the creation and adoption of new words in different subreddits. The paper suggests that majority of new words are first adopted in large subreddits that have general topics and not in smaller ones that have a specific subject.

In another work that analyzes data from Reddit, authors detect a specific group of users that tend to interact with rumors. Dang et al. [31] managed to establish a cohort of users that actively interact with rumors. Authors argue that this cohort

can be divided into three groups: users that support a false rumor, users that refute the rumor and users that joke about the rumor. In this paper they have used machine learning classifier to establish that affiliation with one of the three groups mentioned above can be inferred from user's writing style. Authors conclude that their approach is suitable identifying similar groups in other social media platforms.

A subset of Reddit comments is often used as a corpus for different natural language processing tasks. Khodak et al. [32] used self-annotated comments from Reddit to gather the most extensive corpus of sarcasm. In Reddit, it is common to use special flag at the end of the message to note that it is a sarcastic remark. They published the gathered data and encouraged other researchers to use it in training and evaluation of sarcasm-detection systems.

Another work that used Reddit as a source of data for a corpus that can be used to solve a machine learning task is 'TL; DR: Mining Reddit to Learn Automatic Summarization' [33]. Authors of this research suggest using the Reddit tradition of adding a short summary after the long texts and marking it by the tag 'TL; DR', which stands for 'Too long; didn't read'. By parsing the submissions and comments on Reddit, they managed to obtain roughly 4 millions content-summary pairs. Authors shared acquired data with the scientific community by publishing it by the name Webis-TLDR-17.

In authorship attribution study Ruder et al. [34] used comments from r/Gaming subreddit to compare the performances of different models. They used deep learning models based on convolutional layers with different types of inputs to predict the author of the comment. The study showed that combination of word-level features and character-level features outperforming the baselines that were based only on one type of feature. The deep learning models also outperformed more traditional algorithms like Support Vector Machines on most of the tasks.

To analyze the roles of the users on Reddit platform Buntain et al. [35] collected data from subreddits where users ask and answer different questions. They viewed user and their activities on subreddits as a network and applied network analysis techniques to acquire insights. Authors found that majority of users are rarely active on more that one subreddit. Participants of question answering subreddits usually acquire "answer-person" role in one subreddit, and after that, all their activity is devoted to this subreddit.

The data from Reddit platform provides many possibilities for researchers from different fields. This section introduced several works that used the data from Reddit in linguistics, social science, and medicine. This master thesis offers a way of acquiring additional information about the user that can be applied in future research of Reddit users.

3 Text mining for user demographics

3.1 Background: Machine learning

This chapter will cover theoretical background that is necessary to understand the methods described in following chapters. The deep learning sub-chapter will focus on mathematical foundations and commonly used architectures and optimization techniques of neural networks. The primary focus will lie on machine learning for text processing, but other applications will also be covered.

Machine learning is a subfield of Computer science [36] that studies ways of teaching computers to perform different tasks without hard-coding their execution. The algorithms that are used in machine learning tasks are required to find patterns in provided data [37] that should help them to provide the desired outcome. For example, linear regression algorithm can be used to predict the price of a bottle of wine [38], given its age, region of production, color and other characteristics. The algorithm should be trained on a given set of examples so that it can make predictions in future.

The example above describes the task of supervised learning, which is a subfield of machine learning that includes algorithms that require labeled examples to be trained [39]. The label is usually a category or a number that describes given an example. In case of the wine example, the label is the price that algorithm is trying to predict. Two main subfields of supervised learning are classification and regression. The difference between them is that classification algorithms are predicting classes that example belongs to (e.g., type of fruit, a model of a car) and regression algorithms are predicting continuous values (e.g., price of wine, the age of a person).

Another subfield of machine learning is called unsupervised learning. This subfield solves tasks that do not require human annotated examples. A prime example of such task is clustering algorithms; these algorithms create groups of data points that are similar to each other, that called clusters. Clustering can be used to establish groups of customers of an online shop based on time and money spent on the website. Another unsupervised learning task is dimensionality reduction. It is applied to decrease the number of features in a training and test data.

Following the notation of the 'Machine Learning Techniques for Multimedia' book [40] supervised learning requires learning a mapping between input variables X and a target Y. The function f that performs this mapping $X \rightarrow Y$ is solving a regression problem if $Y \in \mathbb{R}$ and classification problem if Y is discrete. The goal of unsupervised learning, on the other hand, is learning a function g that finds patterns in input variables X without any target. The of unsupervised learning definition is somewhat generic because it has different goals depending on a problem that it is trying to solve. The clustering, for example, can be viewed as mapping X to discreet labels C without having ground truth knowledge about this mapping.

Algorithms that are used to solve machine learning tasks are quite sophisticated and versatile, so to adjust the way they approach the given problem most of them have hyperparameters. The hyperparameter of an algorithm can be viewed as parameters that are determined before the application of the algorithm. Adjustment of this parameters can lead to improvement of the performance of a given algorithm. Finding optimal hyperparameters for a model is a complex task, and it is called 'tuning' [41].

To measure the performance of different algorithms specific metrics are used. The choice of a metric is an integral part of the supervised learning because different metrics are suitable for different types of tasks and may emphasize various aspects of the machine learning task [42]. Performance of different models or the same model with different hyperparameters can be compared when the appropriate metric is chosen. A standard way to establish the performance of a chosen algorithm is to split the entire dataset into two parts and train the algorithm on one part and measure the chosen metric on the other. These parts are called training set and test set, and it is desirable that training set would be larger than the test set [43].

Another important aspect of performing machine learning task is data representation. To apply an algorithm to the input data it needs to be transformed into a numerical format, and the continues values can be converted into discrete ones [44] the categorical features like hair color or song genre should be encoded into numerical values [45]. The preparation of data for algorithms is called preprocessing, and it is applied to all the data that is used for training and testing the algorithms. Additional features can be derived from the data to be added to a feature set of the machine learning task, this process is called feature engineering, but often requires domain knowledge of a given application of the machine learning task [46].

The field of machine learning is diverse and includes numerous tasks, from image recognition to automation of robots. This thesis will mainly focus on approaches that are applied to textual data. Following subchapters will cover the theoretical background of the algorithms and preprocessing techniques that are used in this work.

3.1.1 Natural language processing

Natural Language Processing (NLP) is a field of computer science and machine learning that tackles the problem of computer understanding of the language produced by humans [47]. This field includes many tasks like machine translation, text summarization, speech recognition, sentiment analysis and many others. Moreover, the number of tasks is increasing with the advancements in the field.

The field was established in 1950-s when increasing availability of computing power allowed scientist to tackle broader problems [48]. The speech and written text is an integral part of a day to day life, and when computers started to be more accessible, the processing of text became an important task [49]. The nature of the

Text	Bag of words 1-gram vector
Andy met Ann in the zoo	[1, 1, 1, 1, 1, 1, 0, 0, 0]
Ann met John in front of the zoo	[0, 1, 1, 1, 1, 1, 1, 1, 1]
Vocabulary	[Andy, met, Ann, in, the, zoo, John, front, of]

Table 1: Example of bag of words

majority of the tasks in NLP is to automate things that any human can do, that is why while developing the systems that attempt to solve a given NLP task they often compared to human performance. Machine translation was one of the first tasks that arose in NLP field because solving it would have helped with the translation of documents which otherwise would have require the labor of trained professionals [50].

To apply machine learning techniques to textual data, it must be represented in a numerical format. Traditional approaches usually consider a single word to be the lowest level of representation of the language [51]. Many techniques of text preprocessing are focused on representing the text as a count of n-grams of words. The n-gram is a sequence n consecutive objects, in this case - words.

A standard way to approach the task of representing text in n-grams of words is called "bag-of-words" [52]. The idea behind this method is to omit the order of the words in a text and represent the text as a vector where each element represents the n-gram and value of this element is the count of a number of times this n-gram has occurred in given text. Text can be represented by a combination of different n-grams, for example, it is common to combine 1-grams and 2-grams, because it allows capturing common names like "New York". The example of an application of 'bag-of-words' is shown in table 1, two sentences are converted to numerical vectors of the same length by creating the vocabulary of all used words and counting occurrences of each word.

The limitation of this approach, besides the loss of the structure of the document, is the size of vectors required to represent the text. The "bag-of-words" approach will require additional element in a vector for every unique word, and even similar words like "driver" and "drivers" will be treated like two independent words. To tackle this problem, the process called Lemmatization is used [53]. With Lemmatization, set of rules are applied to convert the words to their base form, for example, "drivers" will be converted to "driver". This approach leads to drastic reduction of vector space required to represent the documents. Another commonly used method is the exclusion of the most common words from the corpus. The words like "the",'is" and "which" usually do not carry additional information, but inflate

the vector space when n-grams with n equal 2 or higher are used. This set of words is called "Stop words" and it can vary depending on a task and a given corpus [54].

To improve the performance of machine learning classifiers in tasks like document classification and document clustering it is desirable to give more attention to the uncommon words because they in most cases are determining the meaning of the document [55]. To do so, the term frequency–inverse document frequency (tf-idf) is calculated for each n-gram.

$$\begin{aligned} \mathrm{idf}(t) &= \log\left(\frac{|D|}{1+|\{d : t \in d\}|}\right) \\ \mathrm{tf-idf}(t) &= \mathrm{tf}(t,d) \times \mathrm{idf}(t) \end{aligned} \tag{1}$$

The corresponding values multiplied by resulting idf values in a vector representation of the document.The equations 1 demonstrate the calculation of tf-idf index for a given term t, D is a document space $\{d_1, d_2, d_3, ..., d_n\}$. The more frequent the word in the corpus, the less idf it will have, this way in the resulting vector representation of the document, rare words will have higher values.

The document can also be seen as a sequence of words that cover different topics. Most of the times documents include multiple topics, and the subfield of NLP called topic modeling is studying how to represents documents as a distribution of different topics. The idea behind most of the topic modeling techniques is that the high co-occurrence of two words means that they are likely to belong to the same topic. Latent Dirichlet allocation (LDA) is one of the most widely used topic modeling algorithms, one of the critical properties that differentiate it from other approaches is the assumption that each document is covering only a few topics and usage of Dirichlet distribution as a prior achieve it. Following the notation of the paper [56] that introduced this algorithm, the algorithm has only one hyperparameter *k* which is a number of topics. It assumes that the document can have *k* underlying topics and each topic can be represented as a multinomial distribution over the $|V|$ words in the vocabulary. And the document can be viewed as a text that has been generated by sampling the mixture of this topics and then sampling words from that mixture. For example, a document of N words is generated by first sampling θ from $Dirichlet(\alpha_1, \alpha_2, \alpha_3, ..., \alpha_k)$ distribution. The Dirichlet distribution is widely used as a prior distribution in Bayesian statistics [57], in this case, it allows to use the assumption that each document is a mix of only a few topics. After sampling the θ, the topic for each word from given N is sampled from $Mult(\theta)$ distribution $p(z_n = i|\theta) = \theta_i$. And the word itself is sampled from multinomial distribution $p(w|z_n)$. The formula 2 from the original paper, describes the calculation of the probability of the document. The $p(\theta; \alpha)$ is a Dirichlet, $p(z_n|\theta)$ and $p(w_n|z_n; \beta)$ are multinomial distributions. The plate notation of Latent Dirichlet Allocation is shown in Figure 1.

$$p(w) = \int_{\theta} \left(\prod_{n=1}^{N} \sum_{z_n=1}^{k} p(w_n|z_n;\beta)p(z_n|\theta) \right) p(\theta;\alpha)d\theta \tag{2}$$

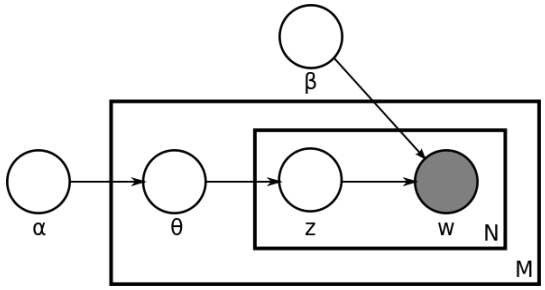


Figure 1: Plate notation of Latent Dirichlet Allocation [58]

After the training of LDA on the text corpus, it can be used to transform any document into a k-dimensional vector, where k is the parameter that represents the total number of topics. The non-negative values of the resulting k-dimensional vector will represent the distribution of the topics of a given document. The topics are latent and don't have any labels, but the most important words of a given topic can help understand an underlying structure of the topic.

In this subsection, we covered basic concepts of natural language processing that will help in understanding the in following chapters of the thesis. The task of textual data representation and existing solution of it were introduced.

3.1.2 Text classification

Text classification or document classification is one of the most important tasks in Natural Language Processing field. Its primary goal is predicting the label or a class of a given document based on words, characters or other available features. Text classification has been extensively studied, and it has been used in a various application, the most famous one would be detecting spam letters in email clients [59]. Analyzing the content of the news articles or blog posts is also often used to make predictions about the prices on stock markets and currency exchange rates.

The problem of text classification consists of two stages: text preprocessing and classification. The preprocessing stage is converting given text to a fixed representation, and the classification stage learns the class of the text based on this representation. Both steps are necessary and require a handful of design choices to produce adequate results.

The preprocessing stage was partially covered in the previous subchapter, and the classical way to approach this task was to represent the document in a vector of word counts. This way each value of resulting vector will represent presence

or absence of n-gram. The "Bag-of-words" preprocessing is mainly used with linear classifiers like Logistic Regression, Naive Bayes and Support Vector Machines. This combination showed good performances in many classification tasks and had several advantages over more complex models [60]. Linear classifiers are more interpretable than non-linear ones [61], and quite fast in test time, these aspects can play a significant role in the choice of the algorithm, because some tasks may be required to perform with restrictions on computational resources and explainability can be crucial for some problems and industries [62].

More formal definition of preprocessing step is mapping each document d_i that consists of words $\{w_1, w_2, ..., w_n\}$ to a vector $\vec{v_i}$ of fixed length. And the classification task is predicting the class c_j from a fixed set $C = \{c1, c2, c3, ...c_m\}$ based on a vector representation v_i.

$$\begin{aligned} d_i \rightarrow \vec{v_i} = [v_1, v_2, v_3, ..., v_k] \\ classifier(\vec{v_i}) \rightarrow c_j \end{aligned} \tag{3}$$

Recent advances in machine learning allowed researchers to produce more complex algorithms that exceed the performance of traditional ones [60]. These algorithms are comprised of different architectures of neural networks, and the term deep learning is also often used to describe the family of this algorithms. More detailed information about deep learning architectures and other aspects will be provided in next chapters.

3.1.3 Deep learning

Deep learning or feedforward neural networks is a method whose primary task is to map input into some target output by function approximation [63]. The approach behind the neural networks is loosely inspired by information processing by biological neurons. The lowest level of the neural network is also called neuron and represents a function that maps multiple inputs into single output [64]. Group of neurons comprise a layer of neural network, these layers are stacked on top of each other, and an output of one layer is an input of next one. The whole network can be viewed as a direct acyclic graph, where each node is a neuron, and the output of one neuron can become an input of another. Deep learning includes many types of architectures and layer types, for the simplicity this subchapter will be covering the network that consists only of fully-connected layers.

The artificial neuron is shown in Figure 2 is a basic function that maps multidimensional input into one single value. Values $x1$, $x2$ and $x3$ are the input features and $w1$, $w2$, $w3$ are the weights of the linear function, in most cases, the bias term b is also used. The equation 4 describes a process that shown in Figure 2. The function that transforms the product of linear transformation from the input values to the final output y is called activation function, and its choice is one of the important hyperparameters of the neural network. Search for best activation function is an area

Function	Equation
Linear	$f(x) = x$
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$
TanH	$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$
Rectified linear unit (ReLU)	$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$

Table 2: Activation functions

of active research and different function can be more suitable for various architectures and tasks. Commonly used [65] activation functions include sigmoid function [66], TanH and rectified linear unit (ReLU) [67]. Single neuron has a limited capacity and can be used to solve simple binary classification or regression problems.

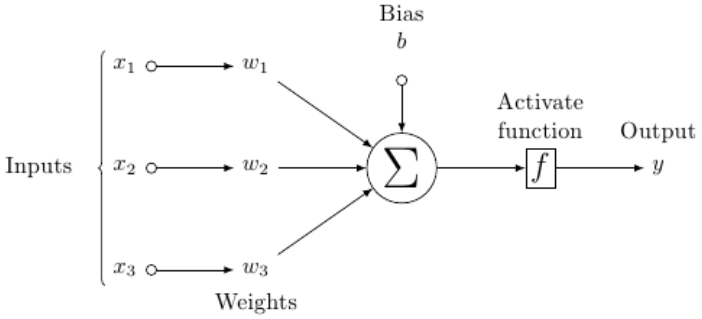


Figure 2: Diagram of neuron [68]

$$y = f\left(\sum_{i=1}^{n} x_i * w_i + b\right) \tag{4}$$

To construct a neural network multiple neurons are combined into one layer, and these layers are connected to each other. Neurons in one fully-connected layer are taking the same input and have the same activation functions, but learn different functions and therefore produce different outputs [70]. The neurons in one layer have no connections between themselves, but each neuron is connected to all neurons from previous and subsequent layers. The bottom layer of a neural network is called input layer, and it maps the provided features to k values, where k is a number of neurons in input layer. The input layer is not counted when the depth of the network is described [69], for example, Figure 4 shows 3-layer neural network with

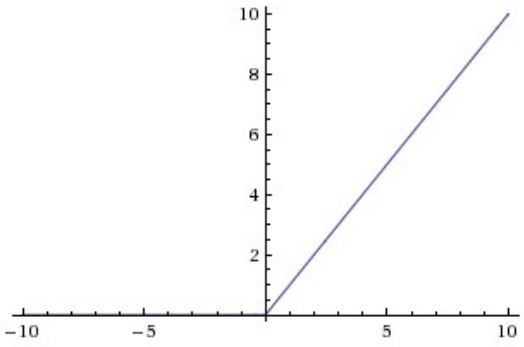


Figure 3: Rectified Linear Unit (ReLU) activation function [69]

two hidden layers. The neurons of output layer do not have the activation functions because they represent the scores of each class. All the layers between input and output layers are called hidden. The name deep learning comes from the depth of the network, the recent advancements in hardware and optimization allowed construction of networks with hundreds or even thousands hidden layers. The deeper the network, the more complex functions it can approximate.

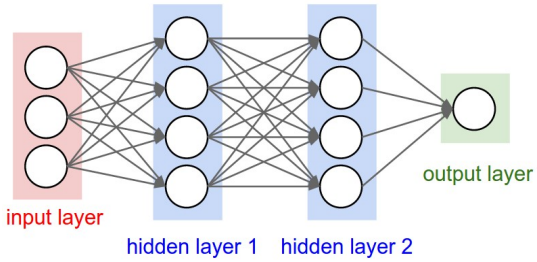


Figure 4: A 3-layer neural network with three inputs, two hidden layers of 4 neurons each and one output layer. [71]

In a multi-class classification task, the last layer usually has a different activation function that hidden layers, because the outputs of this layer should represent the probability of each possible class the $softmax$ activation function is used (equation 5). Note that the output layer should have the same number of units as the number of possible classes. For binary classification problem, the output layer will consist only of one neuron with $sigmoid$ activation. For regression problems, the last layer will also consist of one neuron, but the activation will be $linear$, so the output will not be bounded between 0 and 1.

$$\text{softmax}(z_i) = \frac{\exp z_i}{\sum_k \exp z_k} \tag{5}$$

After the construction of a neural network, all the weights in its neurons are initialized with small random numbers [71]. The network in this state will not produce any meaningful results because it has not been optimized yet. To optimize the weights of the network, we need a way of establishing how good the network performing on the target task. For that purpose, the function which is called loss function is used. For a single training example the inputs of the loss function are the vector (single value for binary classification and regression) that contains the output from the last layer of the network and the target (single value or a vector), and the output is the real value that should be minimized. In other terms, the value produced by the loss function denotes how bad the network performed its task, and the goal is to decrease this value by optimizing the weights in the neurons. The choice of the loss function depends on a task, for classification the cross-entropy loss (equation 6) is the most common choice and for the regression mean squared error or mean absolute error is widely used [71]. The loss function is another essential hyperparameter of a neural network, and even though the standard choices perform well in most of the tasks, one can define their own loss function that is more suitable for a specific task.

$$L(\hat{y}, y) = -\sum_{i} y_i \log \hat{y}_i \tag{6}$$

After establishing the loss function for a given task, we should start optimizing the weights of the network to minimize the loss.The optimization problem is a very common problem that is known in many fields [72]. The method for learning neural network weights is called backpropagation algorithm [73]. This algorithm is trying to minimize the value of a loss function given the weights and bias terms of the network. The combination of weights that produce the minimal value of loss function is considered the solution of the given task [74]. The goal of backpropagation is to compute partial derivatives $\frac{\partial L}{\partial w}$ of the loss function L with respect to all the weights in the network. This is done by computation of the gradient of the loss function at each layer iteratively, which is allowed by differentiability of used activation functions and the chain rule. The chain rule states that the derivative of a function $F(x) = f(g(x))$ for all x is $F'(x) = f'(g(x))g'(x)$ and because every layer of the neural network is a function of an output of the previous layer, this rule can be applied to calculate gradient for every weight and bias term in the network. The gradient of a given weight indicates how this weight should change to minimize the loss function.

After the calculation of gradient for the weights of the network, they are used to update the current values of this weights. The gradients are multiplied by the *learning rate* which is a hyperparameter that determines how big of an update of networks weights will be performed. The learning rate can be constant during the whole training process, but it is recommended to decay the learning rate over time [75]. Decreasing value of learning rates allows reduction of noise during training

and leads to faster convergence to local minima [75]. Another popular approaches propose an application of dynamic learning rates with the usage of momentum and computing the individual learning rate for each weight every iteration. The the most common strategies for learning rate are constant learning rate, learning rate with decay, momentum [73], adagrad [76], RMSprop [77] and Adam [78].

The neural network usually require significant amounts of data to train and to perform weight update for each training example is an expensive procedure, that is why during training the data is usually split in mini-batches the small samples from the whole training set. For example, a training set with 5000 examples and batch size of 25 will have 200 batches therefore during one epoch (a complete pass through training set) the network will have 200 weight updates. The size of the batch is also a hyperparameter of the network, the lower the batch size, the more updates it will have per epoch [75].

The standard technique to avoid overfitting and achieve the best performance of the network is to split the training set into training and validation set and use the validation set to monitor the loss and the target metric over each epoch [79]. To avoid overfitting to the training data, the number of epochs is set to a high number, and the training stopped after the performance on the validation set stopped improving for several epochs. Additionally, it is possible to save the weights of the network after each epoch if it is improved over previous best performance on the validation set. This way after the restoration of weights from the best epoch we will get the best version of our network.

This subchapter covered essential characteristics, and building blocks of neural networks, more detailed information can be found in a 'Deep Learning' book [80]. The neural network can be used in many areas, and they are proven to provide the state-of-the-art performances on many complex tasks, but this thesis focuses on implementations of neural networks for natural language processing problems.

3.1.3.1 Recurrent neural networks

Neural network architecture that was described before is quite robust and can approximate complex functions to solve a given task. One of the limitations of the network that consists only of fully-connected layers is the inability to process sequential data correctly. To tackle this limitation in 1980-s the new type of architecture was introduced, called Recurrent Neural Networks (RNN) [81]. The RNNs are employing the idea of parameter sharing, which means that the same weights of the recurrent layers are used for each timestamp.

The recurrent nature of the network means that the input values are processed by the recurrent layer one by one, therefore keeping the sequential order of the data. In the equation 7 [82] shown the calculation of the output of the recurrent layer at a timestamp t. Hidden state from a previous timestamp $h^{(t-1)}$ is multiplied by the weight matrix W and added to the multiplication of input at current timestamp $x^{(t)}$,

and another matrix of weights U and then added to a bias term b. The process of reuse of weights W and U at each timestamp is called parameter sharing. Note that the hidden state is also shared between steps, therefore the network is 'aware' of the inputs on previous steps. To update the weights of the recurrent network the special case of backpropagation that is called 'Backpropagation Through Time' [83] is used.

$$h^{(t)} = f(Wh^{(t-1)} + Ux^{(t)} + b)) \tag{7}$$

To demonstrate how RNNs work on a more concrete example, let's take the sequence of prices of some stock and try to predict the next price. In Figure 5 [84] the input values (stock prices) are shown as red rectangles, the green rectangles represent the RNN cell, and the blue box is an output. The RNN cell on each step takes in the current value and the output from the previous step, and by using the vector of internal parameters that are transmitted between the steps the network can model the temporal dependencies. Note that even though the figure shows multiple RNN cells, they represent the same layers thus the network forms the chain from the same RNN cell and processes the input sequence element by element.

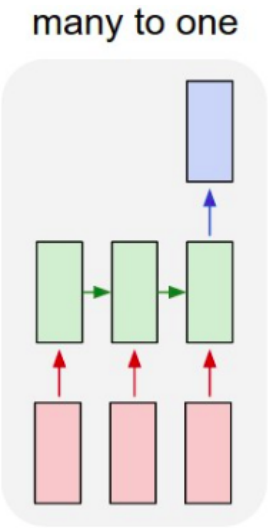


Figure 5: Example of Recurrent neural network with one output. [84]

The ability to process sequential types of data is crucial in tasks related to natural language processing [85]. Without the sequential aspect of the data, the sentences "It is a cat, not a dog" and "It is a dog, not a cat" will have the same vector representation, because they contain the same words. The sharing of hidden states between steps allows the network to "remember" the past inputs and therefore better model sequential data.

Experiments with recurrent neural networks showed that they are not feasible for modeling the long sequences. After few steps, the network started to "forget" past inputs, and as a result, its performance falls with the increasing length of the input sequence. For example in a sentence 'A cat met a dog in the park and then went home' the network can lose the dependency between an action of going home and a cat. To overcome this problem in 1997 Hochreiter and Schmidhuber [86] proposed a new network architecture based on recurrent neural networks that allow accumulating information over the passage through the input sequence. The proposed architecture is called Long short-term memory network (LSTM), and its main contribution is an addition of new layers that handle the external memory vector which is passed between the steps. In Figure 6 the horizontal arrow that goes through the upper part of the LSTM cell represents the memory cell, the yellow boxes represent the layers of the network that "decide" what information goes into the memory cell and if it should be cleared. This architecture allows modeling longer sequences and finding long-term dependencies in data.

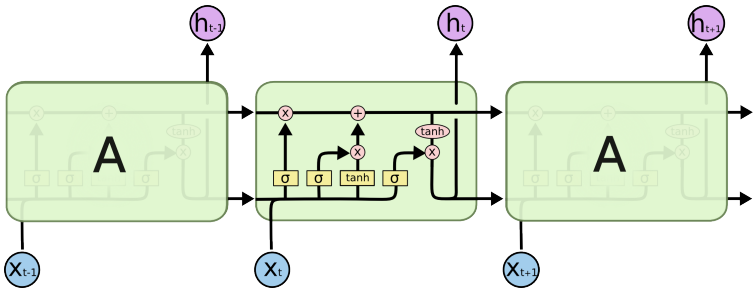


Figure 6: Example of Long short-term memory network [87]

Long short-term memory network architecture is widely used in academia and industry, the majority of tasks related to modeling the natural language or processing other sequential data involve LSTMs. There exist various modifications to recurrent neural networks and long short-term memory networks, but their characteristics are out of the scope of this thesis.

3.1.3.2 Convolutional neural networks

Convolutional neural networks (CNN) [88] were created to reduce the number of weights that need to be trained in a network for an image processing tasks. For example, to process the image of size 640x480 with three color channels each neuron in a first layer of the fully-connected network will require 921600 weights and with large layer sizes the number will grow to hundreds of million weights. Modern images have an even higher resolution, processing which will lead to a substantial number of learnable weights. Another reason of development of CNN's was that in image processing tasks low-level layers usually learn to recognize different types of

shapes and in the fully connected network, each neuron in the first layer needs to recognize all of the various shapes, which is inefficient. Convolutional layers tackle this problem by reusing the learned weights on each layer.

To tackle this limitation convolutional neural networks in image processing use convolutional filters, which are 2-dimensional layers of small size (e.g., 3x3). These filters are applied not to the whole image at once, but to the consecutive small parts of the image of the same size as a convolutional layer. A set of convolutional filters can be stacked to form a convolutional layer. By using the convolutional filters, we can drastically reduce the number of weights that are required to learn. The example in Figure 7 shows the 32x32x3 image as an input and 5-filter 3x3 convolutional layer which produces the output matrix with depth 5. Convolutional layers showed outstanding performance in image related tasks [89], and studies suggest [90] that each filter in first convolutional layer learns different low-level visual features like edges or shapes.

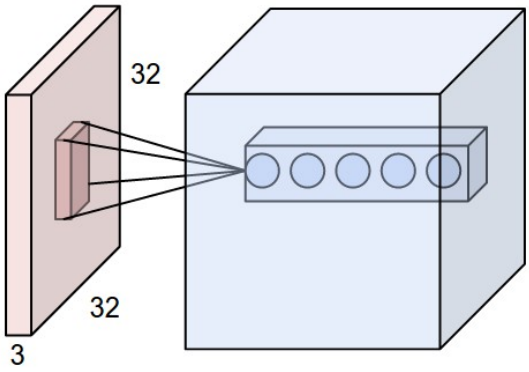


Figure 7: Schematic representation of CNN [91]

The application of convolutional layers requires few parameters that will determine the architecture of layers and their use. The shape and depth are parameters that establish the size of the layer and the total area that the layer will be covering [80], in Figure 7 the shape of the layer is 3x3, this means that the layer is 2-dimensional and will have access to only 9 input values per iteration. But how many values it will return depends on the depth, in the example above the convolutional layer has a depth of 5, therefore, will return 5 values in a 1x1x5 matrix.

The other parameters dictate the strategy of movement of the convolutional layer. The padding is a common approach to increase the input size to control the size of the output matrix of the convolutional layer, usually, it is done by adding zeros on the borders. Figure 8 shows the input of 5x5 that was zero-padded on the borders. Another hyperparameter is a number of cells we move the convolutional layer to the right over input matrix after each iteration. This parameter is called stride, in Figure 8 the black border shows the position of the 3x3 convolutional layer on a first step

and red border shows its position in the second step. The combination of padding and stride is commonly used to preserve the shape of input after the application of convolutional layer [91]. In the example in Figure 8 the 3x3 convolutional layer over zero-padded 7x7 input will return the 5x5 matrix, which is the same spatial size of the input matrix.

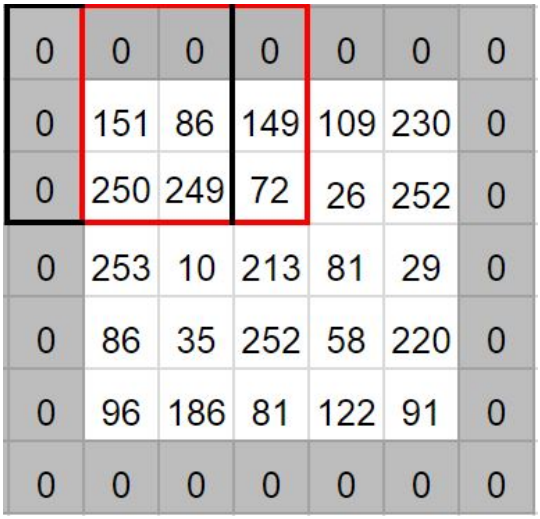


Figure 8: 3x3 convolutional layer over 5x5 input with stride of 1 and zero-padding of size 1

The convolutional neural network consists of series of convolutional layers that are added one after another [91]. The first convolutional layer takes the input data (e.g., an image with 3 channels) as an input and the next layer processes the output of the first one and so forth. Analysis of trained convolutional network showed that first layers of the network learn to detect edges and the further the layer from the input the more complex shapes it will learn to detect [92]. This ability of the network to learn complex general visual representations is employed in transfer learning. Transfer learning allows taking weights of convolutional layers of the network that has been pre-trained on large scale image classification task (e.g., ImageNet [93]) and using them in other image processing tasks (e.g., classification, object detection, image segmentation [94]) with other data.

To reduce the dimensionality of the data that flows through the convolutional network, it is common to use pooling layers [89]. Pooling layers are performing a simple operation of dimensionality reduction of activation map on width and height. The most common pooling operation is the max-pooling, which takes the input of given size and outputs the max value of it. For example, the max-pooling layer of size 2x2 will reduce 8x8 input to a size of 4x4. Another approach of dimensionality reduction is increasing the stride of the convolutional layer. The pooling layers are

added between convolutional layers; therefore the dimensionality of data gets reduced with every pooling layer which will require significantly fewer neurons from last fully connected layers.

Even though convolutional neural networks were introduced in 1980-s to tackle the problem of handwritten character recognition [88] their popularity was limited until 2010-s. The winning solution of the ImageNet [93] challenge of 2012 was a deep convolutional network, and it surpassed other contenders by a wide margin. All the winning solutions in consecutive years were built with convolutional layers. The success of convolutional neural networks on image processing tasks led to the restoration of interest in neural networks in general[95].

3.1.3.3 Word embedding

Neural network architectures like recurrent neural network or long-short-term memory networks are widely used in natural language processing tasks [85]. These networks require a sequential input of numerical values, and we can view the text as a sequence of words. To represent words as numerical values, we will use word embeddings. Word embeddings are a set of vectors that represent given words, where words that are similar by meaning will have word embeddings that are close to each other [96]. One of the simplest ways of producing the embeddings is by learning them with the network. It is possible to initialize the word vectors with small random numbers and learn them with the rest of the neural network. This approach is feasible but requires a significant amount of training data to produce reasonable embeddings, and the speed of training the network will be reduced.

A more common approach is to use specialized algorithms to produce the word embeddings. The main idea behind this algorithms is that similar words will have similar surrounding words around them. Tomas Mikolov et al. proposed an algorithm called word2vec [97] that learns word embeddings by taking k words around the given word. Authors proposed two ways of learning word representations: continuous bag-of-words (CBOW) and continuous skip-gram (see Figure 9. In CBOW the algorithm predicts the given word based on k words before and k words after it, and the order of surrounding words do not matter. In skip-gram the algorithm performs the different task, it tries to predict the center word based on the words that surround it, and it gives more weight to the words that are close to the center word. The hyperparameters for the word2vec algorithm are k - the size of the window surrounding target word and d - a number of dimensions of the word embedding vectors.

Another algorithm that learns word embeddings by word co-occurrence is Global Vectors for Word Representation (GloVe) [98]. The GloVe algorithm produces the co-occurrence matrix A for a given text corpus where entry $A_{i,j}$ is the number of times word *i* occurred in a context window of word *j*. After that, the algorithm applies

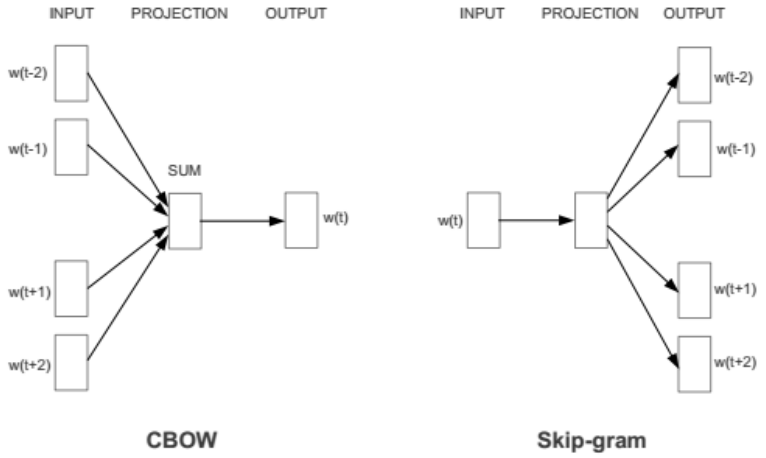


Figure 9: Continuous bag-of-words and skip gram [97]

dimensionality reduction techniques to produce word vectors of given dimensions. In practice, both word2vec and GloVe have similar performances on the same tasks.

One of the benefits of word2vec and GloVe word embedding is that they can be pre-trained on a large corpus of documents. Word embeddings that were trained on Wikipedia corpus or Common Crawl corpus are regularly used in natural language processing tasks.

3.1.3.4 Regularization Deep neural networks are very powerful in approximating complex dependencies between input and output data, but without additional regularization, they can overfit. For the model to adequately generalize, it is common to add a penalty term to networks loss function. The most common ones are L2 and L1 penalties to the weights of the network. L2 regularization adds a sum of squares of all the weights in the network multiplied by hyperparameter called regularization parameter to the loss function. This additional term in a loss function punishes the network for weights of immense magnitude and incentivize keeping them close to zero. L1 regularization is similar to L2, but instead of adding a sum of squares of weights it adds a sum of absolute values to the weights [99]. With L1 regularization the network will be inclined to have both zero weights and weights with a large magnitude. As a result, the network with L2 regularization will tend to use all the inputs where all of them have rather small importance and network with L1 regularization will gravitate towards using only the subset of all inputs but with higher importance for each. In practice, L2 regularization is used as a default because it tends to provide greater performance [71].

Another regularization technique that was introduced by Srivastava et al. [100] is called dropout. Dropout randomly turns off the subset of neurons at each training

iteration but doesn't affect neurons during test time. The probability of turning off the neuron is a hyperparameter of the dropout. Application of dropout forces the network to avoid heavy reliance on a subset of neurons because each training pass will contain a different version of the network. The use of the full model in test time can be viewed as an ensemble of all the versions of the network that were used in training time. Even though the technique was introduced somewhat recently, it has been already widely adopted because of its effectiveness.

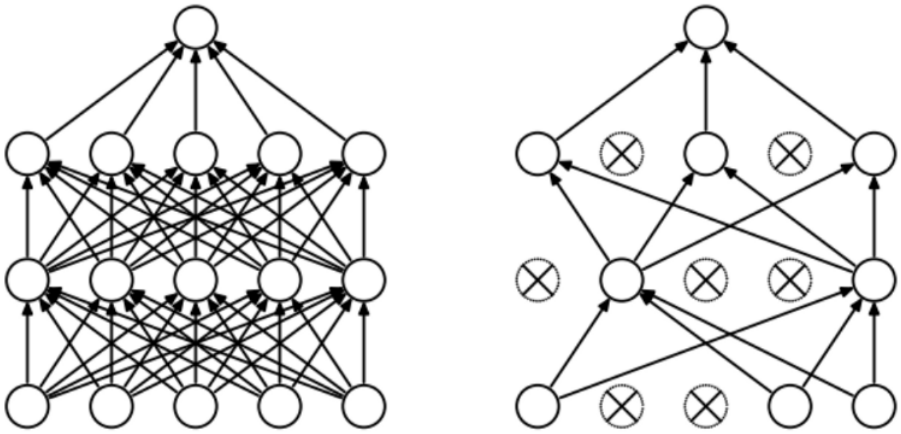


Figure 10: Illustration of dropout from the original paper [87]

Increasing the amount of training data that is available to the network is also a form of regularization because it also leads to better generalization. To produce additional data in computer vision tasks, it is common to use augmentation techniques that modify the images without losing their semantics [101]. Typical approaches are image rotation, image resize, flipping the image and adding noise to the image [102]. In case of natural language processing tasks, it is possible to use a thesaurus to swap some words with their synonyms. Augmentation allows drastically increase the volume of training data when the number of training example is insufficient.

To further improve the conversion of neural networks it is common to use batch normalization. Ioffe et al. [104] found that forcing the activation throughout the network to take on a unit Gaussian distribution [71] is beneficial for the speed of convergence. The normalization of neuron outputs occurs before activation functions and is done for each batch during training. Batch normalization also negates the consequences of the bad neuron weight initialization and allows usage of higher learning rates.

This sub-chapter has introduced theoretical concepts and terminology of machine learning. The first paragraphs covered the background of the field of machine learning and gave a general description of major sub-fields. Next section gave a detailed

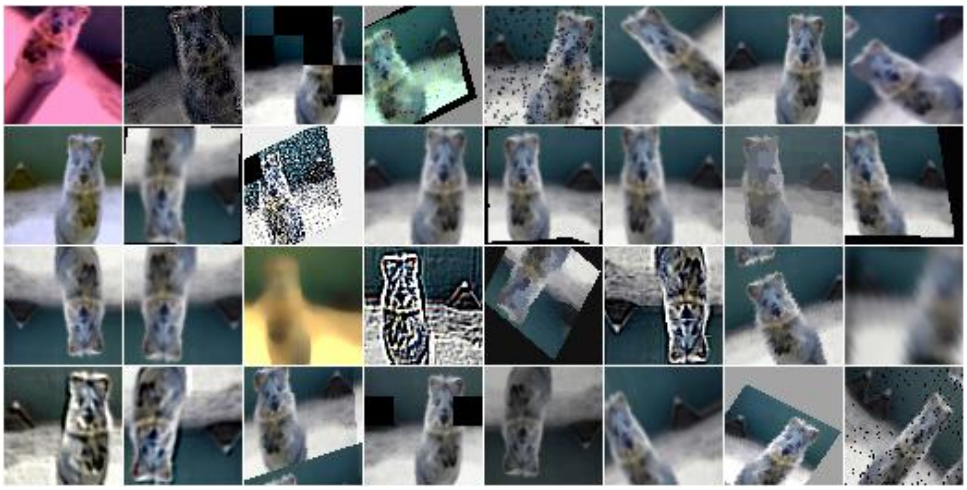


Figure 11: Examples of image augmentation from python imgaug library documentation [103]

description of natural language processing and the problem of text representation for application of text classification algorithms. After that the deep learning was introduced, the section contained information about mathematical foundations of the field as well as commonly used architectures and optimization techniques.

3.2 Methods

This chapter will introduce theoretical background of the approaches that are used to predict the gender of the user based on the text that he or she has produced. The structure of the data, its sources, and assumptions that were made during the data preparations will be covered in a separate subchapter.

Three main approaches were used in the task of predicting the gender of the author based on the generated text differ in the level of language representation and the algorithms that are used. A first approach is a standard approach of viewing the text as a bag of words, the order of words doesn't matter only word presence is accounted for. This approach will transform text of any length to a vector of fixed size, and traditional machine learning algorithms can be applied to it. The second approach will use recurrent neural networks, and the data will be represented as a matrix that is constructed from a sequence of the word vectors. The third approach will use convolutional neural networks, and the text will be viewed on character level. The sequence of characters will be represented as a matrix where each column is a one-hot encoded vector that represents the character.All three approaches have strength and weaknesses that can influence their performance.

3.2.1 Data structure

This subchapter will discuss the source and the structure of the data that has been used in this project as well as the tools that have been used to gather and store the data.

The data collected for this project comes from the BigQuery database that collects and stores all the comments that were ever posted on the Reddit platform. The database requires a Google account to access it, and free querying is limited to 1 TB per month. The tables in the database contain information about when the comment was posted, the text of the comment, the parent comment, the submission that is being commented, the subreddits where it is being published.

After running the queries that would extract all the comments of labeled users the results were written into the table of the database on BigQuery platform. To download the content of produced table for further analysis and application of machine learning algorithms The data that was extracted from this database consists of comments of Reddit users from June 2005 to July 2017. 206 million of comments from 305 thousands of users were collected, all of the users have a gender label, and 73% have an age label. The process of extraction of the demographics labels of users from the BigQuery database is described in Chapter 4.1.

Reddit platform userbase has a non-uniform distribution of demographic groups, according to Pew Research center [105] 67% of Reddit users are male, and 64% are in an age-group of 18-29 years. In the data that has been collected the share of male users is 58.1%. More balanced gender distribution can be explained by the nature of subreddits that were used to obtain labels, the source of 73% of demographic tags are submissions to relationship discussions subreddits.

Reddit originated in the United States, but over the years it became popular worldwide. Even though the platform is visited by people from different backgrounds and countries the primary language of communication remains to be English. In this project, all the collected data assumed to be written in English.

For training machine learning classifiers only text of comments and the gender label of the user that wrote this comment is used. The sample of the training data is shown in table 3, gender column was converted to the numeric format, 1 stand for female user and 0 for male. The preprocessing of textual information depends on an algorithm, and it will be described in the following subchapter.

To properly train a machine learning algorithms, the data was divided into three parts: training, validation, and test. The data is split by the user, meaning that all the comments of one user will occur only in one part of the dataset. This is done to avoid overfitting of the models to writing styles of users with a large number of comments. The test dataset contains 7.4 million comments from 6.9 thousand users, and the validation dataset includes 1.08 million comments from 2 thousand

author	body	subreddit	created_utc	gender
fleurdeliz	Pesto and tortellini. Pasta on top of pizza is the best kind of carbo-loading.	AskReddit	1387395917	1
NV_Geo	People wanting small government and a lack of government interference on how businesses operate.,Republicans, for the most part, have summarily dismissed environmental stewardship.	geology	1478718851	0
danzania	Perhaps look into Memrise? It's more flash-cardy though. I like to use it for vocab whereas Duolingo handles grammar better.	duolingo	1456227123	0
switch_bitch	Yes my first Chopin was also the Minute waltz.	piano	1482010221	1

Table 3: Example of collected data. author - the username of the comment author, body - the text of the comment, subreddit - the subreddit that the comment was posted on, created_utc - UNIX timestamp of the comment, gender - the gender label where 1 is female, and 0 is male.

users and the rest 185 million comments from 250 thousand were put in a training dataset.

3.2.2 Word-level traditional approach

The data that we collected is represented as a text of comments and gender labels of authors of this comments. The task we are trying to solve is predicting the gender of the user, based on the text of a comment that he wrote. This task can be viewed as a standard text classification task because we are predicting gender label (female or male) based on the text. The classification of the text consists of two essential parts, text representation and a choice of machine learning model.

In the text representation part, we need to choose a level of language for preprocessing the data. Two levels that are commonly used are the word-level and character-level text representation. In the word-level representation, the text is viewed as a sequence of words and a word is the lowest level of representation of the language. Different combinations of words compose a sentence, and different combination of sentences compose the whole text. In the character-level approach of text representation, the text is viewed as a sequence of characters that consist of letters, numbers, punctuation and whitespace characters (i.e., space, tab). With this approach, the machine learning model that will be chosen to perform the classification task will not have a notion of a word, and it will use only character combination to predict the class.

This subchapter will cover the application of word-level bag-of-words approach for preprocessing textual data and traditional machine learning algorithms like logistic regression and gradient boosting trees. These algorithms are widely used in the text classification tasks. But even though the step of preprocessing data will be identical for both of this algorithms, the algorithms drastically differ in their approaches in class prediction. These distinctions will be examined in detail.

The general idea behind the bag-of-words approach is in transforming any given text into a fixed-size vector. In this thesis, we used term frequency–inverse document frequency (tf-idf) approach of bag-of-words. This approach accounts for the frequency of occurrence of a chosen word in a text, and the rate of occurrence in the whole dataset. For more information about this approach see subchapter 3.1.2. The transformation of text with a tf-idf approach will lead to a vector of fixed size with continuous values. To implement the transformation mentioned above on collected data the scikit-learn [106] library from python was used. The TfidfVectorizer method of this library provides an API for efficient application of tf-idf transformation. The method should gather the vocabulary and document frequency metrics from a collection of texts, and after that, it can be applied to any other document collection to transform it to the desired vector space.

The method has several parameters that help to regulate the size of the final vocabulary and the preprocessing procedures of the text. The vocabulary size is controlled by min_df and max_features parameters. The min_df stands for minimum document frequency, for example, by setting this parameter to 5 the method will not add the words that occurred in less than five documents into the final vocabulary. And max_features parameter limits the maximal size of the dictionary to a set size. For instance, the TfidfVectorizer with parameter max_features set to 1000 will select only 1000 most frequent words if the number of words exceeds this threshold.

The preprocessing step is required to clean and standardize the text. Because the goal of tfi-idf transformation is to count the occurrences of words, the punctuation marks and other non-alphanumeric characters are omitted. Another element of preprocessing is changing the capitalization of all the words to lowercase, because we don't want to count words like 'sometimes' and 'Sometimes' twice. After establish-

ing the parameters of the TfidfVectorizer method, it is applied to the training data in a 'fit' mode. In this step, the method 'trains' on the training data and establishes the vocabulary, term frequency, and document frequency counts. The next step is using the 'fitted' TfidfVectorizer to the training, test and validation datasets. This step will produce matrices of text representation, where each row will represent one comment and each column corresponding tf-idf value of one word. Because each row will have only a few non-zero values, the matrices are converted into the sparse format from scipy library [107]. This format allows to drastically reduce the memory requirements for the storage and processing of tf-idf matrices.

Note that the TfidfVectoriser is 'trained' only on training part of the data, this is done because we want to get a machine learning model that generalizes well on a given task. That means that all the words that have not occurred in the training data are not represented in the vocabulary and therefore ignored by the machine learning models.

This approach of text representation has its benefits and drawbacks. The benefits include the straightforward approach of data representation which leads to good explainability of the machine learning algorithms that are built on top of this approach. The method is proven to be effective in different text classification tasks [60] and serves as a reliable baseline. The drawbacks of this approach include the large size of the resulting matrix, for example, the matrix that represents 10 thousand documents with the vocabulary of 50 thousand will have half a billion elements. Additionally, as it has been already mentioned the approach ignores the order of words in the text, which lead to loss of semantic structure. The morphological structure of words is also ignored, for example, words 'phone' and 'telephone' will be treated as two distinctly different entities.

Following the step of transforming the data to a chosen representation, we need to select the algorithm that will be applied. The first algorithm that is used on a tf-idf representation of collected data is a logistic regression. Logistic regression is a linear classifier that is widely used [108] in various classification tasks. Logistic regression in a binary classification problem maps the input vector to a probability of positive class by applying the sigmoid function to the product of the input values and their corresponding vectors with additional bias term. Logistic regression has been already introduced in this thesis, and the algorithm can be viewed as a single neuron from subchapter 3.1.3 with sigmoid activation function. The formula for the sigmoid function is shown in table 2.

To implement the logistic regression, the scikit-learn library is used again. The LogisticRegression method from this library provides straight-forward API for applying the algorithm to preprocessed data. This method natively supports the sparse matrix representation of scipy library that was used to save matrices from preprocessing step. The main parameters of logistic regression in scikit-learn implementation are regularization strength C and class_weight. Parameter C is an inverse of regularization, the smaller the number, the stronger regularization is used. Ad-

ditionally, the penalty norm for the algorithms weights is chosen, the choice lies between 'l1' and 'l2' norms, for more information about these norms see paragraph 3.1.3.4. The parameter class_weights is used in cases when the target value is unevenly distributed. In our data, the gender balance of the population is skewed towards males. Therefore we will use this parameter to achieve equal representation of both genders.

The linear nature of the logistic regression limits it's capacity [80] because the algorithm tries to find a hyperplane that linearly separates the classes, and some classification tasks require non-linear algorithms. But the capacity of the linear classifier can be enough for some text classification tasks [109] and because the computational cost of training and inference is small, they are commonly used as baselines. The benefit of linear classifiers is the interpretability of the algorithm. For example, if we have a text transformed to an n-dimensional vector x, where each element represents one word, the logistic regression will also have a vector of weights w of the same size. Each value in this vector w can be viewed as an importance of the chosen word to the prediction of positive class, in other words, the high positive values of w_i mean that presence of word i in the text increases the probability of the positive class. By extracting the mode of all weights of logistic regression, we can establish the words that influence the classification process the most.

Another machine learning algorithm that will be used with produced feature vectors is gradient boosting trees [110]. The algorithm creates an ensemble of decision trees that perform classification or regression, where each tree is a weak prediction model. A decision tree is a tree-like graph, where each node represents the split based on one of the features in a feature vector. Figure 12 shows an example with two trees, the nodes represent the 'decisions' that split the data into subgroups. In case of ensemble models, the trees are called weak because they are intentionally limited by depth [110].

The gradient boosting trees algorithm requires definition of loss function that can be minimized. The algorithm is called boosting because it builds decision trees iteratively. It creates the first tree and finds the data points with high prediction errors, this errors show which data points should be emphasized by the next decision tree. After building all the decision trees, their predictions of these trees are combined by weighted average, where weight depends on a performance of a tree. Different approaches of building gradient boosting trees depend on many hyperparameters and is an active area of research [112]. More detailed description of this approach is beyond the scope of this thesis.

The library that is used for gradient boosting trees in this thesis is called XGBoost [111]. The gradient boosting trees algorithm implemented in XGBoost were a part of winning solutions of multiple machine learning competitions [113], including competitions in natural language processing field. The library also works natively with scipy sparse data format and can convert it to an internal data format, called

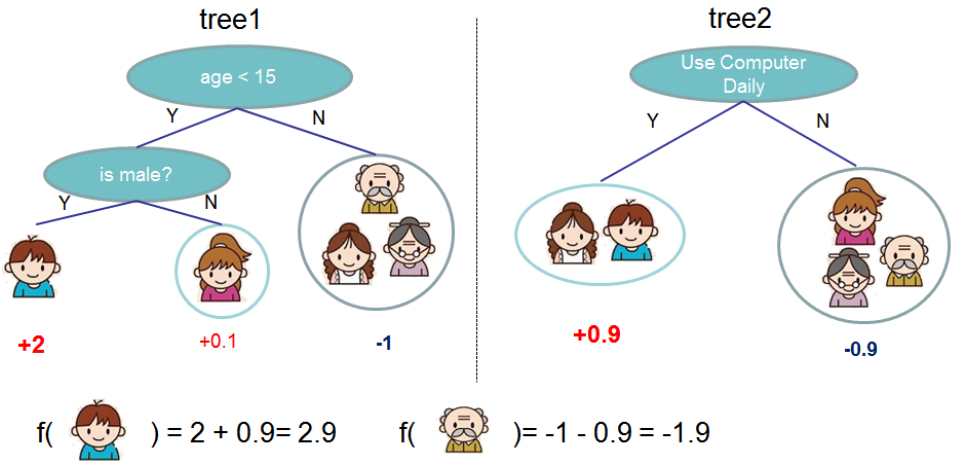


Figure 12: Example of a decision tree. [111]

DMatrix, which speeds up the training process. The main parameters of the algorithm are max_depth, num_boost_round and learning_rate. The maximal number of splits that allowed in a tree is defined by max_depth, and the number of trees in an ensemble is determined by num_boost_round. The learning_rate parameter enables regularization of the model, by forcing the algorithm to build more trees to achieve the same score. The XGBoost library also allows the usage of validation set, to stop training if the score on validation data did not improve for a given amount of boosting rounds.

The gradient boosting trees algorithm is a non-linear algorithm, which means that it can approximate more complex data, but without proper regularization can easily overfit. The non-linear nature of the algorithm means signifies that there is no linear dependency between features (i.e., words) and the target class. But because during the construction of decision trees only the most significant features are used, it is possible to sort features by their importance to a given task. Therefore, in our task, we will be able to use this functionality to extract words that had the most influence in predicting the gender based on the text of the comment.

3.2.3 Word-level neural networks

This subchapter will introduce another word-level approach of text classification that is based on neural network. It also allows preservation the sequential nature of language during data modeling step. In this approach, the text represented as a sequence of words and each word represented as a dense vector of fixed size. The classification step is performed by the recurrent neural network with fully connected layers and additional techniques. The theoretical background of recurrent neural networks and word vectors was introduced in subchapter 3.1.3.

Thee are several approaches for creating word vectors for neural networks, the most common ones are word2vec and GloVe. The critical characteristic of word vectors produced by this algorithm is that similar words will have similar vectors. This property allows neural networks to be aware of synonyms and related words. Studies showed that word vectors pre-trained on large corpora of documents can be used in various natural language processing tasks [114]. In this thesis, we will be working with GloVe word vectors that were pre-trained on a Common Crawl dataset (840 billion tokens). This word vectors were pre-trained by the authors of the GloVe, and there are 2.2 million word vectors. Each word is represented as a 300-dimensional vector that consists of dense numbers. The authors of the GloVe research paper made the choice of the size of the word vector, but it is common to choose vectors size in range 200-300 because increasing the dimensionality further provides diminishing returns and increases the memory requirements [97].

It is common to standardize the length of the text to simplify the architecture of the neural network. This way the matrix that will represent the text will have a constant size. To give an illustration, if we choose to limit the length of the text to 50 words, the matrix that represents any text will have size 50x300. Each row will represent a word with a 300-dimensional word vector, and if the length of the text is less than 50 words, the missing values will be represented by vectors of zero. In the documents with length, more than 50 only first 50 words will be used.

Note that this word-level approach suffers from the similar problems as the previous one when it comes to out-of-vocabulary words. In this approach, the words that are not present in the vocabulary will be ignored entirely. In the preprocessing step of the text after filtering out the non-alphanumeric characters, all the words are matched to the respective word id in the vocabulary. This step converts the text to an array of word id's, which is later used to construct the matrix that will represent a given document. Even words that morphologically similar to the word in vocabulary will be overlooked entirely.

As an example, let's consider a case when we have a sentence "a cat enters the house at night", and our vocabulary with corresponding 5-dimensional vectors is shown in Figure 13. If we assume that the maximal number of words is 10, then we should produce a 10x5 matrix. The first step is mapping our text to the sequence of word id's, in case of our sentence we will get [1, 4, 5, 2, 3]. Note that because the words 'night' and 'at' are not in provided the dictionary, we will ignore them. As a result first five rows of the final matrix that represent given sentence will contain the word vectors from the provided dictionary and other five values will be filled with zeroes.

After establishing the data representation step, we need to define the algorithm that will be used to perform the classification task. As it has been already mentioned, the recurrent neural network will be used. There are several variations of recurrent neural networks architecture, and in this thesis, we will be using long short-term

id	word	1d	2d	3d	4d	5d
1	a	-9.55	6.395	1.653	-1.447	-7.661
2	the	-3.943	11.608	-9.855	9.171	-4.188
3	night	-11.634	-2.062	-8.362	10.996	-2.251
4	cat	-6.837	11.159	5.24	0.858	-4.241
5	enters	-1.203	7.33	-2.638	2.609	0.541

Figure 13: Example of a word vector representation.

memory (lstm) networks. The detailed description of the design of networks and it's theoretical foundations can be found in paragraph 3.1.3.

The network consists of two blocks of layers, and the first block is a block of lstm layers that takes the text representation matrix as an input and produces the k-dimensional vector, where k is the number of lstm layers. The second block consists of fully-connected layers, the input of the first layer is a k-dimensional vector from the previous block, and the output of the last layer of this block is an m-dimensional vector, where m is the size of the last fully connected layer. The output layer will consist of one neuron with sigmoid activation function that takes in the m-dimensional vector and outputs a probability between 0 and 1. Note that the sigmoid activation function was chosen because our task includes only two classes, for multi-class classification usually the softmax activation is chosen.

To implement this neural network architecture, we use a python library called Keras [115], with another library called TensorFlow [116] as a backend. The Keras library allows construction of different architectures of neural networks that can be trained using CPUs or GPUs. Training the network with a GPU is much faster [89] than training with multi-core CPU, but requires additional hardware and software.

If the access to the server with GPU is limited, it is possible to convert all the text data into arrays of word id's beforehand. This preprocessing step will speed up the training process because the time required to construct the representation matrix will decrease.

The main parameters of this neural network are the number of lstm layers, the number of fully-connected layers and their size, the loss function, batch size and patience of early stopping mechanism. The number of layers in lstm defines the dimensionality of the output of the layer. The size and the number of the fully-connected layers define the architecture of the fully-connected block. Parameters like the loss function, learning rate, optimizer, batch size and number of epochs to train are essential to any neural network, and their detailed description can be found in paragraph 3.1.3. The patience of early stopping mechanism is a parameter that

determines how many epochs should network train after achieving the best result on the validation set. This is similar to the approach in XGBoost model when we train the model until it converges. Additionally, the regularization techniques like dropout and batch normalization can be used in selected layers.

The complex structure of recurrent neural networks allows them to model non-linear dependencies and work with sequential data. But the same complex structure makes these algorithms hard to decode, that's why sometimes neural networks are called 'black box' algorithms [117]. If in case of machine learning models that were discussed in previous subchapter we could extract at least the importance of distinct words for the classification problem, in neural networks there is no straight-forward way of extracting this information.

3.2.4 Character-level neural networks

This subchapter will introduce an approach that builds a neural network on top of a character-level representation of the data. The convolutional neural network with 1-dimensional convolutions and fully connected layers will be used to perform the classification task. The text representation and neural network architecture for this approach are inspired by 'Character-level Convolutional Networks for Text Classification' research paper [118]. The benefits and drawbacks of the character-level approach of text representation will also be discussed.

The data representation step of this approach is similar to what was described in a previous paragraph, but instead of decomposing the text to a word level and replacing words with word vectors the text is viewed as a sequence of characters. Like in a word-level approach the vocabulary is limited to a fixed set of characters, in our case, it will contain 69 characters. The character set includes English alphabet, numbers, punctuation marks and math-related characters. Because of the assumption that all the collected comments are written in English, this character set will cover all the meaningful information of the text. But instead of pre-trained word vectors for character representation, we will use one-hot encoded vectors. Therefore each character will be represented by a vector of size 70, where all values except one are zeros. The additional value in a vector is used to represent the characters that are not in the vocabulary. This way we can represent the text without missing any information.

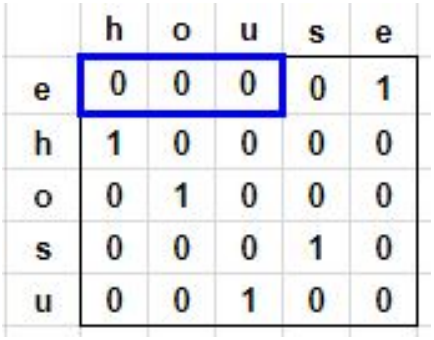
As a short example, let's say that our vocabulary consist of only English alphabet characters, therefore, each character is represented as a 26-dimensional vector. The representation of a word 'house' will be a matrix with five columns and 26 rows with all the values except five equal to zero. The values at positions (7, 0), (14, 1), (20, 2), (18, 3) and (4, 4) will be equal to one and will represent the present characters.

To simplify the architecture of the neural network in this approach we will also use the fixed size of the text length, but in this case, it will be limited to a number of characters instead of the number of words. For example, if we will decide to fix

the maximal length to 150 characters, then we will use a matrix of size 70x150. Each column represents one character, and each row represents the presence of a specific character.

The neural network architecture that is used in this approach utilizing the convolutional layers that were discussed in paragraph 3.1.3. Even though the convolutional neural network originally was primarily used for computer vision tasks, several studies showed that they could be successfully applied in natural language processing tasks [114, 119, 118, 120]. The neural networks based on convolutional layers tend to train faster than networks based on recurrent layers, because of the absence of sequential property of the network [121]. This property allows training of larger convolutional networks with the same computational resources.

The neural network architecture that is used in this approach is constructed from a sequence of 1-dimensional convolutional layers with max-pooling layers between them and fully connected layers after them. The convolutional layer that inputs the matrix of one-hot encoded characters of size k will be performing operations over k sequential character representation of a distinct character. The first iteration of the 3-dimensional convolutional layer over the word 'house' represented with 5-dimensional one-hot encoded vertical vectors is shown in a blue box in Figure 14.



	h	o	u	s	e
e	0	0	0	0	1
h	1	0	0	0	0
o	0	1	0	0	0
s	0	0	0	1	0
u	0	0	1	0	0

Figure 14: Example of an application of 1-dimensional convolutional layer over one-hot encoded word.

One of the characteristics of the character-based neural network is their large size. Because we are not using pre-trained word vectors and therefore there is no meaningful language model is provided to the network beforehand, the network would have to have more capacity to perform the same task. The size of the network is defined by the number of layers and the number of neurons in them.

The network used in this approach is inspired by the architecture that was used in 'Character-level Convolutional Networks for Text Classification' research paper [118]. The schematic representation of the architecture from the mentioned paper

is shown in Figure 16. The introduced neural network uses six convolutional layers with 256 filters each, three max-pooling layers and two fully-connected layers with 1024 neurons each. The combination of convolutional and max-pooling layers reduce the dimensionality of the input so the fully connected layers would not require a significant amount of weights. The Figure 15 shows the input shape of each consecutive layer in a task from the original paper with sequence length of 1014 characters, vocabulary size of 69 characters and classification task with 7 classes.

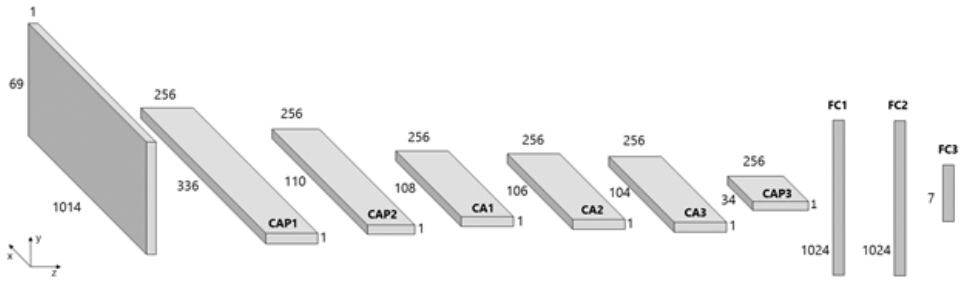


Figure 15: The input sizes of each layer. CAP stands for Convolution, Activation and Pooling. CA stands for Convolution and Activation. FC stands for Fully Connected. Everything is shown for a batch size of 1. [122]

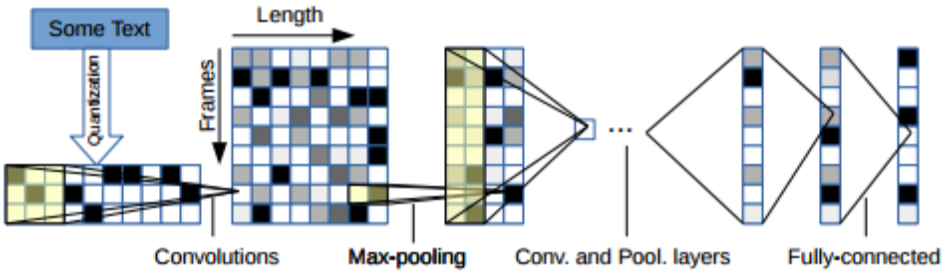


Figure 16: Schematic structure of the character-level network from the original paper [118]

With this approach, the preprocessing step is diminished to mapping the characters to one-hot encoded vectors, so it will not require additional memory or substantial computational resources. Small preprocessing step is one of the advantages of character-based approach because in previously described word-level approaches the preprocessing step required storage of large vocabulary and in word-level neural network case, all the pre-trained word vectors should be pre-loaded into memory. Simple preprocessing step reduces computational costs of inference

of the model, which can be beneficial in situations with limited access to vast computational powers.

Another advantage of character-level approaches is handling out-of-vocabulary words that are morphologically close to the ones that were 'seen' by the network in training data. In user-generated data, it is typical to have words that were misspelled, and as we mentioned earlier, this words would be ignored by the word-level approaches, because they are not present in a vocabulary. The character-level approach will account the misspelled words and might use their morphological structure to extract their meaning. This property is also valuable in natural language processing task that uses data from morphologically rich languages [123], the words in this languages are more complex and not all variations of the word can be added to the vocabulary.

On the other hand, the character-level neural network does not have a concept of a word in 'mind' and needs to learn to understand the language as a sequence of characters. Which also leads to large sizes of the neural network to increase their capacity to learn all the necessary information.

The implementation of the neural network is done in the same Keras library that is used in word-level approach. The neural network hyperparameters are similar to the ones used in a word-level approach, except for parameters that define the architecture of the network, because we are using convolutional and max-pooling layers before fully connected layers instead of long short-term memory layers.

This chapter introduced theoretical background that is required to follow the experiments that were conducted. The chapter opened with establishing the general terminology and followed with more extensive description of machine learning and natural language processing aspects. Deep learning foundations and different neural network architectures were established. Subsequently, the exact methods that were used in this thesis were thoroughly investigated.

4 Deriving user demographics on Reddit

4.1 Data

This subchapter will describe the process of extracting demographic labels of a subset of Reddit users. The rules and standard practices of different subreddits will be analyzed to establish the patterns that can lead to the acquisition of demographic labels. The data analyzed here come from BigQuery database of Reddit submissions and comments, and all following results rely on the validity of this data.

All the demographic labels that have been collected using the subreddit rules rely on data that has been self-reported by users. We also need to acknowledge that the collected sample of users is biased towards users that actively participate in subreddits that have been used to gather the demographic labels. According to the study of only communities "The participation divide: Content creation and sharing in the digital age" [124], only a few users engage with the community and majority consumes the content created by the few. If this finding is applicable to Reddit, then the majority of users are not producing any content, and therefore we cannot establish their demographics.

After establishing the sample of users that shared their demographics and downloading all the comments produced by this users, we analyze collected data. The analysis includes graphs of gender and age distribution in collected sample users, as well as a distribution of comments per user. The differences in behavior of female and male users are found and quantified.

4.1.1 Data collection and labeling

The problem of acquiring demographic labels of users on anonymous forums is a challenging one because the platform discourages users from stating their real name or any other personal information that implicitly states their demographics (e.g., first name). In user registration form, Reddit requires only username and password, and the username is essentially the only thing that is known about the user. But Reddit allows subreddits to have a certain level of autonomy, and many subreddits are using it to enforce rules and add functionality to their pages.

Every popular subreddit has set of rules, and they usually include restrictions on the type of content that can be posted, the topic of the submissions and the language that can be used in comments. Some subreddits also require users to add tags to their submissions, and this tags may include information about the type of the content (e.g., Image, Video, GIF), the general type of the submission (e.g., news, discussion, update). The administration of the subreddits can also add optional tags for the users, these tags are called 'flair,' and they are shown next to a username in each submission or comment of a user in a given subreddit. This tags usually indicate additional information about the user that can be relevant to other participants

of this subreddit, for example, a favorite hero in a game subreddits or language level in a language learning subreddit.

To find subreddits that have restrictions or recommendations for users to explicitly state their gender or age, 500 most popular subreddits were screened. The number of subscribers it has defines the popularity of the subreddit. During the screening, five popular subreddits with desired properties were found: "AskWomen", "AskMen", "tall", "relationships", "relationship_advice". Depending on the source of the demographics information this subreddits can be divided into two groups.

The first group includes subreddits that recommend their subscribers to choose a flair that states their gender. The subreddits "AskMen" and "AskWomen" are devoted to asking questions from users of a specific gender, and both of this subreddits have set of gender flairs that user can select. The Figure 17 shows the submission by user "KungFuDabu" to "AskMen" subreddit. Another subreddit that allows stating the gender through flair is "tall", it is a community of people of higher than average height. Users of this subreddit can state their height and to do so they need to choose the gender-specific background of the flair, red for females and blue for males. The example of this flair is shown in Figure 18.

Figure 17: Example of a submission in /r/AskMen subreddit

Figure 18: Example of a submission in /r/tall subreddit

The second group consists of subreddits "relationships" and "relationship_advice", both of them host questions of users that seek the advice from the community. The submission to this subreddit requires having a gender and age information about the parties of the question. For example, in Figure 19 21-year-old guy is asking for a piece of advice about his 18 years old girlfriend. Both of this subreddits recommend users to add the background information in brackets and few common patterns are used: [22M], [34/F] (29 F). The submissions with valid patterns can be extracted with an application of regular expressions.

Figure 19: Example of a submission in /r/relationship_advice subreddit

The next step after establishing the subreddits and their particular characteristics was to efficiently parse the Reddit data to establish a dataset of users with known demographic labels. Even though Reddit has an API that allows extracting the various information we decided to use the dataset of already extracted Reddit comments and posts.

To collect data for this project, we used two BigQuery databases, one of them contains the submissions that were ever posted on Reddit and the other the comments to the submissions. The database requires a Google account to access it, and free querying is limited to 1 TB per month.

The database of Reddit submissions contains information about all the submissions from December 2015 to July 2017. This database includes 33 fields that describe the submission and its author. The fields that are most important for the task of extracting demographic background are listed in table 4.

Column	**Description**
subreddit	The subreddit where submission was posted.
author	Username of the author of the submission.
title	The title of the submission that is visible on the subreddit page.
author_flair_css_clas	The flair that user has chosen in this subreddit.

Table 4: The important fields of Reddit submission database

At the moment of extracting data, the database of Reddit comments contained information about all the comments that were posted from 2005 to July 2017. The structure of the comment database is similar to the submission one, and the only significant difference is instead of the "title" field the comment database has "body" field. The "body" field contains the content of the comment, and it usually contains the textual data.

To collect gender labels of users based on flairs that they have chosen, we have used a SELECT query with GROUP BY clause, to get all distinct "username" – "author_flair_css_class" pairs. The example in Figure 20 shows the query that extracts

the author and its gender from comments of "AskMen" subreddit that was produced in 2011. Figure 21 displays the first rows of the resulting table.

Constructed queries were run against each table in the comment database with python library pandas-gbq. This library downloads results of the query of the BigQuery database and converts them into a DataFrame, which is a format of the popular python library for data manipulation pandas. The results of all queries were saved to a csv file that stored all the author – gender combinations and the name of the subreddit that was used to extract this information.

```
1 SELECT author, author_flair_css_class FROM [fh-bigquery:reddit_comments.2011]
2 WHERE subreddit = 'AskMen' OMIT RECORD IF author_flair_css_class = ""
3 GROUP BY author, author_flair_css_class
```

Figure 20: Query to extract gender labels from /r/AskMen

Row	author	author_flair_css_class
1	RaiseYourGlass	male
2	phukka	male
3	ajohns95616	male
4	projhex	male
5	CalamityJaneDoe	female
6	dontforgetpants	female

Figure 21: First rows of a /r/AskMen query result

The extractions of demographic labels from 'relationships' and 'relationship_advice' subreddits will require parsing the submission database of Reddit. The users of this subreddits are recommended to add gender and age information about themselves and other parties involved in the question that they are stating (see Figure 19). But not all users are following these recommendations, and those who do can include a type in their submission. Therefore we cannot use all the submissions from this subreddits. The titles in submissions have to be validated, and only valid submission should be collected for further extraction of the demographic background.

The use of regular expressions usually handles the task of validating textual data to specific patterns. Matching regular expressions to a text is a popular approach, and it is supported by all the primary programming and query languages. BigQuery also supports the use of regular expressions for text validation or extract data from specific parts of the text.

The example query that extracts all the submissions that match the given pattern from December 2015 is shown in Figure 22. This query will result in a table that

contains the title, author and creation date of submissions that followed the rules of the subreddit and stated their demographic background in parenthesis. The results of this queries were extracted with pandas-gbq and saved for further processing as a csv file.

```
SELECT title, author, subreddit, created_utc FROM [fh-bigquery:reddit_posts.2015_12]
WHERE (subreddit = 'relationship_advice') AND (REGEXP_MATCH(title,r'(me|Me|I|My|my|myself|Myself)\s?(\(|\[)'))
```

Figure 22: Query to extract gender labels from /r/relationship_advice

After combining the results of all queries to 'relationships' and 'relationship_advice' into one table, we deleted all the submissions from deleted accounts. Given the personal nature of questions in these subreddits, it is common to create "throwaway" accounts that will be used once and then deleted or never used again. Out of 554 thousand collected submissions, only 277.8 thousand were from non-deleted authors. Note that one person can have multiple accounts and there is no straight forward way to find accounts that belong to one person.

The next step of processing collected data is extracting the age and gender from the data in parenthesis. Even though the users in collected submissions followed the recommendations of the subreddit and stated their background information, the output they have produced is not standardized. Figure 23 shows a sample of collected data, the structure of data varies, some users put age first and data second and others do the opposite. The age and gender can also have no delimiters, or space or backslash can separate them. After extracting the values in parenthesis into a separate column in pandas DataFrame with python regular expression library we standardized the resulting values, so the age was followed by lowercase characters 'm' or 'f' that represents gender without any parenthesis. The final step was to use additional regular expression pattern matching to create a column for gender and a column for age in resulting DataFrame. For example, (18/m) will be decomposed to age 18 and gender 'm'.

title	author	subreddit	created_utc
Me [24M] my girlfriend [22F]; 20 months of a r...	JupiterDeusMaximus	relationships	1394866396
I [m/20] can't seem to make her [f/19] happy.	CrazyRide	relationships	1397218593
I (18/M) am fed up, but can't leave my girlfri...	fracati	relationship_advice	1477265475
Attraction in the friendzone, Me (19/M) going ...	18yroldMetalDetector	relationship_advice	1417681132
I [33 M] with had a pen pal [33 F] since I was...	MosesBro	relationships	1463605710

Figure 23: First rows of a /r/relationship_advice query result

The processes described above resulted in two tables, each containing information about gender and age of a specific user that were collected from two groups of

subreddit described earlier. After combining this tables into one and deleting the duplicate entries, we got 305 thousand usernames of users with the known demographic background.

The next task after establishing the subset of users with gender labels is extracting the comments of these users from the BigQuery database. Given that we have more than 300 thousand usernames, multiple tables with data (each table contains comments from one month) and the limitation on free volume by the platform we had to optimize the extraction process. After the study of documented use cases and their bandwidth usage, we decided to load the table with labeled users to BigQuery and extract the comments by an inner join of this table and all the tables from Reddit comments database. To select all the tables from a database, we used a wildcard property that allows the selection of multiple tables by passing a name of the table with a mask that contains "*". Figure 24 shows the query that has been used to produce the table of all the comments posted by the labeled users.

```
SELECT
  *
FROM ( SELECT
    author AS l_author
  FROM
    `reddit-vasilev.reddit.labeled_users`) AS lbld
INNER JOIN (
  SELECT
    body,
    author,
    score,
    created_utc,
    subreddit
  FROM
    `fh-bigquery.reddit_comments.20*`) AS rdt
ON
  lbld.l_author = rdt.author
```

Figure 24: Query to extract all comments of labeled users

The resulting table contained 206 million comments and the query has run for 1 minute 52 seconds and required 679 GB of processed data. To download the resulting table for further work, the table was compressed and transferred in 100 chunks of 200 MB to Google storage.

After downloading and unpacking of all the parts of the table the total size of the files was 45.2 GB. Loading the dataset of this size into the memory is a time-consuming operation and requires access to a computer with the vast amount of RAM. To simplify the further analysis and make the data more accessible to machines with limited memory the table was converted to an HDF5 file. The HDF5 is a data format that allows storage of large amounts of data without putting it in a database. The data in this format can be accessed the same way as data in a numpy matrix, but in case of HDF5 file, only selected data will be loaded into the memory. For example, it is possible to iterate over the rows of our resulting table without loading the whole table into the memory. However, this operation will require the constant overhead of reading the data from the hard drive.

After the examination of data, several accounts with an excessive number of comments were found. These accounts were automatically writing comments of similar nature like providing links to the website or providing a short automatic summary of a submission. To avoid overrepresentation of active users and get rid of bots among labeled users, all the comments of users with more than 100 thousand comments were deleted from the dataset. After the cleaning of the data, only 193 million comments were left.

4.1.2 Examination of data

The final dataset of Reddit comments by the users with inferred demographics contains 193 million comments from 305 thousand users. The information about the gender and age of the user was collected from five subreddits, but because the 'relationships' subreddit is more popular than others 64.9% of all the labeled users were labeled with this subreddits data (see Figure 25).

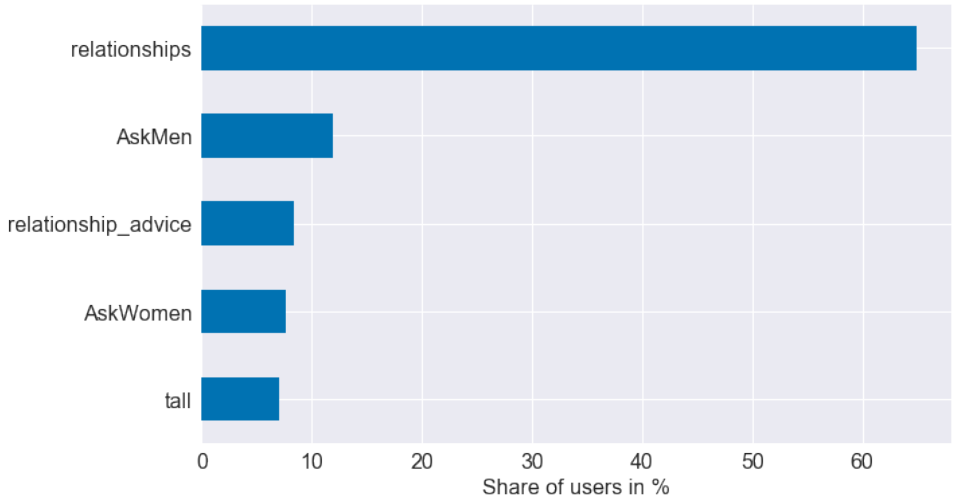


Figure 25: Distribution of labeled users by subreddits

The demographic distribution of labeled users is unbalanced, only 41.2% of users have been labeled as female. This distribution varies depending on a subreddit that has been used to extract the demographic label. Figure 26 demonstrates that the gender balance of the subreddit depends on its topic. The subreddit 'AskMen' has only 24.7% of female users, while 'AskWomen' has 65.4% of females. The most unbalanced representation of genders is found in 'tall' subreddit, where only 17.2% of labeled users are female, and the most balanced is found in 'relationships' subreddit, where 45.5% of labeled users are female. Note that the reported numbers represent only the users of this subreddits that have posted something in a specific

time window and explicitly stated their gender through flair or in a submission title.

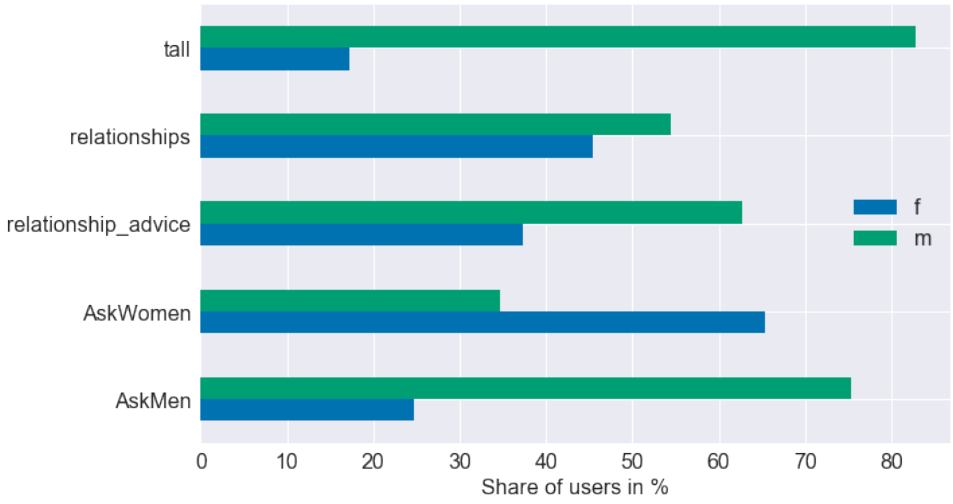


Figure 26: Distribution of gender among users collected by specific subreddit

Additional information about the age of the user is available for 73.2% of collected users because the information about the age of the user was available only in 'relationships' and 'relationship_advice' subreddits. The median age of female and male users is 23, as it is indicated in Figure 27. The age of male share of users varies more than the age of female users and the first quartile for female users is at 21 years, while for male users it is at 20 years. In general, the age distribution of collected user sample agrees with the report of Pew Research Center that states that 59% of Reddit users are in an age group of 18-29 [105].

As we mentioned before, the nature of subreddits that were used to label the users can lead to a large number of one-time (i.e., 'throwaway') accounts. Collected data showed that out of 305 thousand labeled users only 260 thousand wrote at least one comment and 238 thousand wrote at least two. The distribution of a number of comments per user among male is different from distribution among female users. The difference is demonstrated in Figure 28, which is a cumulative distribution function (CDF) plot of a number of comments for male and female users. The graph shows that the median number of comments for female users is 18 and for male users is 80 comments. The male share of users is more active when it comes to commenting on Reddit, 33% female users have at least 100 comments, while 47% male users exceed this number of comments. Therefore, in data that has been collected 41.2% of users are female, but they produced only 24% of comments in collected data.

A similar situation can be observed in the distribution of a number of subreddits user commented in. The median number of subreddits for males is 7 and for female

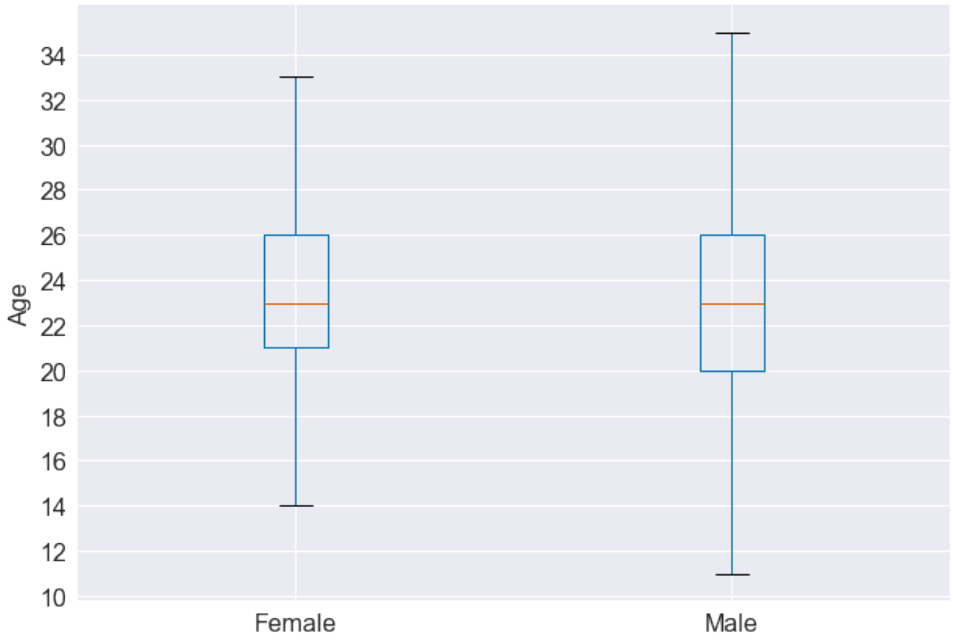


Figure 27: Distribution of age among males and females in collected data.

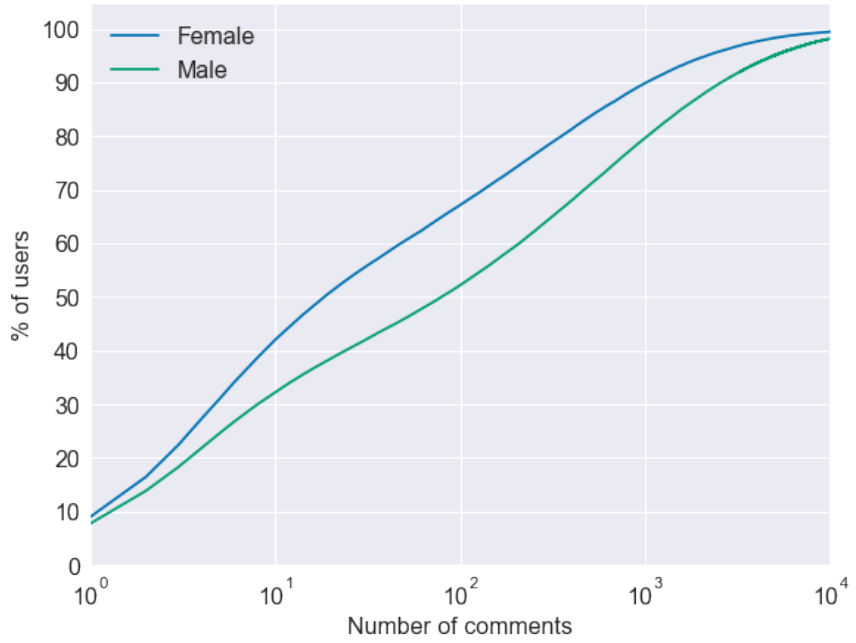


Figure 28: CDF plot of number of comments with logarithmic x-axis.

users only 2. The Figure 29 shows the cumulative distribution function plot of a number of subreddits user commented, divided by gender. Only 37.5% of female users have commented in 10 or more subreddits in contrast 55.5% of male users passed the same threshold. Not that in Figures 28 and 29 the x-axis is logarithmic, because of the long tail nature of the data.

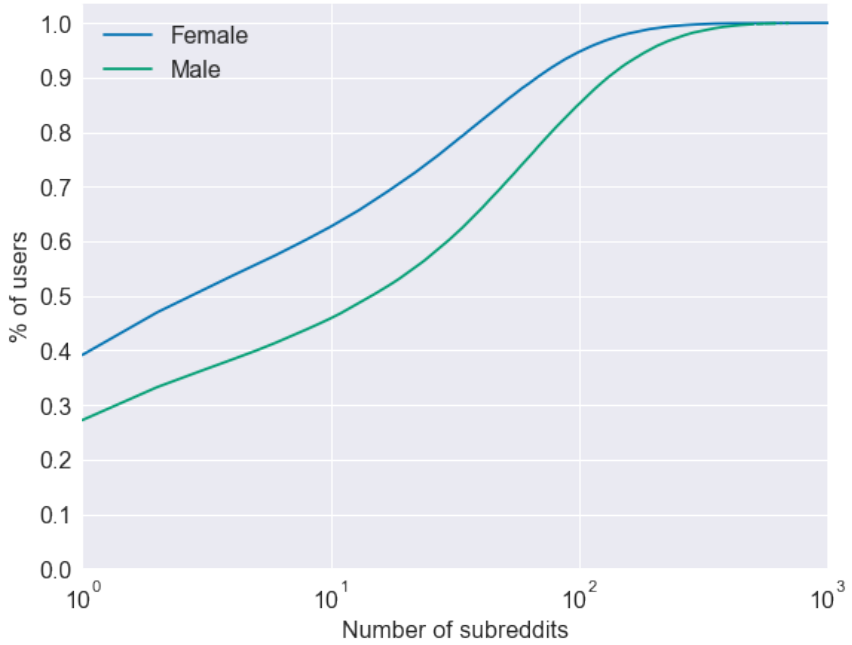


Figure 29: CDF plot of number of subreddits user commented in with logarithmic x-axis.

In this subchapter, we described the choices and assumptions that were made to collect the demographic labels of the subset of Reddit users. Additionally, the python libraries and other technologies that were used to extract, process and store the data were described. Lastly, the exploratory analysis with different plots highlighted the essential features of collected data.

4.2 Implementation and hyperparameters

This subchapter will describe the process of training of the machine learning algorithms and hyperparameters that were used in to train them. The implementation of the algorithms and challenges that were encountered will also be examined.

The computational power required for following implementation varied, depending on an algorithm and its hyperparameters. To train the algorithms, virtual machines with various computational powers were rented from Google Cloud Platform. To train long short-term neural network and character-level convolutional

neural network, the virtual machines with one Nvidia K80 GPU were used to speed up the training process.

To train the algorithms, the collected data was divided into three parts, training, validation and testing. The training part contains 185 million comments from 250 thousand users, the validation part is used to monitor the performances of the machine learning algorithm during the training process and includes 1.08 million comments from 2 thousand users, and the test data contains 7.4 million comments from 6.9 thousand users. The users sampled for validation and testing datasets were sampled with stratification to maintain the same gender balance in training and test data.

The data that is used for training and testing the algorithms includes the text of the comment produced by the user and the gender of this user. The goal of the machine learning algorithm is to predict the gender based only on the text of the comment of the user. The preprocessing and representation of the text depends on the classification approach, more detailed description can be found in subchapter 3.2.

The traditional word-level approach was implemented with python scikit-learn and XGBoost libraries. Due to the limitations of random-access memory, the traditional word-level approach was trained on a subset of training data that contained 30 million comments.

The term frequency-inverse document frequency preprocessing of the data was performed on unigram representation of the data. The minimum document frequency was set to 50, and the smoothing of idf was implemented to prevent zero divisions. This step resulted in a vocabulary of 108 thousand words, therefore the training data, after the transformation had 30 million rows and 108 thousand columns. The stop words that are usually omitted from the input data in text classification tasks were kept because previous research indicated that personal pronouns and other parts of speech usage could vary among females and males [16] and consequently these words can be important features for the machine learning algorithm.

The logistic regression application is straight-forward and it serves as a baseline to compare other algorithms. The algorithm from scikit-learn library was trained with 'l2' regularization with class_weights parameters passed, to account for unbalanced distribution of genders.

The gradient boosting trees approach that is based on the same tf-idf features are implemented with XGBoost library. To select the best hyperparameters for this algorithm the sample of 1 million comments from training data was used to perform a grid search of hyperparameters. The final hyperparameters for the model are shown in Table 5, the min_child_weight, and max_depth parameters were found with grid search, and remaining hyperparameters were set according to the classification task and balance of classes in training data.

Hyperparameter	Value
min_child_weight	1
eval_metric	logloss
scale_pos_weight	2.5178
max_depth	8
objective	binary:logistic

Table 5: Hyperparameters of XGBoost

The application of long short-term memory network required a virtual machine with a GPU, but because these machines are significantly more expensive than regular ones, the batches to train the network were prepared beforehand on a virtual machine without a GPU. This step reduced the time required to prepare a batch for a training step, hence decreasing the cost of the whole training process. Because the neural network is trained on mini-batches of data, the entire training dataset should not be pre-loaded into the memory. This feature allowed us training the neural networks on all 185 million comments available in training data. The batches were generated by randomly selecting the comments from an hdf5 file, which contained already preprocessed training data. The number of words in each comment should be fixed for the representation purposes, and in our implementation the comments were limited to 50 words. The analysis of the random sample of the training data showed that 82% of the comments have less than 51 words.

The long short-term memory network consists of several parts which are connected to each other sequentially. The first part is an embedding part, this a step where the text that is represented as a list of word indexes from a dictionary is converted to a matrix, where each row is a word vector from GloVe pre-trained vectors. The second step is a long short-term memory block, which in our case contained 256 layers and additionally had a dropout of 36 % and TanH activation. The third part consists of a fully-connected layer with 512 neurons with rectified linear unit activation, dropout of 25% and batch normalization. The activation of an output layer was a sigmoid function, and the loss function was binary cross-entropy because we are solving a binary classification problem. For the optimization the stochastic gradient descent with learning rate 0.01 was used. The batch size was set to 1024, according to the memory capacity of the GPU.

The validation part of the dataset was used to monitor the performance of the network after each epoch. The patience parameter was set to 2 epochs, which means that the network was training until the validation score did not improve for two

epochs. The weights of the network from the epoch with the best score on validation set are saved for future use.

The training process of the character-level convolutional network is similar to the one described above for long short-term memory network. The train part of data was preprocessed on a virtual machine without a GPU and saved as an hdf5 file to decrease the overall costs of training. The length of the texts was fixed to 150 characters, and this is done because convolutional neural networks work only with fixed size input. The examination of the same sample that was used for long short-term memory network showed that 67% of the comments have less than 151 characters.

The preprocessing step and the architecture of the network were inspired by the work of Xiang Zhang et al. [118]. The network consists of seven 1-dimensional convolutional layers with 256 filters each, the first two layers have kernel size of 7, while the rest have kernel size of 3. The max-pooling layer added after first, second and seventh convolutional layer. Furthermore, two fully-connected layers added after the convolutional block with 1024 neurons each and a dropout of 50%. The rectified linear unit activation function was used for all the layers except the output layer, where the sigmoid function was used. For the optimization the stochastic gradient descent with learning rate 0.01 and momentum 0.9 was used. The batches were generated by randomly sampling preprocessed comments from an hdf5 file with the batch size of 1024. The validation dataset and the patience were used precisely like in long short-term memory network.

Algorithm	**Size of training data**	**Training time**	**Virtual Machine**
Logistic regression	30 million	2 hours 15 minutes	16 CPU cores 104 GB RAM
XGBoost	30 million	30 hours 56 minutes	16 CPU cores 104 GB RAM
LSTM network	185 million	43 hours 48 minutes	4 CPU cores 26 GM RAM 1 Nvidia K80 GPU
Char CNN	185 million	101 hours 40 minutes	4 CPU cores 26 GM RAM 1 Nvidia K80 GPU

Table 6: Comparison of algorithms with respect to the size of training data, duration of training and used virtual machines.

One of the challenges of generation batches from an hdf5 file for a neural network is the access time. Because the data is stored on the disk and not pre-loaded into the memory, accessing 1024 elements with different indexes to generate a batch could take up to 5 seconds. And considering that each epoch contained 180 thousand batches, it was unfeasible to use this approach. To tackle this problem, we generated the batches from two separate parts that included 512 comments, and each part

contained 512 consecutive comments from the random part of the training dataset. And these two parts were concatenated to create one batch of size 1024. Even though the batches were generated not completely randomly, it allowed to speed up the training process significantly. The training time and characteristics of used servers for the mentioned algorithms is shown in Table 6.

4.3 Comparison of the models

This subchapter will go over the results of selected models and compare their performances on a chosen test dataset. Additionally, the subchapter will point out the design choices that were made and the reasons behind them. Moreover, the approaches of data representation or machine learning algorithm selection that did not provide positive results will be discussed.

4.3.1 Performance of the models

The performance of the machine learning models will be compared on a test dataset that contains 7.4 million comments from 6.9 thousand users. The subreddits that were used to label the users were omitted from the data. The models were trained on a training dataset, and a more detailed description can be found in a previous subchapter. Note that because of memory limitations of the available servers, only neural networks were trained on the full dataset of 185 million comments, while other algorithms were trained only on 30 million comments. This constraint limits the extent to which these models can be compared, but we argue that the machine learning model can adequately generalize on 30 million examples.

To compare the performances of selected models, we will be using an F1 score as a metric because the classification problem that we are solving has unbalanced classes. This metric is commonly used [125, 126, 127] to evaluate the performance of different machine learning models, especially when the classes are unbalanced.

The machine learning models were trained on a comment-level, meaning that we are predicting the gender of the author of the comment based only on the text of one comment. To aggregate the predictions of all the comments, we will be using the probability predictions of the model instead of hard predictions. This way each comment of the user will result in a prediction of a continues value between 0 and 1, where 1 is a probability of 100% that the comment was produced by the female user and 0 is a probability of 100% that the male user generated the comment. To aggregate probabilities of all the comments of the user, we will use a simple unweighted average of probabilities of all the comments of the user and then use the threshold of 0.5 to predict the gender class. For example a user with 5 comments that had probabilities 0.4, 0.9, 0.7, 0.5, 0.7 will have an average of 0.64, therefore, the user will be classified as female.

The performance of the machine learning models both on comment-level and user-level is shown in Table 7. The results show that chosen approaches perform poorly on a comment level, but aggregating the predictions to user-level drastically improves the performance of the models. As can be seen in Table 7 the character-level convolutional neural network outperforms all other models on user-level, even though it has only second to best performance on comment-level. The long short-term memory network with word vectors from GloVe underperformed on both user-level and comment-level, which is an unexpected result because it is a standard neural network architecture for solving the natural language processing tasks. The traditional word-level approaches showed reasonable performance on both user-level and comment-level predictions. The Logistic regression even has the best performance on the comment-level and only slightly behind the character-level network on user-level. Furthermore, we added a baseline of random prediction for comparison and an accuracy score for user-level predictions.

Algorithm	**F1 score comment-level**	**F1 score user-level**	**Accuracy user-level**
Logistic regression	**49.22**%	81.35%	84.89%
XGBoost	48.03%	81.30%	85.16%
LSTM network	47.07%	78.69%	82.41%
Char CNN	48.69%	**82.33**%	86.31%
Random prediction	21.27%	38.22%	53.84%

Table 7: The F1 score and accuracy of the selected algorithms on a test dataset. The user-level score is calculated based on the average probability of all the comments of the user. Random prediction is generated by random permutation of ground truth information.

The confusion matrices of predictions of four selected machine learning models are shown in Tables 8 - 11. All four models are overestimating the share of the female users, the number of male users classified as female is 2.23 time higher in case of predictions of LSTM and only 1.64 higher in case of character-level CNN. This behavior can be explained by the hyperparameters that were used to account for unbalanced classes in a dataset, and this hyperparameter boosts the importance of correctly predicting female class because it is underrepresented in data.

To establish that the predictions of the character-level convolutional neural network are not better than the baseline predictions of logistic regression by a chance we run the statistical test. We used non-parametric Wilcoxon signed-rank test with

a null hypothesis that two predictions are from the same distribution. The resulting p-value of $< 1.0 * 10^{-25}$ indicates that two predictions are not sampled from the same distribution and the character-level network is better than the baseline not by a chance. Some researchers suggest that using a statistical test to compare classifiers on only one domain is insufficient [128], but we argue that it gives more merit to our claim that character-level network outperforms the available baseline.

Actual \ Predicted	Male	Female
Male	3598	740
Female	306	2281

Table 8: Confusion matrix of logistic regression

Actual \ Predicted	Male	Female
Male	3665	673
Female	354	2233

Table 9: Confusion matrix of XGBoost

Actual \ Predicted	Male	Female
Male	3497	841
Female	377	2210

Table 10: Confusion matrix of LSTM

Actual \ Predicted	Male	Female
Male	3748	590
Female	358	2229

Table 11: Confusion matrix of character CNN

One of the possible explanation of exceptional performance of the character-level approach is the informal nature of the comments, written on Reddit. The comments often contain the typos or words that intentionally misspelled, which means that the word-level approaches will not account these words. Additionally, the capacity of the character-level network was significantly larger than the long short-term memory network, the character-level has around 3 million weights, while the word-level network has 700 thousand weights. The difference in capacity is caused by the nature of the approaches, in the word-level network, the meaning of the word is encoded in a 300-dimensional pre-trained vector, while in the character-level network this meaning has to be learned by the network itself.

The user-level prediction aggregates the predictions from all the comments of the users, but the number of comments can range from one to tens of thousands. To establish how the number of comments influences the F1 score of the model we divided users into groups depending on the number of comments and calculated the F1 score for each group. For the demonstration of the difference between the established user groups, we will use the character-level neural network, which achieved

the best performance on user-level of whole test dataset. The Figure 30 shows the scatter plot that demonstrates the dependency of an F1 score on the number of comments a user has. The y-axis displays the F1 score of the user group, the size of the circle represents the size of the user group, the color indicates the share of female users in the group and the numbers in parenthesis denote the range of the number of comments of the user group. The intervals are half-open, meaning that the number of comments of the user should be more than left value of the range and less or equal the right value of the interval. The data that has been used to produce the graph is shown in Table 12, with additional information about the accuracy. The data in this table serves as an excellent illustration of why the F1 score is a useful metric for problems with unbalanced classes. With the increase of a number of comments, the share of female users drops, and the accuracy continuously increases, while the F1 score decreases after a certain point.

Figure 30: Information about user groups that were divided by the number of comments they have. Range denotes the number of comments a user has to have to be part of a group. F1 score and accuracy demonstrate the performance of the model for a given group. The number of users shows how many users are in a group. Prediction is done by character-level CNN.

The graph shows several trends that determine the F1 score of the group. The first visible dependency is the positive correlation between the number of comments and the F1 score, for example, users with 5 or fewer comments have an F1 score of 65%,

Number of comments	F1 score	Accuracy	Number of users	% of female users
(0, 5]	64.66%	62.95%	950	44.84
(5, 10]	78.35%	76.14%	352	49.15
(10, 50]	81.73%	81.34%	1018	48.33
(50, 100]	86.57%	88.00%	600	42.50
(100, 500]	91.03%	93.16%	1755	37.95
(500, 1000]	88.37%	93.20%	735	29.80
(1000, 10000]	88.61%	94.75%	1391	24.23
(10000, 57545]	81.08%	94.35%	124	15.32
Entire test set	82.33%	86.31%	6925	41.2

Table 12: Information about user groups that were divided by the number of comments they have. Number of comments denotes the range of number of comments a user should have to be in a given group. F1 score and accuracy demonstrate the performance of the model for a given group. Number of users shows how many users in a group. Prediction is done by character-level CNN.

while users in a (5,10] group have an F1 score of 78% and so on. The F1 score reaches its peak at a user group (100,500] and slight declines in groups with a user with more comments. This trend is expected because the more data we have about the user, the more precise will be our predictions. The last group of users with at least 10001 comments has a low F1 score in comparison to groups with fewer comments, this can indicate that some accounts with high activity are used by multiple persons, and this leads to misclassification. The second dependency is a decreasing share of a female with the increase of the number of comments required for the group. This dependency was already introduced in Figure 28, where we showed that female user tends to write fewer comments.

The evaluation of models performed on a single test set instead of a commonly used cross-validation [129] approach because of the limited amount of credits that were available on Google Cloud Platform. The cost of servers with GPUs narrowed our ability to experiment with the architectures of the neural network and their hyperparameters on a full training dataset. We applied the 5-fold cross-validation to the training data with the use of logistic regression, because of it's inexpensive com-

putational costs. The results of this cross-validation were similar to the ones that we obtained on a test set, but it does not guarantee that this is the case for all machine learning algorithms.

The results indicate that the gender of the user can be predicted with a high degree of precision. This is especially the case if the user wrote at least 50 comments. The character-level network outperformed other approaches on user-level, but the traditional word-level approaches managed to obtain high scores, even though they were trained on a $1/6$ of the data that were used to train neural networks. This performance illustrates that the linear models like logistic regression are still viable approaches to solve natural language processing tasks.

4.3.2 Analysis of negative results

This subchapter is meant to highlight the design choices that turned out to be detrimental to the performance of the models and were omitted. The final choice of machine learning algorithms and neural network architectures were made after series of experiments on a sample data. The experiments described in this section were conducted on the sample of 7 million comments that was split into training and test parts. This data was used for the search of the optimal machine learning models.

Neural network architecture variations. Even though the training of neural networks is computationally expensive, by using the sample we managed to rent cheaper servers that do not have GPUs to experiment with neural network architectures with reasonable training time. One of the design choices that we had to make is to select the pre-trained vectors for word-level long short-term memory network. The authors of GloVe research paper [98] offer word embeddings of different dimensionality and source. The choice was between 300-dimensional vectors that were trained on a common crawl corpus with 840 billion tokens and 200-dimensional vectors that were trained on 2 billion tweets with 27 billion tokens. Even though the vectors that were trained on Twitter had fewer data to train on, we reasoned that the text produced by Twitter users should be similar to the comments on Reddit because of the informal nature of communication on both platforms. But the results showed that in two long short-term memory networks with the same architectures, but different word vectors the network that used 300-d vectors achieved an F1 score on a test dataset 2% higher than the network that was trained on 200-d vectors trained on Twitter. The possible reasons for this result are the difference in dimensionality (300-d vs. 200-d) and the size of the training data for word vectors (840 billion tokens vs. 27 billion tokens). It is possible that collecting additional data from Twitter and increasing the dimensionality of the vectors would lead to better results, but this task it out of the scope of this thesis.

As it has been mentioned before, the character-level neural network requires the fixed size of the input. In this experiment, we wanted to establish how increasing the number of characters will influence the performance of the neural network. We

trained two neural networks where the inputs were fixed to 150 and 200 characters and compare their F1 scores on a test dataset. Surprisingly, the network trained on shorter comments outperformed the other network by 0.8% even though the number of weights in a network with 200 characters was 3.2 million while the network with shorter input had only 2.95 million weights. Maybe the same effect as in the weighted mean experiment influenced the performance, ignoring the longer comments carries the regularizing impact on the performance of the model.

Other experiments with neural network architecture on a sample data that led to poor results: a combination of convolutional layers and long short-term memory layers on word-level, long short-term memory network on a character level and using the character-level convolutional network with more convolutional layers.

Aggregation of predictions to user-level. The predictions of the machine learning models are made for each comment and afterward aggregated to the user-level by taking the average of all the probabilities for a user. One thing that was attempted is to use the weighted average of the predictions, based on the length of the comment. The idea behind this approach is that some comments include only a few general words (e.g., 'Thanks', 'My bad') and giving more importance to more extended comments of the user will produce better results. This approach was only used with XGBoost model because the neural networks have the fixed input size. This approach led to decrease of the F1 score by 1% on a test set of the sample. One possible explanation is that the even weights of each comment produce regularizing effect, especially for users with the high number of comments. Traditional machine learning algorithms like support vector machines, k-nearest neighbors, and decision trees were also used on a selected sample. They did not provide competitive results when compared to XGBoost.

Additionally, we tried to represent the comments of the user as one large text, by concatenating them. This way each user would have only one long text that will be represented as one vector in tf-idf space. Again, this approach was not viable for neural networks and was used with XGBoost. This approach led to the decrease of an F1 score of 1.5%, which led us to believe that simple averaging of comment-level probabilities will produce best results.

These experiments show the reasoning behind some of the design choices made in this thesis, but they do not reveal the full scope of all possible design choices. The experiments were still limited by the training time and computational resource required to run them. The future work can go deeper into the selection of the most suitable machine learning model for this task. Even though we can not claim that the results obtained on a chosen sample would apply to the whole dataset, but we argue that they provide useful pointers.

4.4 Illustration of model application

This subchapter will demonstrate how machine learning algorithms can be used to analyze content generated on Reddit platform. In the first part, we will get a sample of comments from different subreddits to infer the gender of users, based on this comments. This approach will showcase how to estimate the gender balance of the subreddit community. The second part will showcase how to leverage linear classifiers like logistic regression to better understand what words are more frequently used by male and female users.

To demonstrate how the machine learning classifiers perform on the sample of comments from specific subreddits, we will get a random 10 thousand comments from selected subreddits. Because the comments are sampled randomly, we will not have the true gender labels and will have to rely on our machine learning algorithms to infer this data. The Table 13 shows the chosen subreddits with their categories. The three groups are explicitly chosen to have either a female majority, a male majority of the users or a balanced userbase, additionally, each group has five subreddits. The choice of subreddits is based on our subjective understanding of their description.

The random sample of 10 thousand comments per subreddit was collected from BigQuery tables from May to July 2017, totaling 150 thousand comments. The character-level convolutional neural network was used to predict the gender of the user based on each comment, this model was chosen because of it's performance on a user-level prediction. This approach can be viewed as a subreddit-level prediction because we are trying to establish the gender balance of the users that comment on specific subreddit. Additionally, we collected a random sample of 40 thousand comments that were created in July 2017 from all of the Reddit so that we could use the gender predictions of this sample as a benchmark for chosen subreddits. The Figure 31 demonstrates the distribution of predicted gender, where red dotted line shows the gender distribution of a random sample of 40 thousand comments at approximately 25% and the black dotted line shows the 50% threshold.

The inferred gender balance of chosen subreddits shows that in most of the cases we obtained an expected outcome. The prediction of character-level neural network placed the female share of all the subreddits from the female-majority group above 60%. In the male majority group, all the subreddits female percentage was at most 40%, except for 'MensRights' subreddit, where it was at 42%. The group of balanced subreddits had a mixed distribution of gender, while most of them had a majority of male users, which expected considering the overall gender balance of Reddit, the 'tattoos' subreddit had a 53% female share. Additionally, the 'investing' subreddit, which was in a group of balanced subreddits had a modest 12% female share, which is even lower than the benchmark 25% female share of the entirely random sample of the comments. Only 25% of comments from the random sample from entire Reddit were predicted as female-written, which is lower than the 41.2% share of female users of the labeled user data. One of the explanations can be that female users tend

to visit only specific subreddits, consequently, their presence on Reddit, in general, is limited.

In the second part of this subchapter, we will examine the machine learning model that we trained previously and analyze how it can be used to explain the classifications. We will use the ELI5 python library [130], which is an open-source library that can help to explain the predictions of classifiers from popular scientific libraries. This library can extract weights of linear classifiers and also user LIME algorithm [131] to demonstrate the prediction process of non-linear classifiers. Because the logistic regression outperformed the XGBoost implementation in our test case, we will be using it to visualize the prediction process on our data.

The prediction of logistic regression is calculated by the sum of the bias term and the multiplication of weights of all the words present in the comment and their respective tf-idf weights. Because in our case 1 represented the female user and 0 represented a male user, the positive weight of the word will mean that the appearance of the word in a comment will count towards the female class and the negative weight will count towards the male class. The bias term of logistic regression that was trained on 30 million labeled comments was -0.498, which means that the total sum of the product of weights of appearing words and their tf-idf weights should exceed 0.498 to be classified as a comment by the female author. The Figure 32 demonstrates the sample of words with high or low scores, where a score is the product of the tf-idf weight of the word and its logistic regression weight. Some observable trends in this sample are that the appearance-related words are present for both male and female users (e.g., 'eyeshadow', 'chinos'), the 'husband' and 'wife' words are anticipated, because the appearance of these words will strongly indicate the gender of the author.

One interesting finding in this logistic regression weights was that the misspelling of commonly used words is a strong indicator of a male author. For example, words like 'mabye','probally' and 'altough' have a score lower than -0.75, which indicates that these misspellings were used mainly by male users. And given that during preprocessing we omitted all the words that were used less than 50 times, these misspellings were not rare. The Figure 32 shows only a polarizing group of words out of 108 thousand words in a vocabulary, while the majority of other words have scores that are close to zero.

The ELI5 python library can also be used to visualize the prediction over the text. The Table 14 shows examples of visualization of some comments with ELI5 with additional information about the probability and the subreddit where the comment was posted. The color indicates the sign before the logistic regression weight, and the intensity illustrates the magnitude of the weight.

This subchapter introduced two ways of leveraging the trained machine learning algorithms to understand better the data collected from Reddit platform. In the first approach, we used a character-level neural network to establish the gender balance

of selected subreddits. With the second approach, we displayed the usage of ELI5 a python library for classifier explanation, which works natively with textual data.

Subreddit name	Subreddit description
Female majority	
MakeupAddiction	Subreddit devoted to discussion and demonstrating makeup techniques.
TheGirlSurvivalGuide	Community with advice for females from females.
femalefashionadvice	Fashion tips for females.
TrollXChromosomes	Community of female Reddit users that discusses general topics.
xxfitness	Community for female and gender non-binary users who are fit or want to be fit.
Male majority	
bodybuilding	Subreddit that is devoted to bodybuilding.
MensRights	Community for those who wish to discuss men's rights.
malefashionadvice	Fashion tips for males.
malelivingspace	Subreddit for pictures of male apartments and homes.
wicked_edge	Discussions about shaving with straight, double edge or injector blade razors.
Balanced	
investing	Discussions of investments strategies and stock market news.
lifehacks	Subreddit for sharing of uncommon solutions to common problems.
personalfinance	Discussions and tips on how to better manage your money.
tattoos	Subreddit for posting personal tattoos and tattoo paintings.
travel	Community about exploring the world.

Table 13: Subreddits chosen for demonstration of performance with their descriptions and groups.

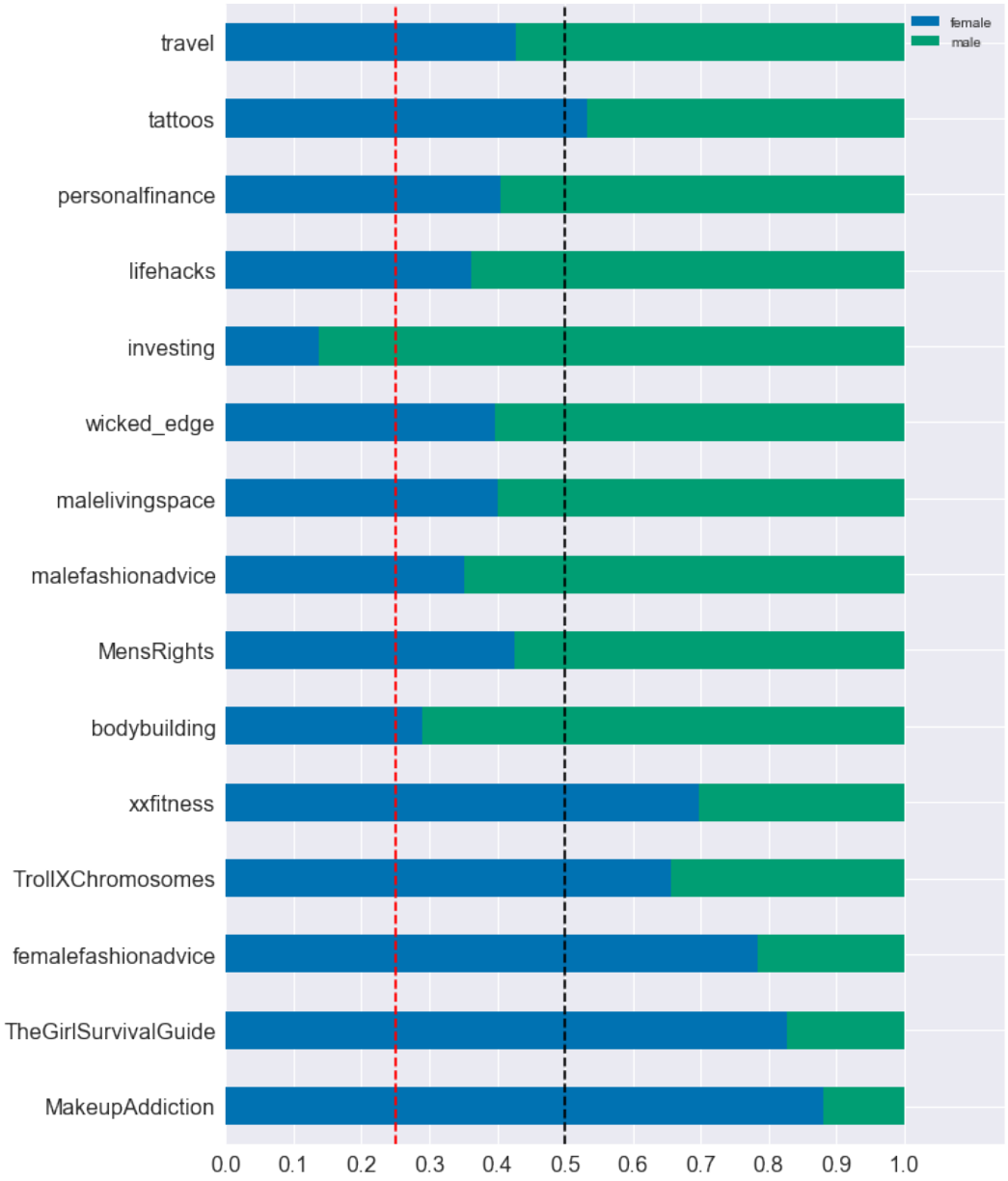


Figure 31: The inferred gender balance of chosen subreddits. Red vertical line denotes the gender balance of random sample of 40 thousand comments from all of the Reddit and black vertical line displays the 50% balance.

Contribution	Feature	Contribution	Feature
+2.110	eyeshadow	-0.434	barbers
+1.375	husband	-0.445	railways
+1.126	pilates	-0.461	mechanically
+0.986	amaaaazing	-0.470	homeruns
+0.918	childcare	-0.474	firearms
+0.831	earrings	-0.477	opponent
+0.720	wishlist	-0.478	github
+0.705	rescues	-0.483	uppercuts
+0.651	fertility	-0.522	wifes
+0.591	erotica	-0.536	ferrari
+0.590	conventionally	-0.574	guitarists
+0.555	heartbreaking	-0.605	understating
+0.538	pregnant	-0.657	awnser
+0.515	piercing	-0.788	mabye
+0.491	lovely	-0.821	probally
+0.437	therapy	-0.866	altough
+0.377	cute	-1.423	chinos

Figure 32: A sample of words with high positive or negative logistic regression weight. Words with positive weights indicate the female class and with negative weights male class. The misspelled words are added to demonstrate the increased number of typos among male users.

Subreddit name	**Probability of female**	**Visualization**
politics	29.9%	i certainly hope so. so far, trump has proven that presidents of the united states and their administrations need not be held to any sort of ethical standard.
Rainbow6	30.2%	'on a scale of 1 to invading russia in winter, how bad is your idea?'
technology	31.2%	nasa tried to build one several years ago but they switched to developing an indoor version first. i guess the automatic navigation was harder than they thought - plus it makes recharging it much easier and they won't lose it if it makes a mistake.
itookapicture	61%	my jaw actually dropped when i saw this, feels like i'm on a month long ocean voyage!
worldnews	63.6%	so all he needs is a positive influence in his life. baby hands just didn't have that growing up, and now he looks for it everywhere in the wrong places.
engineering	63.9%	yeah, and while i was studying i was sometimes unsure if i did the right thing but i always find some comfort in the fact that i could always go back.

Table 14: Examples of text visualization with the ELI5 library. The probability shows the prediction of the logistic regression. Green color indicates positive (female) weight of the word and red color indicates negative weights.

4.5 Discussion of limitations

This subchapter will discuss the assumptions that were made during the collection of data and application of machine learning classifiers, and what limitations these assumptions impose. These limitations can offer insights into directions of future work that can extend current research.

One of the first assumptions that were made in this thesis is that the gender is one of the characteristics that influence the language pattern of the person. Even though many studies [16, 17, 18] suggest that it is possible that the psychological state of the person carries more importance than the gender [20, 21]. Additionally, we assume in this thesis, there are only two genders, which narrows the scope of this research to traditional views on the gender and ignores non-binary options. Moreover, while collecting all the comments produced by the user, we assume that the gender is a constant trait of a person, which is inaccurate in some cases.

Another limitation of this study is the assumption that users provide accurate information about their demographics in subreddits that we chose to label the data. Additionally, the subreddits that were selected to label the data all have similar subjects related to giving advises about the personal life of users. One can argue that users that only visit these types of subreddits will not be representative of all Reddit population. Furthermore, when we mention a user, we assume that there is one person that uses one Reddit account, which is impossible to confirm. It is plausible that there exist 'group accounts' where a group of people uses one account together. The opposite situation is also possible, one person can user several accounts and a situation when the Reddit accounts are created for one comment or one submission is a common occurrence and this one-time accounts are called 'throwaway' accounts.

One of the hyperparameters that we used while training machine learning classifiers is 'class_weight', which is used to give more importance to underrepresented classes. The values passed in this hyperparameter were chosen for the assumption that the data contains only 41.2% of female users, which will result in the model that will be giving more emphasis to the female class. It is possible that the gender balance of the labeled users is not representative of the whole population and future uses of the trained model should account for that. Additionally, the gender balance is not a constant value and can change over time, which also limits the applicability of the trained models in the future.

Additionally, during the bag-of-words preprocessing steps of traditional word-level approaches we used only single words to limit the size of the vocabulary. Addition of higher order n-grams might improve the performance of classifiers because it will include the common phrases and named entities. Character n-gram combinations can also be used as additional features to existing word n-grams.

The large size of the collected data and the limited computational resources available is another restraint that did not allow us to investigate the broader scope of

machine learning models. With additional resources, we could have trained neural networks with the larger capacity or experiment with different architectures that might have achieved a higher F1 score in gender-prediction task.

This subchapter stated a few limitations of this study that mostly based on the assumptions that had to be made for the research purposes. Further exploration of these topics can serve as a good start for future research in this area.

5 Conclusions and future work

This thesis focused on establishing a way to collect the demographic data about Reddit users and leverage capabilities of recent advances in machine learning to understand the demographics of Reddit platform. To establish the most fitting machine learning algorithm and neural network architecture we run various experiments on a sample data. The process of training neural networks on a large corpus of Reddit comments required significant computational resources of servers with graphical processing units. To rent these servers we used Google Cloud Platform, and it's Nvidia K80 GPUs. The extraction and storage of massive amounts of data were also done with BigQuery database and Google Storage, which are a part of Google Cloud Platform. Additionally, we considered different approaches to text representation for solving the task of predicting gender based on a Reddit comment. Three main approaches were a bag-of-words for traditional machine learning models, word vectors for recurrent neural networks and one-hot encoding of characters for the convolutional neural network.

Another noteworthy element of this thesis is a novel strategy of obtaining demographic labels of a subset of Reddit users. We used the internal rules of several Reddit communities (I.e., subreddits) that dictate the style requirements of submissions to extract the gender and age of users that posted in this subreddit. This approach allowed us to collect the gender labels of 305 thousands of users and after retrieving all the comments that this users ever wrote on Reddit, we assembled a dataset of 206 million comments. To our knowledge, this is the largest dataset of Reddit users with a demographic label and the largest dataset of labeled Reddit comments.

The main findings that answer with the research questions of this thesis are:

1. The self-reported demographic attributes like gender and age can be obtained by taking advantage of rules of the specific type of subreddits.

2. Character-level neural network showed the best performance on the task of predicting the gender of Reddit user based on his/her comments. The more comments user produced, the more accurate the network was.

3. The gender balance of collected data is skewed, only 41.2% of labeled users were female. Additionally, female users tend to post fewer comments in fewer subreddits than male users.

The best machine learning classifier that was based on character-level convolutional network achieved an F1 score of 82.33% on a test set of 7.4 million comments from 6.9 thousand users. Grouping the users by the number of comments that they produced showed that the classifier could achieve an F1 score of at least 85% for a group of users with 51 or more comments. The approaches with tf-idf preprocessing and logistic regression or gradient boosting trees showed slightly worse performance with an F1 score of 81.3%. But given that it was achieved with six times fewer

data that was used to train a neural network this result show that this approach is a reliable baseline that can be used in environments with limited computational capacities (e.g., embedded systems).

The superior performance of the character-level network can be explained by the nature of the content generated by users on Reddit. The language on Reddit most of the times is informal, which leads to the use of abbreviations and slang words, which are entirely ignored by the word-level approaches because this words won't be present in a dictionary if they were not used often enough. Additionally, the misspelled words can be ignored by the word-level approaches, while the character-level model will handle any sequence of characters.

To validate that the machine learning algorithms trained on the data are performing as expected, we collected 10 thousand comments from 15 different subreddits. Theses subreddits were divided into three groups by the expected gender balance of their userbase: the female majority, male majority, and even gender balance. The groups were formed by our subjective understanding of the subreddits topic. The results showed that the balance of gender in predictions of character-level neural network reflects our expectations, in the subreddits from the female majority group most of the comments were predicted to be produced by the female user and vice versa. Additionally, the random sample of 40 thousand comments was taken from the pool of all the subreddits, and only 25% of them were predicted to be written by female users. This finding can indicate that even though the female users are present on the platform, they tend to be active only in specific subreddits.

One of the possible directions of future work is extending the machine learning approach to predicting the age of the user. Most likely predicting the age group of a user will be more feasible than exact age, because without knowing the exact birthday of a user it is impossible to know his/her exact age. Alternatively, the presented gender prediction approach that is based solely on the text of the comments of the user can be extended to account other platform-specific information like the subreddit that the comment was posted in, a number of subreddits the user have published in, the average length of the comment and many more. These additional features possibly will improve the performance of suggested machine learning models. Additionally, more subreddits can be analyzed to establish ones that can provide demographic information about the user, therefore boosting the number of labeled users and making the sample more representative.

The code of the machine learning algorithms and pre-trained versions of algorithms can be found on `https://github.com/SomeSnm/master_thesis`. Additionally, the repository contains the SQL queries that were used to extract the data from BigQuery tables and the list of labeled users with their demographic labels.

References

[1] *Reddit mediakit January 2017*. URL: https://reddit.zendesk.com/hc/en-us/articles/205183225-Audience-and-Demographics (visited on 01/13/2018).

[2] *Alexa.com reddit.com statistics*. URL: https://www.alexa.com/siteinfo/reddit.com (visited on 01/13/2018).

[3] *New reddits by month*. URL: http://redditmetrics.com/history/month (visited on 01/13/2018).

[4] *Frequently Asked Questions about Reddit*. URL: https://www.reddit.com/wiki/faq (visited on 01/13/2018).

[5] Lizhou Zheng et al. "Predicting age range of users over microblog dataset". In: *International Journal of Database Theory and Application* 6.6 (2013), pp. 85–94.

[6] Rita Georgina Guimarães et al. "Age Groups Classification in Social Network Using Deep Learning". In: *IEEE Access* 5 (2017), pp. 10805–10816.

[7] Jun Ito et al. "What is he/she like?: Estimating twitter user attributes from contents and social neighbors". In: *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*. IEEE. 2013, pp. 1448–1450.

[8] Mary Ritchie Key. *Male/female language: With a comprehensive bibliography*. Scarecrow Press, 1996.

[9] Robin Lakoff. "Language and woman's place". In: *Language in society* 2.1 (1973), pp. 45–79.

[10] Jennifer A Simkins-Bullock and Beth G Wildman. "An investigation into the relationships between gender and language". In: *Sex Roles* 24.3-4 (1991), pp. 149–160.

[11] Olivier Y de Vel et al. "Language and gender author cohort analysis of e-mail for computer forensics". In: (2002).

[12] Gareth JF Jones et al. "Experimental ir meets multilinguality, multimodality, and interaction". In: *8th International Conference of the CLEF Association, CLEF*. Vol. 2017. Springer. 2017, pp. 11–14.

[13] David Pearce and H Sofia Pinto. *STAIRS 2016: Proceedings of the Eighth European Starting AI Researcher Symposium*. Vol. 284. IOS Press, 2016.

[14] Prasenjit Majumder et al., eds. *FIRE'17: Proceedings of the 9th Annual Meeting of the Forum for Information Retrieval Evaluation*. Bangalore, India: ACM, 2017. ISBN: 978-1-4503-6382-2.

[15] Francisco Rangel et al. "Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter". In: ().

[16] Shlomo Argamon et al. “Gender, genre, and writing style in formal written texts”. In: *TEXT-THE HAGUE THEN AMSTERDAM THEN BERLIN-* 23.3 (2003), pp. 321–346.

[17] Matthew L Newman et al. “Gender differences in language use: An analysis of 14,000 text samples”. In: *Discourse Processes* 45.3 (2008), pp. 211–236.

[18] Malcolm Corney et al. “Gender-preferential text mining of e-mail discourse”. In: *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*. IEEE. 2002, pp. 282–289.

[19] James W Pennebaker and Lori D Stone. “Words of wisdom: Language use over the life span.” In: *Journal of personality and social psychology* 85.2 (2003), p. 291.

[20] James W Pennebaker, Martha E Francis, and Roger J Booth. “Linguistic inquiry and word count: LIWC 2001”. In: *Mahway: Lawrence Erlbaum Associates* 71.2001 (2001), p. 2001.

[21] James W Pennebaker et al. *The development and psychometric properties of LIWC2015*. Tech. rep. 2015.

[22] Tayfun Kucukyilmaz et al. “Chat mining for gender prediction”. In: *Advis*. Vol. 4243. Springer. 2006, pp. 274–283.

[23] Maarten Sap et al. “Developing age and gender predictive lexica over social media”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1146–1151.

[24] Alan Mislove et al. “Understanding the Demographics of Twitter Users.” In: *ICWSM* 11 (2011), 5th.

[25] Dong Nguyen et al. “" How Old Do You Think I Am?" A Study of Language and Age in Twitter.” In: *ICWSM*. 2013.

[26] Angelo Basile et al. “N-GrAM: New Groningen Author-profiling Model”. In: *arXiv preprint arXiv:1707.03764* (2017).

[27] Philipp Singer et al. “Evolution of reddit: from the front page of the internet to a self-referential community?” In: *Proceedings of the 23rd International Conference on World Wide Web*. ACM. 2014, pp. 517–522.

[28] Rebekah Overdorf and Rachel Greenstadt. “Blogs, twitter feeds, and reddit comments: Cross-domain authorship attribution”. In: *Proceedings on Privacy Enhancing Technologies* 2016.3 (2016), pp. 155–171.

[29] Albert Park, Mike Conway, and Annie T Chen. “Examining thematic similarity, difference, and membership in three online mental health communities from Reddit: a text mining and visualization approach”. In: *Computers in Human Behavior* 78 (2018), pp. 98–112.

[30] Jeremy R Cole, Moojan Ghafurian, and David Reitter. "Is Word Adoption a Grassroots Process? An Analysis of Reddit Communities". In: *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation*. Springer. 2017, pp. 236–241.

[31] Anh Dang et al. "Toward understanding how users respond to rumours in social media". In: *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*. IEEE. 2016, pp. 777–784.

[32] Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. "A Large Self-Annotated Corpus for Sarcasm". In: *arXiv preprint arXiv:1704.05579* (2017).

[33] Michael Völske et al. "TL; DR: Mining Reddit to Learn Automatic Summarization". In: *Proceedings of the Workshop on New Frontiers in Summarization*. 2017, pp. 59–63.

[34] Sebastian Ruder, Parsa Ghaffari, and John G Breslin. "Character-level and Multi-channel Convolutional Neural Networks for Large-scale Authorship Attribution". In: *arXiv preprint arXiv:1609.06686* (2016).

[35] Cody Buntain and Jennifer Golbeck. "Identifying social roles in reddit using network structure". In: *Proceedings of the 23rd International Conference on World Wide Web*. ACM. 2014, pp. 615–620.

[36] *Machine Learning*. URL: `https://www.britannica.com/technology/machine-learning` (visited on 01/13/2018).

[37] "Glossary of Terms". In: *Mach. Learn.* 30.2-3 (Feb. 1998), pp. 271–274. ISSN: 0885-6125. URL: `http://dl.acm.org/citation.cfm?id=288808.288815`.

[38] Orley Ashenfelter. "Predicting the quality and prices of Bordeaux wine". In: *The Economic Journal* 118.529 (2008).

[39] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[40] Pádraig Cunningham, Matthieu Cord, and Sarah Jane Delany. "Supervised learning". In: *Machine learning techniques for multimedia*. Springer, 2008, pp. 21–49.

[41] Kaibo Duan, S Sathiya Keerthi, and Aun Neow Poo. "Evaluation of simple performance measures for tuning SVM hyperparameters". In: *Neurocomputing* 51 (2003), pp. 41–59.

[42] Nada Lavrač, Peter Flach, and Blaz Zupan. "Rule evaluation measures: A unifying view". In: *International Conference on Inductive Logic Programming*. Springer. 1999, pp. 174–185.

[43] Isabelle Guyon. "A scaling law for the validation-set training-set size ratio". In: ().

[44] Michal R Chmielewski and Jerzy W Grzymala-Busse. "Global discretization of continuous attributes as preprocessing for machine learning". In: *International journal of approximate reasoning* 15.4 (1996), pp. 319–331.

[45] Ian H Witten et al. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[46] Pedro Domingos. "A few useful things to know about machine learning". In: *Communications of the ACM* 55.10 (2012), pp. 78–87.

[47] Gobinda G Chowdhury. "Natural language processing". In: *Annual review of information science and technology* 37.1 (2003), pp. 51–89.

[48] Elizabeth D Liddy. "Natural language processing". In: (2001).

[49] Bill Manaris. "Natural language processing: A human-computer interaction perspective". In: *Advances in Computers*. Vol. 47. Elsevier, 1998, pp. 1–66.

[50] Peter F Brown et al. "A statistical approach to machine translation". In: *Computational linguistics* 16.2 (1990), pp. 79–85.

[51] Ronan Collobert et al. "Natural language processing (almost) from scratch". In: *Journal of Machine Learning Research* 12.Aug (2011), pp. 2493–2537.

[52] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. "Understanding bag-of-words model: a statistical framework". In: *International Journal of Machine Learning and Cybernetics* 1.1-4 (2010), pp. 43–52.

[53] *Stemming and lemmatization*. URL: https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html (visited on 01/13/2018).

[54] Christopher Fox. "A stop list for general text". In: *Acm sigir forum*. Vol. 24. 1-2. ACM. 1989, pp. 19–21.

[55] Gerard Salton and Christopher Buckley. "Term-weighting approaches in automatic text retrieval". In: *Information processing & management* 24.5 (1988), pp. 513–523.

[56] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[57] Charles E Antoniak. "Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems". In: *The annals of statistics* (1974), pp. 1152–1174.

[58] Wikipedia contributors. *Latent Dirichlet allocation*. [Online; accessed 16-feb-2018]. 2017. URL: https://en.wikipedia.org/w/index.php?title=Latent_Dirichlet_allocation&oldid=817442908.

[59] George Forman. "An extensive empirical study of feature selection metrics for text classification". In: *Journal of machine learning research* 3.Mar (2003), pp. 1289–1305.

[60] Armand Joulin et al. “Bag of tricks for efficient text classification”. In: *arXiv preprint arXiv:1607.01759* (2016).

[61] Berk Ustun and Cynthia Rudin. “Methods and models for interpretable linear classification”. In: *arXiv preprint arXiv:1405.4047* (2014).

[62] Bryce Goodman and Seth Flaxman. “European Union regulations on algorithmic decision-making and a" right to explanation"”. In: *arXiv preprint arXiv:1606.08813* (2016).

[63] H. Mhaskar, Q. Liao, and T. Poggio. “Learning Functions: When Is Deep Better Than Shallow”. In: *ArXiv e-prints* (Mar. 2016). arXiv: 1603.00988 [cs.LG].

[64] Jürgen Schmidhuber. “Deep learning in neural networks: An overview”. In: *Neural networks* 61 (2015), pp. 85–117.

[65] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. *Commonly used activation functions*. URL: http://cs231n.github.io/neural-networks-1/#actfun (visited on 01/13/2018).

[66] George Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[67] Vinod Nair and Geoffrey E Hinton. “Rectified linear units improve restricted boltzmann machines”. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[68] *Diagram of an artificial neural network*. URL: https://tex.stackexchange.com/questions/132444/diagram-of-an-artificial-neural-network (visited on 01/13/2018).

[69] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. *CS231n lecture notes: Neural Networks Part 1*. URL: http://cs231n.github.io/neural-networks-1/ (visited on 01/13/2018).

[70] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), p. 436.

[71] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. *CS231n lecture notes: Neural Networks Part 2*. URL: http://cs231n.github.io/neural-networks-2/ (visited on 01/13/2018).

[72] Andreas Griewank. “Who Invented the Reverse Mode of Differentiation?” In: *Documenta Mathematica, Extra Volume ISMP* (2012), pp. 389–400.

[73] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), p. 533.

[74] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

[75] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. *CS231n lecture notes: Neural Networks Part 3*. URL: http://cs231n.github.io/neural-networks-3/ (visited on 01/13/2018).

[76] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive subgradient methods for online learning and stochastic optimization". In: *Journal of Machine Learning Research* 12.Jul (2011), pp. 2121–2159.

[77] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. *Neural Networks for Machine Learning-Lecture 6a-Overview of mini-batch gradient descent*.

[78] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[79] Yoshua Bengio. "Practical recommendations for gradient-based training of deep architectures". In: *Neural networks: Tricks of the trade*. Springer, 2012, pp. 437–478.

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

[81] John J Hopfield. "Neural networks and physical systems with emergent collective computational abilities". In: *Proceedings of the national academy of sciences* 79.8 (1982), pp. 2554–2558.

[82] Zachary C Lipton, John Berkowitz, and Charles Elkan. "A critical review of recurrent neural networks for sequence learning". In: *arXiv preprint arXiv:1506.00019* (2015).

[83] Paul J Werbos. "Backpropagation through time: what it does and how to do it". In: *Proceedings of the IEEE* 78.10 (1990), pp. 1550–1560.

[84] Andrej Karpathy. *The Unreasonable Effectiveness of Recurrent Neural Networks*. URL: http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[85] Tomáš Mikolov et al. "Recurrent neural network based language model". In: *Eleventh Annual Conference of the International Speech Communication Association*. 2010.

[86] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[87] Christopher Olah. *Understanding LSTM Networks*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (visited on 01/13/2018).

[88] Yann LeCun et al. "Backpropagation applied to handwritten zip code recognition". In: *Neural computation* 1.4 (1989), pp. 541–551.

[89] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[90] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

[91] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. *CS231n lecture notes: Convolutional Neural Networks*. URL: http://cs231n.github.io/convolutional-networks/ (visited on 01/13/2018).

[92] Ali Sharif Razavian et al. "CNN features off-the-shelf: an astounding baseline for recognition". In: *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*. IEEE. 2014, pp. 512–519.

[93] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE. 2009, pp. 248–255.

[94] Vladimir Iglovikov and Alexey Shvets. "TernausNet: U-Net with VGG11 Encoder Pre-Trained on ImageNet for Image Segmentation". In: *arXiv preprint arXiv:1801.05746* (2018).

[95] Roger Parloff. "Why Deep learning is suddenly changing your life". In: *Fortune http://fortune. com/ai-artificial-intelligence-deep-machine-learning* (2016).

[96] Richard Socher. *CS 224D: Deep Learning for NLP1 1 Course Instructor: Richard Socher Lecture Notes: Part I*. URL: https://cs224d.stanford.edu/lecture_notes/notes1.pdf (visited on 01/13/2018).

[97] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[98] Jeffrey Pennington, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014, pp. 1532–1543.

[99] Hui Zou and Trevor Hastie. "Regularization and variable selection via the elastic net". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2 (2005), pp. 301–320.

[100] Nitish Srivastava et al. "Dropout: a simple way to prevent neural networks from overfitting." In: *Journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[101] Andrew G Howard. "Some improvements on deep convolutional neural network based image classification". In: *arXiv preprint arXiv:1312.5402* (2013).

[102] Jason Wang and Luis Perez. *The effectiveness of data augmentation in image classification using deep learning*. Tech. rep.

[103] *imgaug: Python image augmentation library*. URL: http://imgaug.readthedocs.io (visited on 01/13/2018).

[104] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. 2015, pp. 448–456.

[105] Michael Barthel et al. “Nearly eight-in-ten Reddit users get news on the site”. In: *Pew Research Center* 25 (2016).

[106] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[107] Eric Jones, Travis Oliphant, Pearu Peterson, et al. *SciPy: Open source scientific tools for Python*. [Online; accessed <today>]. 2001–. URL: http://www.scipy.org/.

[108] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.

[109] Sida Wang and Christopher D Manning. “Baselines and bigrams: Simple, good sentiment and topic classification”. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics. 2012, pp. 90–94.

[110] Jerome H Friedman. “Greedy function approximation: a gradient boosting machine”. In: *Annals of statistics* (2001), pp. 1189–1232.

[111] Tianqi Chen and Carlos Guestrin. “Xgboost: A scalable tree boosting system”. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM. 2016, pp. 785–794.

[112] Alexey Natekin and Alois Knoll. “Gradient boosting machines, a tutorial”. In: *Frontiers in neurorobotics* 7 (2013), p. 21.

[113] *Xgboost GitHub: Machine Learning Challenge Winning Solutions*. URL: https://github.com/dmlc/xgboost/tree/master/demo#machine-learning-challenge-winning-solutions (visited on 01/13/2018).

[114] Yoon Kim. “Convolutional neural networks for sentence classification”. In: *arXiv preprint arXiv:1408.5882* (2014).

[115] François Chollet et al. *Keras*. https://github.com/keras-team/keras. 2015.

[116] Martın Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[117] Julian D Olden and Donald A Jackson. “Illuminating the “black box”: a randomization approach for understanding variable contributions in artificial neural networks”. In: *Ecological modelling* 154.1-2 (2002), pp. 135–150.

[118] Xiang Zhang, Junbo Zhao, and Yann LeCun. “Character-level convolutional networks for text classification”. In: *Advances in neural information processing systems*. 2015, pp. 649–657.

[119] Baotian Hu et al. “Convolutional neural network architectures for matching natural language sentences”. In: *Advances in neural information processing systems*. 2014, pp. 2042–2050.

[120] Alexis Conneau et al. “Very deep convolutional networks for natural language processing”. In: *arXiv preprint arXiv:1606.01781* (2016).

[121] Tao Lei and Yu Zhang. “Training RNNs as Fast as CNNs”. In: *arXiv preprint arXiv:1709.02755* (2017).

[122] *Cloud-Scale Text Classification with Convolutional Neural Networks on Microsoft Azure*. URL: https://blogs.technet.microsoft.com/machinelearning/2017/02/13/cloud-scale-text-classification-with-convolutional-neural-networks-on-microsoft-azure/.

[123] Xiang Yu and Ngoc Thang Vu. “Character composition model with convolutional neural networks for dependency parsing on morphologically rich languages”. In: *arXiv preprint arXiv:1705.10814* (2017).

[124] Eszter Hargittai and Gina Walejko. “The participation divide: Content creation and sharing in the digital age”. In: *Information, Community and Society* 11.2 (2008), pp. 239–256.

[125] Inderjeet Mani and I Zhang. “kNN approach to unbalanced data distributions: a case study involving information extraction”. In: *Proceedings of workshop on learning from imbalanced datasets*. Vol. 126. 2003.

[126] Bryan Klimt and Yiming Yang. “The enron corpus: A new dataset for email classification research”. In: *European Conference on Machine Learning*. Springer. 2004, pp. 217–226.

[127] Liangjie Hong, Ovidiu Dan, and Brian D Davison. “Predicting popular messages in twitter”. In: *Proceedings of the 20th international conference companion on World wide web*. ACM. 2011, pp. 57–58.

[128] Janez Demšar. “Statistical comparisons of classifiers over multiple data sets”. In: *Journal of Machine learning research* 7.Jan (2006), pp. 1–30.

[129] Ron Kohavi et al. “A study of cross-validation and bootstrap for accuracy estimation and model selection”. In: *Ijcai*. Vol. 14. 2. Montreal, Canada. 1995, pp. 1137–1145.

[130] *ELI5: Python package which helps to debug machine learning classifiers and explain their predictions.* URL: https://github.com/TeamHG-Memex/eli5 (visited on 03/13/2018).

[131] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “Why should i trust you?: Explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2016, pp. 1135–1144.