



SCHOOL OF COMPUTATION,
INFORMATION, AND TECHNOLOGY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Data Engineering and Analytics

**Extraction and Analysis of Alcohol Craving
Situations from Discussions in
Self-Help-Groups**

Yi-Chen Andrew Liao





SCHOOL OF COMPUTATION,
INFORMATION, AND TECHNOLOGY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Data Engineering and Analytics

Extraction and Analysis of Alcohol Craving Situations from Discussions in Self-Help-Groups

Extraktion und Analyse von Alkohol-Craving-Situationen aus Diskussionen in Selbsthilfegruppen

Author:	Yi-Chen Andrew Liao
Supervisor:	Prof. Dr. Georg Groh
Advisor:	Prof. Dr. Georg Groh
Submission Date:	2022-11-15



I confirm that this master's thesis in data engineering and analytics is my own work and I have documented all sources and material used.

Munich, 2022-11-15

Yi-Chen Andrew Liao

Acknowledgments

To Anna, who has been there with me on every step of this journey;

To my family, who gave it their all to give me the best chance to grow and be successful;

To Anson, a fantastic friend and partner in mathematics, creative storytelling, and life in general;

To Robi, my fellow mastermind in programming and zoo design;

To Justin, my fellow animal enthusiast, who kept me sane by peppering me with a stream of cute animals and sports analysis;

To the saints in Munich and Budapest, who welcomed me with open arms and supported me in every way they could;

To my landlord Alexander, who swooped in at my time of need and taught me much;

To the AlcPrIntTUM members, for their rapport and being generally awesome:

To Thea and Kristina, fellow data labeling experts and classical piano enthusiasts, for our meetings and the progress we made;

To Georg, who gave me this wondrous opportunity, fostered such a collaborative work environment, and will definitely keep on rocking in whatever he does –

Without you all, this endeavor would not have been possible. This work is dedicated to you.

Abstract

Alcohol, widely regarded as one of the most dangerous and commonly abused substances in the world, is responsible for the premature death of millions of people each year. Alcohol abuse not only causes bodily impairment in the form of loss of coordination or judgement but also tops the list of risk factors for numerous life-threatening diseases and conditions. It is no small wonder, then, that many people who have experienced these symptoms would seek to break themselves free of such a predicament. However, alcohol is able to foster dependence from its users through physical and psychological factors, leading to cravings that can flood the mind of one who is trying to break free from their addiction to alcohol. Such cravings can be unpredictable and can vary from person to person.

In our work, our research group sought to uncover deeper insights regarding alcohol cravings by analyzing relevant text-based data. Through a branch of computer science called natural language processing (NLP), we endeavored to extract important entities that could be relevant to alcohol cravings from texts on the subject. We conducted a pre-study on the feasibility and effectiveness of various NLP techniques on text-based posts in alcohol self-help-groups. We also researched named entity recognition (NER) methods at length in an attempt to move forward with our entity-based analysis, and determined that a special approach would be necessary to extract custom entities such as ours.

Our first main approach used Google Maps reviews as a labeled training dataset to extract custom-domain Location entities from text. A binary classification model trained on this data returned an F1-score of 0.882, while a multi-class classification model was hampered by an undersized training set but still managed to produce results well worth discussing. We then devised and implemented a rule-based automated data labeling approach to create a training dataset for further NER approaches. Our Spark with BERT model achieved a maximum F1-score of 0.939, while our Bi-LSTM model managed a maximum test accuracy of 0.982. Further details regarding these figures will be revealed in this work.

Our models were tested on a custom in-house test set, devised and produced by members of our research group. We believe our test set is the first of its kind in the field of NER for alcohol cravings.

Contents

Acknowledgments	iii
Abstract	iv
1 Introduction	1
1.1 Alcohol and Cravings	1
1.1.1 Introduction to alcohol and its negative effects	1
1.1.2 The ongoing debate regarding alcohol cravings	4
1.2 Natural Language Processing	6
1.2.1 Introduction to Natural Language Processing	6
1.2.2 Introduction to Named Entity Recognition	8
2 Background and Related Research	10
2.1 Current Understanding and Original Goals	10
2.2 Related Research	11
2.2.1 BioNerDS	11
2.2.2 drNER	12
2.2.3 Transfer learning with Wikipedia	12
2.3 Research Question	13
3 Methodological Design	15
3.1 Our Dataset	15
3.1.1 Introduction	15
3.1.2 Statistical analysis	16
3.1.3 Accessing the data	17
3.2 Exploratory Research and Early Implementations	19
3.2.1 Word-level clustering analysis	19
3.2.2 Sentence-level clustering analysis	22
3.2.3 Analysis on current NER packages	27
3.3 Test Set Creation and Labeling	29
3.3.1 Background	29
3.3.2 Mathematical background	33
3.3.3 NER document annotation	35

4	Google Maps Reviews	37
4.1	Introduction and Background	37
4.2	Methods	39
4.2.1	Sentence Similarity Scores	39
4.2.2	Binary Classification Model	41
4.2.3	Multi-class Classification Model	42
4.3	Results and Discussion	45
4.3.1	Results	45
4.3.2	Discussion	48
5	Rule-based Labeling Approach	53
5.1	Introduction and Background	53
5.2	Methods	53
5.2.1	Data Labeling	53
5.2.2	Machine Learning Model	55
5.3	Results and Discussion	56
5.3.1	Results	56
5.3.2	Discussion	56
6	Spark with BERT	61
6.1	Introduction and Background	61
6.2	Methods	62
6.2.1	Data Considerations	62
6.2.2	Model	62
6.3	Results and Discussion	64
6.3.1	Results	64
6.3.2	Discussion	65
7	Bi-LSTM	67
7.1	Introduction and Background	67
7.2	Methods	68
7.2.1	Data format	68
7.2.2	Model	68
7.3	Results and Discussion	69
7.3.1	Results	69
7.3.2	Discussion	70
8	Conclusion	73
8.1	Outlook and Future Work	74

Contents

List of Figures	77
List of Tables	78
Bibliography	79

1 Introduction

1.1 Alcohol and Cravings

1.1.1 Introduction to alcohol and its negative effects

It may seem counterintuitive that alcohol, freely and legally available for purchase in most countries worldwide, is considered the most dangerous drug in the world [Pre22; McC19]. In fact, the consumption of alcoholic beverages, often colloquially referred to as simply "drinking," has a significant role in the social culture of many countries in the world. However, the harmful use of alcohol, often termed as *alcohol abuse* or *alcohol misuse*, is one of the leading risk factors for the health of a population [PR18]. In this section, we will lay out the numerous ways in which alcohol abuse can directly or indirectly lead to detrimental effects on the health of an individual and a population.

A global status report published by the World Health Organization (WHO) in 2018 catalogued the myriad of health consequences stemming from the overconsumption of alcohol. According to the report, approximately 3 million deaths occurred in 2016 as a direct or indirect consequence of alcohol misuse, a figure that represents over 5% of deaths worldwide that year. This percentage rises even higher when isolating to premature deaths only, with 7.2% of premature deaths (defined as deaths of persons 69 years of age and younger) attributable to alcohol. The most startling statistic concerns the deaths of younger people – alcohol was responsible for 13.5% of all deaths among people between 20 and 39 years of age in 2016 [PR18]. Nearly one in every seven deaths among young people aged 20 to 39 in 2016 was caused by alcohol. This data clearly demonstrates the destructive potential of alcohol when it is misused. Although sometimes viewed as a harmless social construct, alcohol consumption can lead to issues that have fatal consequences.

A further breakdown of the above statistics on alcohol-related deaths reveals additional insight. Table 1.1 documents the breakdown, by percentage, of the leading direct causes of alcohol-related deaths in 2016 [PR18]. The leading cause for alcohol-related deaths, as can be seen in Table 1.1, is injuries. That this is such a broad term, encompassing so many possible forms of bodily harm, serves as a testament to the potential hazards of alcohol abuse. "Injuries" here refers to, but is not limited to, several possibilities, including falls, drowning, and – perhaps the most important of them

Table 1.1: A breakdown of the leading direct causes of alcohol-related deaths in 2016.

Cause of death	% of deaths
Injuries	28.7%
Digestive diseases	21.3%
Cardiovascular diseases	19.0%
Infectious diseases	12.9%
Cancers	12.6%

all – injuries related to car accidents caused by a drunk driver [Dri22]. Staggeringly, around 30% of all traffic accident fatalities in the United States involve at least one drunk driver [Nat20a; Cen20]. This is a figure that has remained roughly consistent from 2016 to 2020 [Nat20a]. It is also crucial to recall that many of the deaths caused by alcohol-impaired drivers are to completely innocent drivers, passengers, and pedestrians who just happened to be in the wrong place at the wrong time. This, then, is an example of how alcohol abuse can have a damaging effect on not just an individual but a community as a whole.

Returning to Table 1.1, it can be seen that the next four leading direct causes of alcohol-related deaths are all related to diseases. Through this, a key fact is clearly demonstrated. In addition to its more direct ability to impair a person’s coordination and judgement and thus cause injury, alcohol can also indirectly harm a person’s body from within by increasing susceptibility to certain diseases. Chief among these are digestive diseases. It is clearly evident that the digestive system is closely linked with alcohol [PMD21]. The stomach is the first organ to have extended contact with alcohol, while the liver is the principal site of alcohol metabolism in the body [alc22]. The other diseases listed in Table 1.1 are brought about in similar manners. One important biological insight to draw here is that alcohol misuse is one of the most frequently encountered risk factors for diseases; there are a plethora of diseases for which high alcohol consumption increases one’s vulnerability to contracting them or the severity of them upon onset.

Following our review of alcohol and the depth of its negative effects, we turn our attention to the breadth, or global extent, of alcohol’s negative potential. By examining statistics on the levels and patterns of alcohol consumption, as well as the number of individuals worldwide affected with alcohol use disorder (AUD), we may gain a fuller understanding of the degree to which alcohol’s negative effects extend.

WHO data reveals the percentage of the population that consumed alcohol within the last 12 months, broken down by country. Notably, the United States of America

and Germany had 71.7% and 79.4% of their respective populations recorded [Wor]. These statistics serve to remind us how heavily the consumption of alcohol is integrated into the social culture of both countries. Table 1.2 gives a full recap of the data for the United States and Germany and a few other selected countries.

Table 1.2: Statistics on alcohol consumption for the year 2016, provided by the WHO.

	% of population	Men	Women
Alcohol consumers past 12 months: USA	71.7%	83.0%	60.7%
Germany	79.4%	88.5%	70.6%
Australia	79.4%	88.3%	70.6%
Ireland	81.3%	89.5%	73.1%
Luxembourg	91.8%	95.8%	87.7%
Heavy episodic drinking among drinkers: USA	36.4%	50.0%	18.3%
Germany	43.1%	58.7%	24.4%
Russia	60.6%	78.8%	43.7%

Another statistic we can analyze refers to the alcohol consumption per capita among drinkers aged 15 and above. While the previous statistic touches on the number of people who drink, it does not reveal anything about their drinking amounts and drinking habits. This statistic, on the other hand, can give us an idea of the amount of alcohol consumed by drinkers and whether or not it represents a serious problem. For this statistic, the US clocks in with 13.6 liters of alcohol consumed per capita, while Germany tops that with 16.7 liters of alcohol consumed per capita [Wor]. It is important to note that these measurements refer to the consumption of pure alcohol and not to the consumption of alcoholic beverages as a whole.

To gain an even more meaningful context for these numbers, we examine another statistic – the percentage of the drinking population that engaged in heavy episodic drinking over the previous period of 30 days [Wor]. This statistic allows us to isolate the drinkers who appear to have a clear alcohol issue. Table 1.2 gives the breakdown for the US and Germany by gender, demonstrating that a significant portion of both populations appear to struggle with alcohol addiction to some degree.

To confirm this, let us finally examine data on the prevalence of alcohol use disorder (AUD). According to the National Institute of Alcohol Abuse and Alcoholism (NIAAA), AUD is a medical condition characterized by an impaired ability to control alcohol use, and it encompasses the conditions often referred to as alcohol abuse, alcohol dependence, and alcohol addiction [Nat20b]. Therefore, we will henceforth use the term AUD instead of referring to the more specific and colloquial terms as we have

done before. Table 1.3 shows the percentage of the population 15 years old and above that are classified as having AUD, broken down by country [Wor]. Again, we choose to focus on the US and Germany and a few other countries.

Table 1.3: 12 month prevalence for AUD for the year 2016, provided by the WHO.

	% of population	Men	Women
USA	13.9%	17.6%	10.4%
Germany	6.8%	9.8%	4.0%
Hungary	21.2%	36.9%	7.2%
Russia	20.9%	36.9%	7.4%

That these numbers are so high is a global concern. Even the 6.8% for Germany, the lowest overall figure in the table, represents more than 1 out of every 15 people and constitutes a set of millions of people. When the other higher figures are considered, one can truly begin to understand the massive scale to which AUD is an issue.

That over one in every three men in Hungary and Russia is considered to have AUD beckons the question – how and why is alcohol so addictive? Other more biology- and psychology-heavy papers will cover this topic at more length, but the general idea is that alcohol is able to physically alter the brain’s chemistry and functions [Car]. There are other factors that will become relevant in our further discussion, which we divide into physical factors and psychological factors. Physically, alcohol causes the brain to release dopamine and endorphins, which produce feelings of pleasure and also act as a painkiller. Psychologically, a person may become dependent on alcohol either as a reward or as a coping mechanism [Car].

1.1.2 The ongoing debate regarding alcohol cravings

Having seen the widespread effect and destructive potential of alcohol as well as its ability to manipulate users into alcohol dependence, we now move on to the next section. As mentioned previously, alcohol is able to foster dependence and addiction from its users by way of physical and psychological factors. From a physical standpoint, an alcohol user can become addicted to the feelings of pleasure produced by the consumption of alcohol and the subsequent release of dopamine and endorphins. This is a physical addiction to alcohol. If the psychological angle is considered, it can be seen that an alcohol user can become addicted to the reward of alcohol. Similarly, an alcohol user going through a difficult time may psychologically depend on alcohol as a coping mechanism or as a means of social inclusion. Both the physical and psychological aspects of alcohol addiction can flood the alcohol user’s mind with thoughts, desires,

or cravings for alcohol, ranging from periodic to incessant. Such thoughts, hereafter referred to as *cravings*, may be seen as a sign of alcohol addiction. Unfortunately, there are some scientific disagreements about how cravings should be defined [Haz20]. Thus, there is no official definition of alcohol-related cravings. We will here examine the methods by which the Diagnostic and Statistical Manual of Mental Disorders (DSM-V) and the International Classification of Diseases (ICD) define AUD and the associated cravings.

Published in 2013, the DSM-V is the fifth edition of a manual that serves as the standard classification of mental disorders used by mental health professionals in the US. It features the most recent updates in the field based on the scientific contributions of over 200 subject matter experts [Ame22]. The DSM-V definition of AUD is given in the form of an eleven-step diagnostic criteria evaluation. The overall definition states that AUD is a "problematic pattern of alcohol use leading to clinically significant impairment or distress, as manifested by at least two of the following [diagnostic criteria], occurring within a 12-month period" [13]. Intriguingly, one of the eleven diagnostic criteria regards cravings: "4. Craving, or a strong desire or urge to use alcohol" [13].

The ICD is a globally-used diagnostic tool that provides knowledge on the causes and consequences of various human diseases, allowing it to be used globally as a diagnostic tool. Diagnoses for diseases and other conditions are stored in the form of alphanumerical codes in a systematic, hierarchical manner. For example, the code F10 refers to alcohol-related disorders. Underneath this code, F10.1 refers to alcohol abuse and F10.2 refers to alcohol dependence. A further breakdown can be found on [22a]. While no explicit mention of alcohol cravings can be found throughout the conditions listed under alcohol-related disorders, it is worth mentioning how the definitions for alcohol-related disorders in the ICD are largely binary – a patient either suffers from the condition or not. In contrast, the definition in the DSM-V can be seen as a spectrum, ranging from light to moderate to severe. According to experts in the field, defining AUD on a spectrum is much preferred due to the flexibility it affords during diagnosis and treatment. The ICD definition, however, is the definition used in the insurance field for reimbursements and other activities so it does make sense for them to emphasize simplicity over all else.

There exist numerous other definitions for cravings which vary and differ in subtle ways. The NIAAA defines cravings as a "broad range of thoughts, physical sensations, or emotions that tempt you to drink, even though you have at least some desire not to" [Ret]. This definition brings two critical points to the forefront. One is that cravings can be either physical or psychological, and the other is the fact that cravings refer specifically to urges to drink only when the affected person is trying, in some capacity, to limit his or her drinking.

Alcohol cravings are highly subjective; they may come in differing or even opposite forms for different people [Haz20]. Many sources believe cravings may be intense but are generally short-lived, often lasting less than five minutes in duration [Ray21; Ret]. But this may vary based on a person's experience – other people experience less-intense but nagging cravings that push them for hours into the night. Cravings can also be triggered by a cue. These cravings find their source in an external stimulus that triggers latent memories in the brain [Haz20]. Internal triggers typically involve memories or emotions that prompt a desire to drink, such as sadness, anxiety, or physical pain. External triggers refer to environmental cues linked with alcohol, such as locations, the time of the day, or people [Ray21]. From a physical standpoint, some AUD patients suffer from withdrawal-induced cravings. Withdrawal-induced cravings make the brain demand more alcohol in order to maintain homeostasis [Haz20].

Cravings are a natural symptom of addiction and they manifest themselves in a variety of ways – each recovering AUD patient's path to sobriety is a unique story [Haz20]. Ultimately, the most important action for AUD patients to consider when it comes to alcoholic cravings is to understand their cravings and where they come from, and decide how to address them [Ray21]. In our research, we hope to leverage our results to shed more light on the nature of cravings.

1.2 Natural Language Processing

1.2.1 Introduction to Natural Language Processing

In computer science, the type of data a programmer encounters may vary greatly. When the data is in the form of human language in speech or text form, then this is a specific branch of computer science called *natural language processing* (NLP). NLP can be seen as the intersection of computer science, artificial intelligence (AI), and linguistics, as one of the primary goals of the field is to program a computer or AI to have the ability to process and analyze large amounts of natural language data [IBM20; Wik22b]. NLP employs different strategies and techniques from different fields to create models that can process human language and understand its full meaning. These models include rule-based models, statistical models, machine learning models, and deep learning models [IBM20]. These models face the challenge of learning to understand spoken or written language in terms of its meaning, its linguistic nuances, and the sentiment behind it.

NLP has many uses and applications in the industry. For example, NLP provides an aid in translation tasks where a program has to translate text from one language to another. NLP can help a computer-based system to respond to human voice commands, both in the form of completing the task required and replying in human language. This

has its uses in voice-operated navigation systems, customer service chatbots, and other customer-oriented services [IBM20].

In order to be able to accomplish these high-level tasks, lower-level tasks are necessary. Here, we will outline a few such tasks which are particularly vital, before moving on to a deeper dive on Named Entity Recognition (NER).

Part-of-speech tagging

Part-of-speech tagging (POS tagging) is a NLP technique that focuses on categorizing words within a corpus by their part of speech. Discerning between different forms of the same word is one challenge faced by algorithms attempting to solve this task. Such algorithms must be able to use the context in which the word appears to determine which form of the word is present, and thus which part of speech it belongs to. This task is called word sense disambiguation.

Word sense disambiguation

As mentioned above, *word sense disambiguation* is the NLP subtask concerned with determining which word sense of a word is correct given the context it appears in [EA08]. Selecting the correct word sense is referred to as disambiguation. Word sense disambiguation is useful for other tasks such as POS tagging and machine translation.

Co-reference resolution

Co-reference resolution is the NLP task of finding all expressions in a given text that refer to the same real-world entity [MM22; The]. For example, co-reference resolution attempts to find the antecedent of each pronoun in the given text. Figure 1.1 demonstrates a successful application of co-reference resolution.

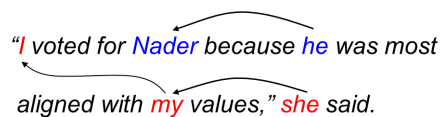


Figure 1.1: An example of a successful co-reference resolution. The antecedent for "he" in the example sentence is "Nader," while the three pronouns "I," "my," and "she" actually refer to the same person [The].

Sentiment analysis

Sentiment analysis seeks to identify the emotion or opinion revealed by text. Sentiment analysis algorithms seek to analyze text and determine its sentiment or intent. A popular example involves classifying user reviews on a certain topic (e.g. a movie or a restaurant) into those with a positive sentiment and those with a negative sentiment. More challenging examples, though, deal with obstacles such as sarcasm and idioms, which mask the true meaning of the text.

Natural language generation

Natural language generation is a task involving the programming of artificial intelligence to leverage computer-accessible data in order to generate written or spoken texts [Rei96]. In addition to being used for tasks such as writing of documents or the production of reports in an automated manner, natural language generation also finds its uses in human-to-machine or machine-to-human interactions [Wig21]. The primary goal of natural language generation is to produce written or spoken language that is understandable and fits the context of the conversation or situation.

1.2.2 Introduction to Named Entity Recognition

Named Entity Recognition (NER) is a NLP subtask that aims to extract specific words and phrases, known as entities, and classify them into predefined label classes or semantic types [EKK17; Fan+21; MS20]. Also known as entity chunking, entity extraction, or entity identification, NER's core challenge is to identify and categorize key information in text-based data. Entities extracted and classified by NER can fall under two main categories: standard entities and domain-specific entities. Standard entities refers to the generic entity types present in most NER systems, such as people (the names of recognizable people), locations (countries, cities, landmarks, etc.), organizations (companies, universities, etc.), and times (years, months, days of the week, etc.) [Mar19]. Most other entity classes fall under domain-specific entities, entities that are specific to a certain field or a certain task. While there are many NER packages that have already been trained to perform NER on more generic entities, performing NER on domain- or task-specific entities is a much greater challenge due to the relative novelty of the problem and the need to learn afresh. The difficulty increases substantially when there is a lack of labeled data with which to train the NER system.

Several ways of approaching the NER problem exist, each coming with its advantages and shortcomings [EKK17]. For example, some rule-based methods use regular expressions and pre-defined dictionaries to locate entities of interest. Other machine learning approaches use large corpora as training data and learn to extract and classify

entities. Literature concerning terminologically-driven NER methods that map concepts to terminology also exists [EKK17].

A successful NER algorithm can be applied to numerous use cases. A NER algorithm working in conjunction with a topic identification algorithm can help to optimize search queries and to rank search results properly [Kan+16; Gup18b]. NER is also utilized in machine translation, assisting the translation algorithm in determining which named entities need to be translated and which ones do not [Kan+16]. NER is also helpful in customer support, as a NER algorithm can extract custom entities from queries for assistance (e.g. which product is causing the problem, which part of the shipping process is causing the problem) and redirect the customer to the correct customer representative that works in that particular department or on that particular issue [Gup18b]. As an example of an application of domain-specific NER, the biomedical field uses NER on biomedical data to extract certain genes, drugs, or diseases specific to their field [Kan+16].

2 Background and Related Research

2.1 Current Understanding and Original Goals

In this section, we discuss the background of our project, including a larger view of the overarching goal and various aspects of the overall project. The origins and motivations of the project will be discussed. Following this, the original goals and intentions of the project will be outlined. Then, after reviewing some research and literature that employ methods relevant to our project, we will define the research question.

The overall project was in part the brainchild of Nathalie Stuben, an internationally-recognized author, podcaster, and social media presence [Mov22]. Her work, inspired by her own journey to defeat AUD, focuses on gently guiding participants towards a life of stable abstinence from alcohol [Stu]. This project was born from a collaboration between her and the Social Computing research group at the Technical University of Munich (TUM), headed up by Prof. Dr. Georg Groh [Tec22]. Dubbed AlcPrIntTUM, the project centers on alcohol prevention and intervention. Figure 2.1 shows a visualization of the various lines of work within this project.

To briefly touch upon the other components of this project, we see that another major portion of this project involves the collection and analysis of physiological data. Team members working on this portion of the project created a mobile application to be used with a smart watch device that would prompt users to log when they experienced cravings. The application would also collect snapshots of the user’s physiological data, such as their heart rate and blood pressure, in a bid to detect a relationship between users’ physiological data and self-reported cravings. Under the NLP-related group, there are two other major tasks. One involves the analysis of text-based posts to gather data regarding cravings in order to create a simulation model for modeling the pattern of cravings. The other is a pre-study that seeks to determine the feasibility of different state-of-the-art language models used to analyze text-based data. The overall long-term vision of the project is to create an algorithm-based early-warning system, most likely in the form of a mobile application, that supports users when they experience cravings.

Moving on to the portion of the project covered in this thesis, the original objective was to extract events from text-based data in order to determine which entities had an influence on alcoholic cravings. Doing so would allow us to integrate this knowledge into the intervention stage of this project, where we would be able to send warnings

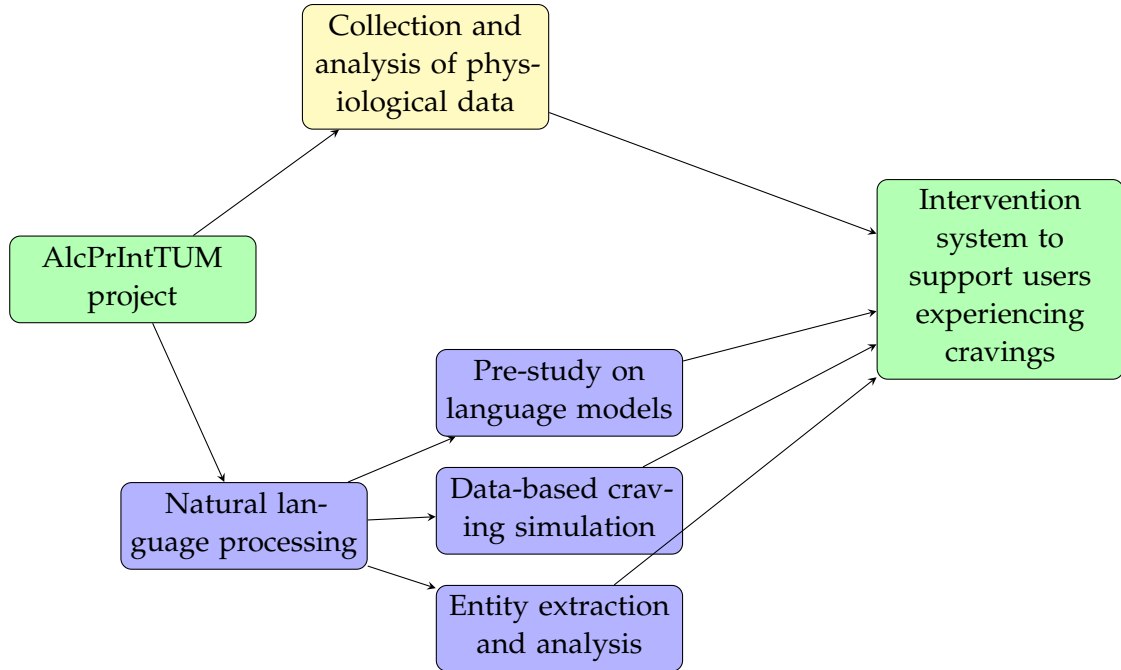


Figure 2.1: A rough breakdown of the projects and tasks within AlcPrIntTUM

and advice to the user when certain events are detected.

2.2 Related Research

It quickly became evident that the area of research we were dipping into was not one that has been extensively fleshed out. With our initial goal being to extract entities that bear a relationship to the author’s alcoholic cravings from a text post, many NER packages fell short in their abilities to detect such fine-grained entities on their own. In addition, many approaches to do so require large amounts of labeled training data, which we lacked. In this section, we examine a few approaches that either boast similar goals or make use of strategies that are relevant to our task.

2.2.1 BioNerDS

BioNerDS is a cleverly-named NER system that performs NER to extract mentions of the names of bioinformatics-related databases and software from scientific journals [Duc+13]. It is interesting and relevant to our line of work because it is a specialized NER system that extracts a specific type of entity that is generally not included in

standard NER packages. BioNerDS accomplishes this task by utilizing a pre-defined dictionary combined with a rule-based scoring system, which is documented in the paper [fig16]. The scoring system, for which we will give a few examples here, is highly customized to the domain and the anticipated form of the desired entities. The highest positive influence to a score can be derived from an entity that directly matches with an entry in the pre-defined dictionary, while other positive score influences come from entities that are part of a Hearst pattern, are followed by a version number, or are in all-caps. On the other hand, the score is deducted for entities that are an English dictionary word or are known bio-acronyms. After the collection of entities which finish the rule-based evaluation with a score above their specified threshold, a second pass is done to analyze these entities.

2.2.2 drNER

Another specialized NER system we studied was called drNER. Its purpose is to extract the names and quantities of dietary nutrients (e.g. 25 grams of unsaturated fat) [EKK17]. One particularly intriguing aspect of drNER was how the system attempts to take the context of the sentence where the entity was extracted from into account. In this way, the system could contextualize whether the amount of the extracted nutrient is too high, too low, or just right, according to the recommended intake amounts. This is relevant to our project because we also stand to gain from being able to understand the context of the sentence in which the entity was detected. If such a technique is possible, we may be able to detect whether the mentioned entity in our sentence is a positive influence or a negative influence on the speaker's alcohol habits.

2.2.3 Transfer learning with Wikipedia

A surprising source of labeled data is the well-known online encyclopedia, Wikipedia. Wikipedia hierarchically classifies each of its articles, starting from specific categories all the way to very broad categories, including "People" and "Places" [Wik21]. By observing the highest-level class that a Wikipedia article is categorized into, one can deduce that the subject of that Wikipedia article is an entity of the type denoted by the highest-level category of the article. Kim et al. used these labels to perform a two-step adaptive NER using distant supervision [Kim+21]. The first step, referred to as distant supervision, involved the use of category information from Wikipedia to build up weakly labeled data. In the process of doing so, named entity pairs were collected when two entities appeared in the same sentence together. In the second step, the manually-labeled CoNLL 2003 dataset was used to fine tune the model. Kim et al. reported strong results using this technique to perform NER on contemporary texts. They were able to get a

model to perform well in a domain that differed from the domain on which the model was trained.

2.3 Research Question

Given social media utterances of people with substance dependency problems, we would like to extract a more in-depth knowledge model to find out more about the context of cravings. For example, what are the typical locations in which people, or a certain user specifically, experience cravings? What about the time of day? Is there any correlation between certain social contexts and experiencing cravings? Deducing causal relations in some cases would be extremely helpful, but is expected to be more difficult.

We focus on external events and situations because we expect that people generally try to pin their cravings to something external within their reach. In order to do this, we define an event as a (location, time, purpose) tuple. Here, it is immediately apparent that we will need to extract more specific information than what is given in most traditional NER subfields. For example, many NER systems are able to detect location entities, but these location entities are often countries or other large geopolitical entities. Knowing which country a person is may be slightly helpful for our purposes in rare cases, such as if the country has alcohol prohibition laws or if the country happens to be hosting a major alcohol-centered festival. However, it is safe to say that we would generally be able to gather a lot more information from a more fine-grained location, such as a person's location within town and his or her proximity to sources of alcohol. We would also benefit from locations that are contextualized and generalized. The location entities we would analyze for this project are broken down in Section 3.3.1.

In a similar vein, the person entities we extract must also be fine-grained enough for them to be useful for our purposes. Many NER systems are able to recognize the names of famous people and extract the entity as a person; however, we are more concerned with the person's relationship to the author of the text post. Does the presence or mention of a particular person that the author is familiar with induce an alcoholic craving? Are the author's cravings brought about by seeing strangers drink? We want to analyze the psychological source of cravings and gather data regarding it, so we want the person entities we extract to be able to be generalized for all people suffering from AUD, to some extent. The person entities we analyzed for this project are broken down in Section 3.3.1.

Another consideration that had to be made for our project is the limited quantity of data available. Our data would always be limited to some extent. We also do not have the means to annotate the data due to its size, the inefficiency of human labeling, and the time span of the project. Thus, we expected to have to use semi-supervised or

unsupervised or transfer learning techniques, significantly increasing the difficulty of our research question.

Given all the aforementioned challenges, determining the scope of this project became a challenge in itself. Whereas we initially had hoped that we would be able to move on from entity recognition to perform deeper analysis on the influences of external factors on the cravings of patients suffering from AUD, it soon became evident that the NER portion of the project would consume the vast majority of the time and resources of our project. Our project, then, was an exploratory analysis (dubbed a "pre-study") that would focus on the feasibility of different NLP and NER methods on what we could gather as our data. As such, the overarching objective of this project was not necessarily to achieve groundbreaking results but rather to evaluate a plethora of techniques and their effectiveness in our problem scenario.

3 Methodological Design

This chapter will recount the design and implementation decisions behind our research and experimentation prior to the implementation of the main methods. We begin by introducing the dataset that we ended up working with, then move on to outline the tasks accomplished and document the insights gathered during this process. Finally, we introduce the test set that we worked with and report what we did as a group here. The creation of our test sets was a vital and time-intensive task that cannot be overlooked.

3.1 Our Dataset

3.1.1 Introduction

We needed a text-based dataset through which we could draw meaningful conclusions on patterns of alcohol cravings. In the early stages of our project, numerous data sources were considered for analysis. A few aspects regarding these datasets were seen to be of paramount importance when we were in the process of selecting datasets. Firstly, we wanted the dataset to be in English, due to it being the only common language that all our team members speak. Secondly, the dataset needed to be freely and legally accessible.

One option emerged as the most feasible way forward: a forum on the American website Reddit, a website for content aggregation and discussion. Reddit is arranged as a collection of smaller communities, ranging in size from a single-digit number of subscribers to tens of millions of subscribers. Referred to as subreddits, each of these communities focuses on a specific topic, including world news, finance-themed discussion, sports, and cute animals. The forum we were interested in is one called `/r/stopdrinking` (subreddits are often prefixed with an `/r/` because that is the form of their URL) that acts as a support group for those who wish to stop or control their drinking. This will henceforth be referred to as the *Reddit dataset*.

We chose the Reddit dataset for many reasons. It represents our target population very well, since all users there want to stop drinking to some extent and are on various stages of their respective recoveries. It is freely and legally accessible, with data scraping via an API as a possibility, and has a search function. Another perk of the Reddit

dataset is that it is completely anonymous. This presents a twofold benefit: users are more willing to pour out their honest thoughts when hidden by the cloak of anonymity, leading to what we would hope to be more informative and more accurate posts. Secondly, we are much less likely to run into any ethical issues regarding user privacy.

Regarding the dataset itself, the format of the majority of the posts works to our favor as well. Many posts are verbose, honest, and colloquial. Many posts span multiple paragraphs, allowing us to get a good idea of the background and context of the subjects the author brings up. Most posts also read in a natural way, with what we see as an appropriate level of colloquialisms and slang.

3.1.2 Statistical analysis

There are over 380,000 (and growing) subscribers to `/r/stopdrinking`. While not all subscribers to the subreddit post regularly, not all posts are written by subscribers. On average, between 150 and 200 posts are made to `/r/stopdrinking` each day, and this number is rising year by year. In the five-year span that our dataset covers, we can see that around 50,000 posts were created in 2017, while around 70,000 posts were created in 2021. A significant spike in the number of posts occurred starting in May 2021, perhaps as a direct or indirect result of the worsening of the ongoing COVID-19 pandemic [Kra22].

The subreddit utilizes BadgeBot, an automated bot-based system which flairs users with the number of days elapsed since their last alcoholic drink [22b]. This figure represents a method through which a member's journey to sobriety may be tracked. Members with a small number of days on their so-called flair are not necessarily in the early stages of recovery, however; they may in fact be far along in their journey but recovering from a recent relapse. According to data analysis on our dataset, the vast majority of posts are made by users with a single-digit number on their user flair [Kra22]. This is in stark contrast to the overall breakdown of user flairs on `/r/stopdrinking`, as seen in Figure 3.1.

Conceptually, this makes sense and can be explained by the fact that users that have not consumed alcohol for longer periods of time would be less concerned with their alcohol consumption and thus less likely to post on a forum that is entirely centered around alcohol consumption. In addition, these statistics may indicate that a good proportion of users experience success in their journey towards sobriety and abstinence, though such a statement is merely a hypothesis based on raw statistics and cannot be taken as the truth without further investigation.

In relation to our project, however, the statistic demonstrating that most posts are made by users with very low numbers on their BadgeBot flairs is another point to underscore the relevance and usefulness of the Reddit dataset to our study. In our study,

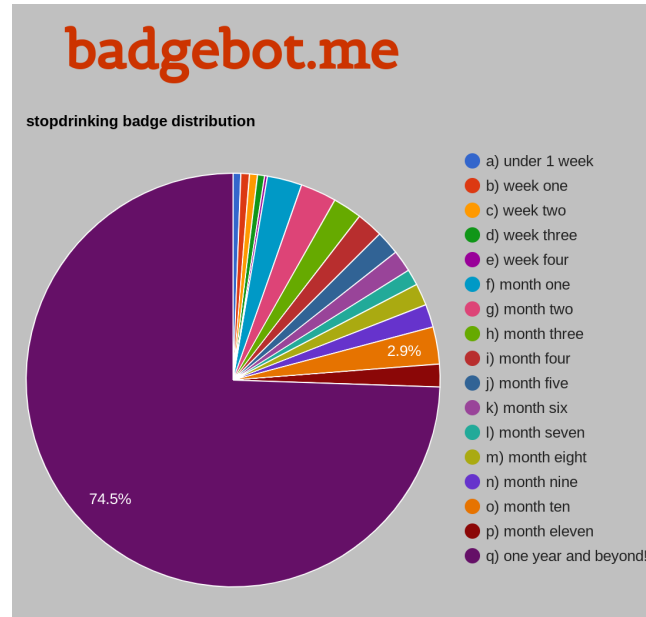


Figure 3.1: Statistics provided by BadgeBot with regards to how many days flaired users have been sober for [22b].

we are not as concerned about AUD patients who are so far along in their recovery so as to neglect a public forum on recovering from AUD; rather, we are interested in those that are still experiencing cravings and may still benefit from regular care and guidance.

3.1.3 Accessing the data

We needed a systematic and efficient way to access the data made available to us through /r/stopdrinking. To do this, the Reddit API was used [Bri20; Red22]. In order to scrape the data, several steps had to be followed in order to accomplish several tasks.

Getting access

The Reddit API allows for users to create a personal use script associated with a Reddit account. We can then request a temporary authentication token, called an OAuth, from Reddit.

Scraping the data

We make a request to grab posts from a Reddit link, and store the result as a JSON file. Since the JSON file is similar to a Python dictionary, we can loop through the posts in a manner similar to how we would access a Python dictionary. We store the relevant data in a Pandas dataframe. Below, in Table 3.1, a rundown is given of the data fields that we pulled.

Table 3.1: A breakdown of the data fields scraped from /r/stopdrinking using the Reddit API.

Field name	Description	Access
subreddit	The subreddit from which the data was pulled	post['data']['subreddit']
title	The title of the post as written by the author	post['data']['title']
selftext	The body of the post as written by the author	post['data']['selftext']
upvote_ratio	The ratio of upvotes to downvotes	post['data']['upvote_ratio']
score	The number of upvotes minus the number of downvotes	post['data']['score']

Filtering the data

In order to filter the dataset to the point where it contained the most relevant posts to our projects, we utilized a two-pronged approach. We commenced by retrieving five years of data, from April 2017 to April 2022 (the date the data scraping was performed). The first filter was by post length. Since we are only interested in posts that contain enough substance for us to analyze, we filtered out short posts which are less than 60 characters in length [Kra22]. This was to put the focus on longer posts that contain more content to analyze and to remove short posts consisting of greetings and short words of encouragement, common on /r/stopdrinking. The second filter was by topic. Our priority in analyzing the dataset was to center on informative posts about cravings, including their context. To achieve this, we implemented a Python RegEx-based filter to selectively retrieve posts that contain any form of the word "craving" or any of its closest synonyms. Some example words we used were "craving," "desire," "urge," "trigger," and "relapse" [Kra22]. We also took the additional step to filter out posts that

contained one of our above keywords, but negated. In all, our resulting dataset totaled about 40,000 posts.

The dataset pulled in this manner ended up forming the basis for our test set, which we expound upon further in Section 3.3.1.

3.2 Exploratory Research and Early Implementations

This section documents the insights gained and the progress made in the early experimental stages of our project. These are techniques and methods that were implemented and carried out in the early stages as a means of gaining more knowledge about our data and more experience with applying NLP to real world examples. Although these methods may be somewhat primitive, they yielded some insightful results and should not be excluded from this report.

3.2.1 Word-level clustering analysis

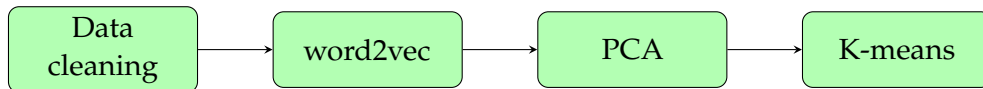


Figure 3.2: Outline of data processing pipeline for word-level clustering.

Figure 3.2 gives a visualization for the data pipeline followed during our initial stages of feasibility research on word clustering. Other works follow a similar process when performing analysis on other corpora, such as world news headlines [Kor21]. Such an approach would surely be useful to us if we are dealing with shorter, concise social media posts. The approach may differ slightly when we are dealing with lengthier forum posts, however.

Data cleaning would involve stripping filler words out of the posts and ensuring that the posts are uniformly formatted in a way that we would be able to work with. Then, a word embedding technique such as word2vec could be used to turn natural language into mathematical vectors. word2vec employs a neural network model to learn word associations given a large corpus of text. Following this, principal component analysis (PCA) could be performed on the word vectors to reduce the dimensions of the data while preserving as much information as possible. Stripping the data down to fewer dimensions not only simplifies the mathematical calculations that have to be done but also makes the data easier to visualize for humans. Then finally, the K-means algorithm would be performed to cluster the words based on their word embeddings. The results of such a clustering are shown below in Figures 3.3 and 3.4.

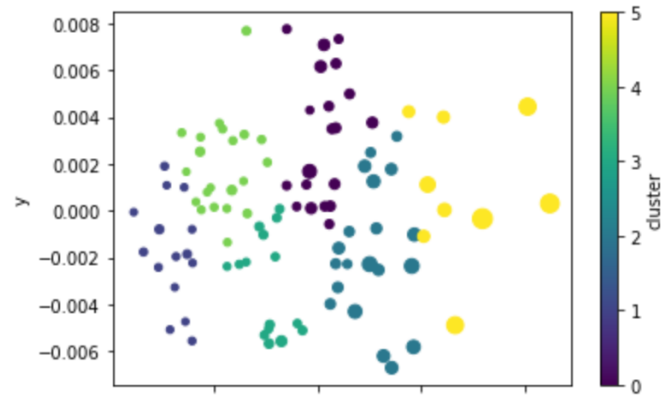


Figure 3.3: Visualization of the top 100 words by frequency from the Reddit dataset, quantified using the word2vec model, reduced to two dimensions using PCA, and clustered into a pre-determined number of clusters using the K-means algorithm. The size of each dot is proportional to the frequency of the word it represents.

```

CLUSTER 4 WORDS:
5      week
6      months
8      find
11     see
21     without
28     went
29     part
33     love
35     great
49     give
56     mind
58     help
64     lot
65     year
67     say
79     anyone
83     thought
88     quit
90     family
96     anxiety
97     take
Name: word, dtype: object

```

Figure 3.4: A partial capture of the results of K-means clustering on top 100 word vectors. The number on the left represents the frequency ranking of the word.

As can be seen in Figure 3.4, the clusterings we were able to obtain were not terribly insightful. We had hoped that the clustering algorithm could reveal some intrinsic classes of words, such as words that are related to relapse, words that are related to cravings, and words that are related to successful abstinence. If this had been the case, we would immediately have been able to gain a better understanding of which entities are positive, neutral, or negative influences on people suffering from AUD who are trying to stop drinking. But unfortunately, this was not the case and it was difficult to deduce the differences between words in different clusters from a logical standpoint.

Another proposed method was to run NER with a pre-trained model and then perform situation detection by using relation inference, a collection of techniques that would give us an idea of the relationships between different words. Here, we highlight a few such techniques that were seen as promising for our data. Unfortunately, as we will see later in Section 3.2.3, this was not immediately feasible due to the poor performance of standard NER libraries on our dataset.

Co-occurrence

Words that occur together in the text more frequently have a stronger relationship.

Rule-based

Relationships are generated through rule-based approaches, including grammar, part-of-speech tagging, and syntactic analysis.

Machine learning and deep learning

A machine learning model is trained to determine the strength of the relationships between different words. The use of a support vector machine (SVM) is popular here due to their ability to perform classification. In the deep learning approach, the word vectors would be passed into a deeper neural network.

Graph-based

The text is converted into a graph, with entities as nodes (vertices) and relationships as edges. There are then many graph theory-based techniques that could be utilized on the graph representing the text.

3.2.2 Sentence-level clustering analysis

The next level of analysis performed was sentence-level clustering. The idea behind this was to have a larger window of context to consider. A word by itself can have multiple meanings and connotations, and because of that we need a larger scope so we can contextualize the words. Going off of previous similar works, we also wish to see if we can find patterns between sentences [Kor21]. For example, it would be very insightful to our project's purposes if we were able to discover clusterings of craving sentences versus non-craving sentences, or maybe sentences spoken from a certain location.

To begin, we clustered post titles. We did this because post titles generally had a length of around one sentence, and because post titles would, ideally, serve as an effective summary of the content of the body of the post. Clustering on individual text body sentences was planned if this approach yielded useful results for our work. We performed sentence clustering using three different types of sentence embedding methods: spaCy, SentenceBERT, and Universal Sentence Encoder. The results from one run of sentence clustering are shown in Figure 3.5.

```
CLUSTER 0 SENTENCES:
The Daily Check-In for Tuesday, May 3rd: Just for today, I am NOT drinking!
Guys, I woke up great today for the first time on a Monday morning in almost 10 years!
I'm thinking about my alcohol journey and wow. I drank 109 times in 2021. Take 1 hangover day goes to 218
The Daily Check-In for Monday, May 2nd: Just for today, I am NOT drinking!

CLUSTER 1 SENTENCES:
As of today (5/03/22) I have been clean from hard liquor/spirits for almost 4 months now
I starting using a colouring sheet to monitor my drinking habits and now I am 6 months sober
I've been a high functioning alcoholic for about 6 years now, and I'm terrified to have my family find out.
I have been sober for 3 days, but I doubt my ability to keep it going.
Been a heavy drinker for 2 years. What are some really positive things you noticed after stopping?

CLUSTER 2 SENTENCES:
Had to separate a otherwise happy marriage for mental health reasons. I'm just on the emotional floor.
I haven't had this much fun at a party in a long time

CLUSTER 3 SENTENCES:
Day 3
Day 1 again
Day two
Day 1.
Day 7

CLUSTER 4 SENTENCES:
I want to stop drinking
I need to stop binge drinking
```

Figure 3.5: A screen capture of the results of clustering on 100 post titles from the Reddit dataset. spaCy was used to generate sentence embeddings while DBSCAN was used to cluster the sentences.

spaCy sentence embeddings

We used spaCy to create sentence embeddings, which allowed us to cluster the sentences as sentence vectors instead of natural language sentences. We then used DBSCAN to cluster the sentence vectors. DBSCAN stands for Density-Based Spatial Clustering of Applications with Noise, and it is a clustering algorithm that does not take in an argument representing a pre-determined number of clusters from the user. Rather, it takes in an epsilon value ϵ from the user to determine how fine-grained the clusters should be. The epsilon value represents the maximum distance between two data points for one to be considered as being in the neighborhood of the other [sci22b]. A higher epsilon value tends to create fewer clusters, as the distance between data points will easily fall under the threshold and cause the two samples to be considered as being in the same neighborhood, increasing their likelihood of ending up in the same cluster. On the other hand, a lower epsilon value tends to create more clusters, as many data points will not be considered as being in the same neighborhood and thus will end up in their own tiny clusters.

To reach a clustering with an ideal number of clusters, the same clustering algorithm was run in a loop on the same data with differing epsilon values. Based on the number of clusters in the resulting clusterings, certain epsilon values were trialed in order to determine which clustering made the most logical sense. The clustering in Figure 3.5 was one such chosen clustering. As can be seen, the sentences in Figure 3.5 that belong to the same cluster do bear relationships to one another. A cursory examination of the clusters allows us to see that Cluster 3 is composed of posts titled with the day number (referring to the number of days since their last drink) while Cluster 4 has two sentences with similar meaning: both post titles strongly express that the author would like to stop drinking.

However, a closer examination of other clusters reveals some faults with our clustering. For example, the sentences in Cluster 0 seem not to be united by any trait other than the fact that they all contain references to periods of time such as years, months, and days. Cluster 1 is similar, containing sentences that have a cardinal number followed by a unit of time (e.g. 6 months). Real issues start to arise when we investigate Cluster 2. We can see that the two sentences may be linked due to the fact that both of them contain positive emotion words, but closer examination reveals that the sentiments of the two sentences are actually the exact opposite.

The number of sentences seen in Figure 3.5 is nowhere near 100. The reason for this is that the DBSCAN algorithm excludes a sentence from being in any cluster if the epsilon threshold is not met. Conceptually, this means that sentences that are not closely-related enough to any other sentence will not be grouped into a neighborhood. Such sentences will not be in any cluster and thus will not show up in the clustering.

SentenceBERT embeddings

SentenceBERT is a modification of the BERT network that uses siamese and triplet networks in order to determine semantically meaningful sentence embeddings [RG19]. The BERT network is explained in more detail in Section 6.1. Since one of the main goals of BERT is to generate a language model, BERT can be tuned to be used on a wide variety of NLP-related tasks [Dee21]. SentenceBERT is one such application of BERT that leverages the pre-training of BERT to accomplish state-of-the-art results in the field. We used SentenceBERT as the sentence embedding method in the same context as we used spaCy embeddings above – clustering 100 post titles from the Reddit dataset with DBSCAN. Figure 3.6 shows the results of our sentence-level clustering using SentenceBERT embedding.

As was the case with our clustering using spaCy embeddings, not all 100 sentences show up in the clustering due to the epsilon value chosen. However, we can see that the sentences that did end up in the clustering are well-clustered for the most part. In particular, Cluster 2 in Figure 3.6 is filled with post titles laden with positive sentiment, while Cluster 6's post titles express challenges for the post author. Cluster 4 contains two post titles that both celebrate a milestone, and this result is noteworthy because the two posts share no common words besides "no."

Universal Sentence Encoder embeddings

Another sentence embedding system we trialed was Universal Sentence Encoder. The Universal Sentence Encoder, published on TensorFlow by Google, encodes sentences into high-dimensional vectors that can then be used to work on NLP tasks [Gooa; Cer+18]. Universal Sentence Encoder is trained and optimized for tasks dealing with text longer than just one word such as sentences and paragraphs, making it a suitable candidate for our task [Gooa]. The underlying framework of the model is trained by a deep averaging network encoder, also known as a DAN. In a DAN, input embeddings for words and bi-grams are averaged together; following that, they are passed into a feedforward deep neural network (DNN). This results in a 512-dimensional sentence embedding. DAN's have a computation advantage over other encoders due to the fact that the compute time is linear on the length of the input sequence. As was the case in the previous two sections, we used DBSCAN to cluster 100 post titles from the Reddit dataset after using Universal Sentence Encoder to generate sentence embeddings. The results can be seen below in Figure 3.7.

Upon human examination, the clusters generated seem to be generally quite accurate. Clusters 1, 2, and 3 in Figure 3.7 have very centralized meanings: post titles in Cluster 1 seek help, post titles in Cluster 2 celebrate sobriety streaks, while post titles in Cluster 3


```
CLUSTER 0 SENTENCES:  
happy sober Sunday.  
Happy Sunday SD  
  
CLUSTER 1 SENTENCES:  
day 1  
Day 2  
1 Year  
I'm back, day 1  
Hey guys! 1 year!  
Day 11 complete.  
Day 1  
  
CLUSTER 2 SENTENCES:  
lots of support 😊  
good news  
An incredible feeling  
The best gift  
  
CLUSTER 3 SENTENCES:  
Breakthrough.  
Hello  
  
CLUSTER 4 SENTENCES:  
4 months no alcohol &lt;3  
5 years no drinking  
  
CLUSTER 5 SENTENCES:  
Made it through a wedding!  
Wedding Party  
  
CLUSTER 6 SENTENCES:  
having a hard time  
Today was a tough one
```

Figure 3.6: A screen capture of the results of clustering on 100 post titles from the Reddit dataset. SentenceBERT was used to generate sentence embeddings while DBSCAN was used to cluster the sentences.

```
CLUSTER 0 SENTENCES:
30days
day one - here we go again!
day 3
Day 10
Day 13 complete.
Day 1 for me again
52 days
Day 50.
Day One
Day 1
Day 2
30 days.

CLUSTER 1 SENTENCES:
Any advice?
Need Tips

CLUSTER 2 SENTENCES:
Almost five days sober
101 days sober

CLUSTER 3 SENTENCES:
I hate drinking
Sleep problems makes me want to stop drinking
I'm drinking too much. I don't know how to stop.
```

Figure 3.7: A screen capture of the results of clustering on 100 post titles from the Reddit dataset. Universal Sentence Encoder was used to generate sentence embeddings while DBSCAN was used to cluster the sentences.

express strong negative emotions towards the author's drinking problem. While these results are insightful, it is unfortunate that, besides generic day-counting posts, only seven post titles were successfully clustered. The clustering shown in Figure 3.7 was already the result of hyperparameter tuning to find the ideal hyperparameter values to return a clustering that yields the most information.

3.2.3 Analysis on current NER packages

In this section, we go over some of the top NER packages available online or as Python packages. While this analysis yielded almost no useful insight to our project, it still composed a significant chunk of our research and should not be ignored. However, due to its inconclusive nature, the analysis here will be quite brief.

spaCy

spaCy, an open-source software library specializing in advanced natural language processing, offers an entity recognition package that allows us to download an English language analysis pipeline and then perform NER on text using one of its components. The models we used, `en_core_web_sm` and `en_core_web_lg`, are the small and large versions, respectively, of English language models trained on written text from the web [spa]. While the NER package performed admirably on text laden with countries and the names of famous organizations and people, it was unable to recognize even a single entity from sentences from our dataset.

GATE with ANNIE

GATE, which stands for General Architecture for Text Engineering, boasts an online NER system powered by an information extraction system known as ANNIE (A Nearly-New Information Extraction system) [GAT]. ANNIE was developed with the goal of extracting person, location, and organization entities from American news articles in mind, but it was unable to extract any of our more specific person and location entities.

NLTK

NLTK, which stands for Natural Language Toolkit, is a package that can be imported via Python. It is proficient at extracting generic entities but failed in our task.

Stanford NER

The Stanford Natural Language Processing group provides three flavors of Java-based NER systems free and available for use [The20]. Each of the three different versions works on a different number of entity classes. The 3-class version can tag three entity types: Locations, Persons, and Organizations. The 4-class version can tag Miscellaneous entities in addition to the three entities covered by the 3-class version. The 7-class version can tag Money, Percent, Date, and Time entities in addition to the three entities covered by the 3-class version. Each of the models was trained on a different training dataset. Unfortunately, none of the three were able to produce any satisfactory results on our dataset.

Polyglot

Polyglot NER was the next system tried [AIR+15]. Polyglot, as its name indicates, works on multiple languages. However, we were only interested in its English NER capabilities for our project, and unfortunately it was unable to sufficiently extract entities from our dataset. One interesting thing about Polyglot is that it learns by creating its own datasets from Wikipedia. The topics of many Wikipedia articles are labeled in a hierarchical manner. If one of the higher-level labels is taken, we can use Wikipedia as a labeled dataset of persons, locations, and organizations [Wik21]. This is a similar approach to that found in [Kim+21], discussed in Section 2.2.

Flair

Flair is another state-of-the-art NLP system that boasts NER and text embedding capabilities [Git]. The most advanced NER model provided by Flair can tag 18 English entity classes, including Person, Location, and Geopolitical Entity [Hugb]. Unfortunately, Person entities refer to person names, so this model would also have a difficult time extracting the fine-grained Person entities we used for our project described later in Section 3.3.1. Something similar would be true for our fine-grained Location entities. When applied to our dataset, Flair was able to extract several entities with relatively high confidence, but none of them were useful for the purposes of our project. Most of the entities extracted by Flair were DATE entities (e.g. 'last week,' 'one month') or ORDINAL entities (e.g. 'the first time,' 'the second time').

BERT Base NER

The bert-base-ner model on the HuggingFace database is a state-of-the-art NER model that has been fine-tuned from a BERT model [Huga]. It can recognize four entity

types: Persons, Locations, Organizations, and Miscellaneous entities. The model was fine-tuned on the CoNLL-2003 NER dataset, which is a well-known NER dataset that consists of Reuters news stories from between August 1996 and August 1997 [Pap]. Later, in Chapter 6, we will work to create datasets that are structured in a manner similar to the CoNLL-2003 NER dataset. Regrettably, `bert-base-ner` was ultimately unable to successfully extract even a single entity from our dataset.

3.3 Test Set Creation and Labeling

3.3.1 Background

In earlier sections, we saw the importance of using a training set with a sufficiently large number of posts to train a model on. In addition to this, though, we also needed a test set in order to accurately evaluate the performance of any models trained on the training data. We have the Reddit dataset, from where it is possible to pull thousands of posts relevant to our topic of study. Specifically, we also have the filtered dataset from Section 3.1.3 which is a subset of the Reddit dataset that contains posts that were selected by our rule-based filter. However, none of this data is labeled – it is impossible to tell which of these posts in the filtered cravings dataset are actually posts that could be considered as cravings posts. Moreover, many of these posts mention persons and locations and sentiments but we have no way of evaluating a model trained to extract these entities unless we have a test set available. Since our task deals with entities from a specific domain, no suitable labeled test sets were freely available for us to use. Thus, we had to create our own.

As a base, we started with a subset of the filtered cravings posts from the Reddit dataset. After much discussion, it was agreed upon that we would label these posts in four areas: Craving, Persons, Locations, and Emotions.

Cravings

This was a simple boolean field for whether the post in question is a craving post or not. We defined a craving post as a post that mentions any reason at all to drink, or any reason to relapse, for a person who is trying to stop drinking. This could be for the author of the post or someone else mentioned in the post. Of course, any posts with clear mentions of cravings related to alcohol would count as well. On the other hand, a post could not be a craving post if the people in question had not yet discovered or accepted their alcohol problem.

Persons

We structured the Persons, Locations, and Emotions sections as a set of boolean labels for a carefully selected set of entity names. Each entity could be marked as true or false, meaning that the presence of multiple labels per post was allowed. Our motivation behind the chosen set of possible persons was to cover a wide span of different people that one could come into contact with on either a regular, semi-regular, or non-regular basis. In Table 3.2, our chosen Person labels and short descriptions regarding them are listed.

Table 3.2: A list of the Person entities used in our projects with their respective descriptions, along with the percentage of posts in our test set labeled with each person entity.

Entity name	Description	Percentage
alone	A specific mention of the person being alone or by himself/herself	12.29%
one friend	A mention of a singular friend, regardless of the context	6.86%
multiple friends	A mention of more than one friend at a time, regardless of the context	16.29%
family	A mention of any member of immediate or extended family	24.00%
partner	A boyfriend, girlfriend, husband, or wife	22.00%
colleagues	A mention of one or more people who are associated through work, including a boss	4.86%
strangers	A mention of one or more persons with whom there are no known connections	8.29%

Locations

The logic behind the Locations section is nearly identical to the logic behind the Persons section. For our locations, we tried to consider the places and venues where alcohol may be consumed (e.g. bars and restaurants), as well as places that people naturally spend a lot of time at (e.g. home and work). We also focused on places where alcohol may be visually seen. In addition, as we pored over the Reddit dataset manually as a means of familiarizing ourselves with the general characteristics of the dataset, we noticed a non-insignificant amount of posts that mentioned the airport as a trigger for

drinking, and vacation as a cause for relapse. For that reason, we added these two locations into our list of locations as well. Our full list of locations is documented in Table 3.3.

Table 3.3: A list of the location entities used in our projects with their respective descriptions / labeling rules, along with the percentage of posts in our test set labeled with each location entity.

Entity name	Description	Percentage
home	A specific mention of someone’s own home or house	26.57%
school/university	Any school or university or place of education	4.00%
work	Any mention of a job or workplace	23.14%
restaurant	A place where people pay to sit and eat meals	4.00%
bar	A place where alcohol and refreshments are served, especially over a counter	6.00%
pub	A place that serves both food and alcohol in a sit-down environment	0.57%
party	A specific mention of a social gathering or celebration involving invited guests	10.00%
physical activity	Any mention of exercise that requires energy	11.71%
supermarket	Any large shop that sells consumable goods and more	6.57%
outdoors	An explicit mention of being outdoors, particularly in a park or similar environment	6.86%
liquor store	A store that specializes in selling alcohol	3.43%
airport	A specific mention of the airport	0.86%
vacation	Any mention of traveling or being on vacation	6.57%
other	Any other location which is not defined above	6.29%

Emotions

The structure is the same as for Persons and Locations. This section was actually not relevant to the project described in this paper, but it was relevant to other parts of the overall AlcPrIntTUM project. The reason for this is because we wanted to investigate which emotions are associated with drinking. Each person suffering from AUD is different, so people drink for broadly different reasons. Once again, we wanted to

capture a wide spectrum of emotions, and our chosen emotions are documented in Table 3.4.

Table 3.4: A list of the emotion labels used in our projects with their respective descriptions, along with the percentage of posts in our test set labeled with each emotion.

Emotion	Description	Percentage
anxious/afraid	Feelings of anxiety or fear. Examples: anxiety about work, fear of the unknown	34.86%
sad	Feelings of sadness or depression. Examples: struggling with depression, crying	23.71%
stressed	Examples: feeling swamped at work, feeling overwhelmed by family responsibility	11.43%
tired	Examples: feeling exhausted after work, feeling sleepy	9.43%
frustrated	Examples: frustration about situation, frustration about drinking	22.57%
happy	Positive feelings of joy. Examples: drinking to celebrate, feeling happy about not drinking	23.43%
angry	Examples: drinking as a result of a fight, being an angry person	6.29%
proud	Examples: prideful feelings that one can control his/her drinking, feelings of accomplishment from reaching a milestone	18.29%
bored	Examples: drinking because there is nothing else to do, having a boring lifestyle	6.00%

Labeling rules

In the process of labeling the test set data, those who undertook the task of manually labeling the data followed a set of rules, which we determined together as a team. Some of these rules were drafted prior to the commencement of our labeling, while others were added in as we ran into special cases during our labeling.

For Cravings, any post that involved cravings, relapse, or wishes to drink were marked as true, given that the affected person is trying to stop drinking. This is in accordance to one of the unofficial definitions of cravings that we went over in Section

1.1.2. The affected person does not necessarily have to be the author of the post. In contrast, posts containing mentions of cravings, relapse, or urges are marked false if the affected person is not aware of his or her drinking problem yet.

For Persons, a type of person entity is marked true if the specific name or synonym of the person or group of persons is mentioned. This includes possessives; for example, a post containing a mention of "my father's influence" would warrant a label of `Person:family` for the "father." On the other hand, the names of famous people are omitted in our labeling. In addition, we clarify that `Person:alone` should not be automatically marked as true just because no other Person entities are mentioned.

The labeling rules for Locations were quite similar to that of Persons. A type of location entity is marked true if the specific name or synonym of the place is mentioned. `Location:other` is only used when a location not included in our predefined locations list is mentioned. However, location entities in posts where that particular location entity is not mentioned directly and can only be inferred (e.g. "I fell asleep at my desk" could refer to a desk at home, a desk at work, or a desk at school) are not marked as true unless there is a mention of it elsewhere.

For Emotions, any direct mention of an emotion being experienced by any person in the post results in that emotion being marked as true. Synonyms of emotions are mapped to the closest emotion label available (e.g. "depressed" is mapped to `Emotion:sad`).

We created a expansive spreadsheet to coordinate and keep track of our labeling in a semi-synchronous manner. Each person tasked with data labeling was given his or her own sheet on the spreadsheet where boolean values could be input for each field through a simple click of a checkbox. A further sheet consolidated the results of our labeling, while other sheets kept track of statistics and other matters.

3.3.2 Mathematical background

With regards to the creation and manual labeling of our test set, one matter became of utmost importance – the size of the test set. Determining a good test set size had a twofold importance for our project. For one, we wanted to establish a lower bound for an effective test set size. It was crucial to ensure that the number of samples we could feasibly label as a team would actually end up being a sample size that would be helpful for us to accurately draw conclusions from our experimentation. If the minimum number of samples required was impractically high, then this would be an endeavor that we should not spend the time and effort to pursue manually. On the other hand, we also wanted to establish an upper bound for our test set size. Since manually labeling test set samples was a time-consuming task, we desired to ensure that we would not spend unnecessary amounts of time for minimal gains once we

exceeded such an upper bound.

To proceed, we had to consider the nature of our data. As mentioned above in Section 3.3.1, our test set was to be labeled with four sets of boolean information fields, named Craving, Persons, Locations, and Emotions. Besides the Craving field, each of these sets of fields allows for multiple labels – for example, each post in the dataset may have more than one location labeled. In total, there are 31 boolean fields under these four main categories. Fortunately, they operate independently of one another. While there may be some correlation between two labels, there should not be any causation or direct relationship. Thus, we can represent the problem of our test set labeling as a Bernoulli experiment with independent Bernoulli random variables [Kra22].

Based on this, we desired a metric that would give us an idea of how representative our test data set was of the overall dataset. We wanted to be able to quantitatively assess how true to the overall dataset our results based on our test set were, based on the number of test set samples. Intuitively, we expected that with a smaller test set, the variance would be higher and it would be possible that there would be a significant disparity between results stemming from evaluation on our test set and the true results. On the other hand, it was our expectation that evaluating a model on a larger test set would yield results that we would have greater confidence in. If such a confidence metric could be determined, then we would be able to better determine the desired size for our test set.

Hoeffding’s inequality is an inequality in probability theory that provides an upper bound on the probability that the sum of bounded independent random variables such as ours differs from its expected value by more than a given amount [Wik22a]. This concept was used and applied to the calculation of a desired test dataset size. While these calculations are relevant to our project, they did not fall within the scope of our project and so the details are omitted here. The results, though, are presented in [Kra22] and in Figure 3.8 below.

Interpreting the table is not straightforward. Each row represents the size of the test set. Each column represents the amount by which the test set accuracy differs from the whole dataset accuracy. So for any pairing (N, ϵ) , the number found in the corresponding cell represents an upper limit on the probability that the accuracy when evaluated on a test set of size N differs from the accuracy when evaluated on the entire dataset by more than ϵ . For example, taking a pairing of $(N = 300, \epsilon = 0.075)$, we see that the probability that the accuracy from evaluation on a test set of size 300 and the accuracy from evaluation on the entire dataset differ by more than 0.075 is going to be less than or equal to 0.07, the value in the corresponding cell. $N = 300$, then, became a goal for us to shoot for in our test set labeling – the value of 0 for $(N = 300, \epsilon = 0.1)$ was appealing to us and represented a solid first step in our projects. In effect, evaluation on a test set of size 300 guarantees that our evaluation accuracy would differ from the

$\epsilon \backslash N$	0.012,5	0.025	0.037,5	0.05	0.062,5	0.075	0.087,5	0.1	0.112,5	0.125
25	1.98	1.94	1.86	1.76	1.65	1.51	1.36	1.21	1.06	0.92
100	1.94	1.76	1.51	1.21	0.92	0.65	0.43	0.27	0.16	0.09
200	1.88	1.56	1.14	0.74	0.42	0.21	0.09	0.04	0.01	0
300	1.82	1.37	0.86	0.45	0.19	0.07	0.02	0	0	0
500	1.71	1.07	0.49	0.16	0.04	0.01	0	0	0	0
1000	1.46	0.57	0.12	0.01	0	0	0	0	0	0
2000	1.07	0.16	0.01	0	0	0	0	0	0	0
5000	0.42	0	0	0	0	0	0	0	0	0
10000	0.088	0	0	0	0	0	0	0	0	0

Figure 3.8: Table showing the upper bound on probabilities that the accuracy from evaluation on the test set and the accuracy from evaluation on the overall dataset differ by more than ϵ , given test set size N [Kra22].

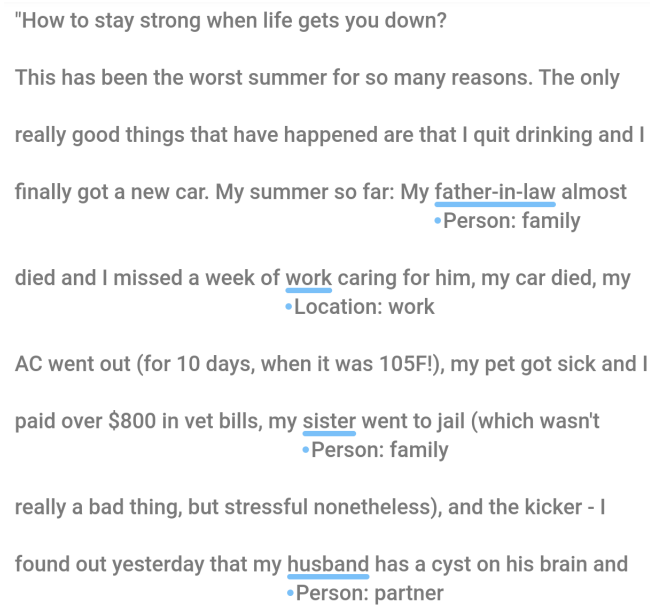
actual accuracy by no more than 0.1. Our Reddit test set ended up being a dataset of size 350.

3.3.3 NER document annotation

As documented in Section 1.2.2, an NER model seeks to learn how to extract entities of certain types from text. In order to evaluate such a model, the test set would need to have labels not only for the text post in general but specifically for each entity as well. The test set labeling process detailed in Section 3.3.1 will provide a label that tells us that the post contains a `Person:friend` entity, for example, but will not specify where in the text that entity occurs. A NER model would need to learn from and be evaluated on data where individual entities are specified and labeled.

For this, we needed to create a new test set that would contain gold standard labeled entities. We wanted a semi-automated method to assist us in labeling individual entities. An open-source document annotation tool named Doccano was chosen as the means to create our NER test set. Doccano was chosen for its user-friendly interface and the features it possesses to help us create labels, label posts, and convert our labeling into a computer-readable file. Below is an image illustrating the workflow of Doccano.

As reiterated in previous sections of this paper, one of the goals for our project was to not spend a significant amount of time on tasks involving manual labeling. As such, the number of test set samples created in this manner was very limited.



"How to stay strong when life gets you down?"

This has been the worst summer for so many reasons. The only really good things that have happened are that I quit drinking and I finally got a new car. My summer so far: My father-in-law almost
•Person: family

died and I missed a week of work caring for him, my car died, my
•Location: work

AC went out (for 10 days, when it was 105F!), my pet got sick and I paid over \$800 in vet bills, my sister went to jail (which wasn't
•Person: family

really a bad thing, but stressful nonetheless), and the kicker - I found out yesterday that my husband has a cyst on his brain and
•Person: partner

Figure 3.9: An open-source document annotation tool named Doccano was used to create our NER test set. This photo shows the user interface and labeling capabilities of Doccano.

4 Google Maps Reviews

4.1 Introduction and Background

One of the main methods used in our project involved Google Maps reviews alongside the Google Cloud Platform (GCP). In this section, we discuss the motivation behind this method, as well as the procedure of the methods that were carried out. While we were ultimately rendered unable to proceed with research and development down this particular route, it was at one point seen as an avenue that would provide the bulk of the content for this project. As such, this discussion will reflect the sizeable efforts put into this method.

Our lack of labeled training data is an issue that has been covered in previous sections. Here, it served as motivation for us to find another similar dataset which we could use to assist the training process for a machine learning model. The reviews posted by users around the world on Google Maps were chosen as such a dataset due to a few reasons, which we will reveal here. For one, the dataset would be very large due to copious amounts of people who use Google Maps and the numerous locations around the world for which reviews can be posted. That the data would most likely read in a manner somewhat similar to the Reddit dataset due to its colloquial, user-written nature was another consideration. Finally, another very important reason for us was that the majority of locations on Google Maps are categorized into classes [Goo22]. With large amounts of labeled text data available, we were able to move forward utilizing Google Maps reviews as a labeled training dataset.

We used the Google Maps API in order to gain access to Google Maps reviews and scrape the data [Goob]. After registering for an authentication token, we were able to run Google Maps searches through the API URL and receive output in JSON format. Below, Figure 4.1 shows a snapshot of some of the fields of information received.

Since the output is in JSON format, we can treat it in Python as a sort of nested dictionary object. There is one dictionary entry per establishment found by our search query, and then the dictionary entry of each establishment is laid out with various information fields, as seen in Figure 4.1. Of these, the most crucial and relevant pieces to our project are the latitude and longitude coordinates of the location as well as the `place_id` field. The latitude and longitude coordinates are what allow us to find the corresponding establishment during our initial search, as we will see later. The

```
{
  "html_attributions" : [],
  "results" : [
    {
      "business_status" : "OPERATIONAL",
      "geometry" : {
        "location" : {
          "lat" : 29.773869,
          "lng" : -95.39785859999999
        },
        "viewport" : {
          "northeast" : {
            "lat" : 29.77521927989272,
            "lng" : -95.39649002010727
          },
          "southwest" : {
            "lat" : 29.77251962010727,
            "lng" : -95.39918967989271
          }
        }
      },
      "icon" : "https://maps.gstatic.com/mapfiles/place_api/icons/v1/png_71/restaurant-71.png",
      "icon_background_color" : "#FF9E67",
      "icon_mask_base_uri" : "https://maps.gstatic.com/mapfiles/place_api/icons/v2/restaurant_pinlet",
      "name" : "1751 Sea and Bar",
      "opening_hours" : {
        "open_now" : false
      },
      "photos" : [
        {
          "height" : 3024,
          "html_attributions" : [
            "\u003ca href=\"https://maps.google.com/maps/contrib/104228578639082988074\" \u003eDavid H"
          ],
          "photo_reference" : "AeJbb3c2IyRf2Dx_yG_ZY9qXqAre1A1R6ICBD8DGC8oxDMPLAC9TxJmW5KtVh0AJ-xcRLaG",
          "width" : 4032
        }
      ],
      "place_id" : "ChIJAzV3-2THQIYRhiygh5E2D_A",
      "plus_code" : {
        "compound_code" : "0JF2+GR Washington Avenue Coalition / Memorial Park, Houston, TX",

```

Figure 4.1: A screen capture of the output received from Google Maps API, in JSON format. Of particular note are the latitude and longitude coordinates of the location as well as the `place_id` field.

`place_id` field is a unique identifier for that particular establishment that we then use to extract the reviews for it.

Using the `place_id`, a Place Details request can be initiated to obtain more details about the corresponding establishment. A Place Details request can return a comprehensive list of information fields, including reviews, on the queried location. Up to five reviews are returned for each location (and no more than that, because we used the free tier of GCP resources). Each review comes as an array of information fields, including the author of the review, the rating given (in stars), the language of the post, the text of the post, and more [Goob]. Of these, only the text of the post is relevant to our current project.

In order to build our Google Maps reviews training set, several steps had to be taken. A very useful Python package named `GeonamesCache` provided us with a list of latitude and longitude coordinates of major cities in the United States [Gom22]. This allowed us to iterate through US cities, searching for locations of a certain type within a specified radius of the city. Iterating through cities in the US was ideal for the purposes of our project due to the fact that Google Maps is based in the US and the further fact that Google Maps reviews on locations in the US would almost exclusively be in English [Gal16]. In this manner, we were able to build up a sizeable training set of Google Maps reviews, the details of which we will elaborate further upon in Sections 4.2.2 and 4.2.3.

4.2 Methods

4.2.1 Sentence Similarity Scores

As an initial proof-of-concept idea, we took sentences from Google Maps reviews and compared them to sentences in our test set. The logic behind this idea was to take a sentence that was clearly about a certain topic, most commonly a certain location, and see which sentences in our test set are the most similar to it. In doing so, we would be able to infer which sentences from our test set were posted from a similar location to our original sentence from the Google Maps review. Below, we see the results of one such test run. The sentences in Figure 4.2 are post titles from the Reddit set, and the similarity scores displayed are in relation to a sentence praising the food and service at a restaurant. A similarity score closer to 1.0 indicates that the two sentences are more similar, while a similarity close to 0.0 or even below 0.0 indicates that the two sentences share very little in common.

This strategy yielded some results worth discussing. If we examine a few of the sentences that have higher similarity scores, we see that most of them have something to do with food. A sentence that mentions "drinking alcohol eating lots" earned the

```
Sentence = need bit support today work event ; similarity = 0.097244196
Sentence = days sober rant need advice ; similarity = 0.2261197
Sentence = seven months drain last night ; similarity = 0.057998504
Sentence = went curry last night ended pints ; similarity = 0.3288942
Sentence = day calmness ; similarity = 0.17597495
Sentence = tried drinking alcohol eating lots get urges get drunk ; similarity = 0.4374025
Sentence = weddings tough ; similarity = 0.06660948
Sentence = friday night party ; similarity = 0.0941461
Sentence = want sober ; similarity = 0.08468655
Sentence = almost threw towel ; similarity = 0.12899214
Sentence = blew week sober ; similarity = 0.10288742
Sentence = need plan ; similarity = -0.013819057
Sentence = wanted say thanks folks great ; similarity = 0.28240117
Sentence = years since stopped drinking years ago almost killed ; similarity = 0.1614945
Sentence = juicy breath shallow sigh ; similarity = 0.3012544
Sentence = absolutely cannot sleep ; similarity = -0.060975753
Sentence = yesterday gf years broke drinking ; similarity = 0.12554485
Sentence = first dinner ; similarity = 0.35937387
Sentence = many things sober ; similarity = 0.10078791
Sentence = daily check friday may th today drinking ; similarity = 0.16568342
```

Figure 4.2: A screen capture of the results of sentence similarity analysis. Listed sentences are post titles belonging to the Reddit dataset, having been pre-processed to remove stop words. The sentence from which the sentence similarity scores are derived is a generic positive Google Maps review sentence on a restaurant.

highest similarity score among the set of sentences here at 0.437. Another post title simply stating "first dinner" earned a 0.359 similarity, while another post title that mentions "curry," a food item, scored a 0.329. Incidentally, another sentence that expressed gratitude and happiness ("wanted say thanks folks great") scored a relatively high 0.282, most likely due to the similarly positive and grateful tone of the sample sentence. Structurally, none of these sentences were similar to the sample sentence, so these sentence similarity scores were probably mostly based on the factors mentioned above rather than the sentence structure.

For this technique to have returned very insightful results, we probably would have had to extend the domain to entire paragraphs. Two ideas are possible here – we could either find a vector embedding for each paragraph and run a similar algorithm to perform paragraph similarity, or we could iterate through a paragraph in a sentence-by-sentence manner, collecting sentence similarity scores along the way. The former idea would involve comparing paragraphs using paragraph similarity scores and could be useful to identify posts that are written about certain locations. The latter idea could result in a more fine-grained analysis where each paragraph is considered as the sum or average of its sentences. Each sentence could be expressed as a similarity vector, where each element in the similarity vector refers to the similarity score between the sentence and a certain sentence on a certain entity, or a certain sentence on alcohol cravings.

4.2.2 Binary Classification Model

We built a binary classification model as another initial proof-of-concept. Once again, a restaurant or bar was taken as the target location for the initial experimentation. The goal was to train a model to learn to distinguish between posts related to a restaurant or bar and posts not related to a restaurant or bar. In order to accomplish this, the first step was to train the model to distinguish between Google Maps reviews that are on restaurants and bars versus Google Maps reviews that are on other locations.

Our dataset for this problem was built with the assistance of GeonamesCache, mentioned above [Gom22]. We split the dataset into positive samples (Google Maps reviews written on a restaurant or bar) and negative samples (Google Maps reviews written on a location other than a restaurant or bar). Negative samples were drawn from a sampling of Google Maps location types, including airport, bank, bookstore, casino, church, gym, hospital, library, pharmacy, school, and university [Goo22]. GeonamesCache was used to obtain a list of latitude and longitude coordinates of US cities around which to base our searches, and then the dataset was populated through search queries that specified the location and the location type. A snapshot of our dataset is given below in Figure 4.3. Here, target refers to the label for each data entry – we can see that the fourth entry, labeled 3252, is the sole entry among these five which represents the review of a bar or a restaurant. The text field gives us the actual text of the review, while data refers to the multi-dimensional vector embedding of the text of the review, generated by spaCy.

	target	data	text
24279	0.0	[0.0064834356, 0.19516923, -0.16754441, -0.141...	Yoga District is the perfect yoga studio. They...
19937	0.0	[0.029777, 0.14626977, -0.14517179, -0.0581043...	Same thing change your add to the proper time ...
10669	0.0	[-0.06720874, 0.15611579, -0.1836434, -0.05919...	Brittney just answered the phone and in 1 minu...
3252	1.0	[0.044040564, 0.22163357, -0.10923502, -0.1246...	Wonderful food, friendly staff, and excellent ...
19385	0.0	[-0.25417343, 0.14122644, -0.05976057, -0.0960...	Welcoming

Figure 4.3: The dataframe of our dataset for Google Maps reviews binary classification. target is the label for each data point: 1 if the location is a restaurant or bar, and 0 otherwise. data is the vector embedding for the review. text is the actual text of the review.

We could play a role in controlling how large the dataset would be by setting the number of loop iterations. We used GeonamesCache to prepare a list of the latitude and longitude coordinates of major US cities. The number of cities returned in this manner was 3264. When running the search query with the Google Maps API, we set

the radius at 1500 meters. The query would then return all establishments within 1500 meters of the specified latitude and longitude that fit the criteria. The limit of reviews returned per establishment using a free-tier Place Details request is 5. This meant that the maximum amount of data we could prepare in this manner would be somewhere in the ballpark of 3264 multiplied by however many establishments there are within a 1500 meter radius, multiplied by the number of location types we used, multiplied by 5. In one of our most successful runs, we used 200 cities to provide the dataset with positive examples (bars and restaurants). We then used 100 cities and 11 different location types to provide the dataset with negative reviews. The resulting number of samples in our dataset was 9537 positive reviews and 22,399 negative reviews. With the dataset prepared, it was then split into 75% training data and 25% test data. We standardized the data and fed it into a logistic regression model.

The model we used was a simple logistic regression model [Kar22]. This was a fitting model for the task because it could take in a multi-dimensional input in the form of our sentence embeddings and return a binary output. On this task, our binary classification model performed admirably and returned an F1 score of nearly 0.92. The full results will be detailed and discussed in Section 4.3.

Since the model was trained entirely on Google Maps review data, we wished to confirm that the abilities it learned would carry over to other domains. In Figure 4.4, we see that the model was able to correctly discern between two sentences that reference different locations but are otherwise identical. The sentence where the author is eating food was marked positive (restaurant or bar), while the sentence where the author was reading books was marked negative (not a restaurant or bar). We did not proceed much farther with this line of verification due to the fact that this binary classification model was considered just a stepping stone of sorts. In addition, the problem that the binary classification model was trying to solve is easier than the actual problem we want to solve – in the end, we will have to distinguish between restaurants and bars as well, instead of lumping them together into one category. As such, we did not extensively test this particular binary classification model on our test set created from the Reddit dataset. However, since the accuracy statistics were quite good, this motivated us to proceed with a multi-class classification model.

4.2.3 Multi-class Classification Model

Seeing as our trained model was able to perform excellently on its original data domain and performed admirably on test data, we decided to implement a multi-class classification model and evaluate its performance. Conceptually, a multi-class classification model can be thought of as the combination of many binary classification models, each corresponding to one of the multiple classes. When the multi-class

```
[ ] text1 = "Yesterday I was happily eating bread and drinking soup with my friend when I just felt like I really wanted to order a massive beer."
    text2 = "Yesterday I was happily reading books and grading papers with my friend when I just felt like I really wanted to order a massive beer."

    model.predict( standardizer.transform( [nlp( text1 ).vector, nlp( text2 ).vector] ) )

array([1., 0.] )
```

Figure 4.4: Sample output for a test case. In this example, the two sentences passed in to the model are referencing different locations but are otherwise identical. The model is successfully able to make the distinction.

classification model is choosing which class to assign a data point into, each of the many binary classification models returns a value which can be interpreted as the confidence the model has that the data point belongs in that corresponding class. In the case of a multi-class classification model, the model would then pick the class with the highest confidence and assign the data point to that class.

Once again, we began by using GeonamesCache to get a list of latitude and longitude coordinates of major US cities. This time, however, we used a different list of location types. Our list of location types included ten location entity types from Table 3.3 in Section 3.3.1, plus six more location types that did not fall under the category of any of the previously-mentioned location entities. The ten Google Maps location classes used for our multi-class classification model and their corresponding counterparts from our test set entity types are listed in Table 4.1.

Table 4.1: A list of the Google Maps location classes used for our multi-class classification model plus their corresponding counterparts from our test set entity types from Section 3.3.1.

Google Maps location class	Test set location entity
school	school/university
university	school/university
restaurant	restaurant
bar	bar
night_club	party
gym	physical activity
store	supermarket
supermarket	supermarket
liquor_store	liquor store
park	outdoors

A snapshot of the dataset generated in this manner is given below in Figure 4.5. Here, target refers to the label for each data entry – in our case, 2 corresponds to a restaurant, 5 corresponds to a gym, and 6 corresponds to a supermarket. As was the case with our binary classification model, the text field gives us the actual text of the review, while data refers to the multi-dimensional vector embedding of the text of the review, generated by spaCy.

	target	data	text
4327	6.0	[-1.132999, -1.9314116, -2.2189999, -0.2832391...	Great church, with meaningful community involv...
4296	6.0	[-0.4395538, 0.24582574, -3.4923997, -1.398534...	I remember when my mom shopped at what now is ...
706	2.0	[-1.2055342, 0.19449653, -2.3641975, 0.2077623...	No seasoning on the chicken or fried tomatoes ...
4452	6.0	[-0.6088774, 1.3552842, -2.3689303, -1.4209839...	Went to an event here during the World Games, ...
3184	5.0	[-0.70005107, 2.9135802, -3.6584392, -0.838792...	There was one sales guy there and he was eager...

Figure 4.5: The dataframe of our dataset for Google Maps reviews multi-class classification. target is the label for each data point: 2 represents a restaurant, 5 represents a gym, and 6 represents a supermarket. data is the vector embedding for the review. text is the actual text of the review.

We used an SVC support vector machine provided by scikit-learn to serve as the backbone of our model [sci22c; sci22a; Ped+11]. A support vector machine (SVM) seeks to find a linear barrier that maximally separates the data points of two classes, making it useful for classification problems. In a similar manner to how we considered a multi-class classifier to be a collection of binary classifiers, SVM can perform multi-class classification by breaking down the multi-class classification problem into multiple binary classification problems [bae21].

For our problem, we had to deal with imbalanced data. When running our Google Maps review query, we could only control how many cities to loop through – we could not control how many locations of each type would be returned from each city. Due to this, the number of samples for each class were somewhat imbalanced. For example, in one run where we used 20 cities per loop, 1674 establishments were returned for the class restaurant but only 143 were returned for the class night_club. To remedy this, we used a Python package named SMOTE [The22]. SMOTE, which stands for *Synthetic Minority Over-sampling TEchnique*, performs over-sampling to balance out the labeled classes. Following this, we were able to train our SVC model on our multi-class training data and evaluate it on our test data of Google Maps reviews. While the results

were not terribly impressive, our multi-class classification model was still able to return somewhat insightful results on our example sentences, as demonstrated in Figure 4.6. A sentence referring to food was classified into the restaurant class, which is arguably the correct classification.

```
[ ] text1 = "Yesterday I was happily eating bread and drinking soup with my friend when I just felt like I really wanted to order a massive beer."
    text2 = "Yesterday I was happily reading books and grading papers with my friend when I just felt like I really wanted to order a massive beer."

    model.predict( standardizer.transform( [nlp( text1 ).vector, nlp( text2 ).vector] ) )

    array([2., 5.] )
```

Figure 4.6: Sample output for a test case. In this example, the two sentences passed in to the model are referencing different locations but are otherwise identical. The model is successfully able to classify the first sentence into the label class 2, which represents the restaurant class.

The next step was to set up the evaluation pipeline for our multi-class classification model on the Reddit test set. We pulled our manually-labeled test data as a CSV file and converted it to a Python pandas dataframe, a preview of which is shown in Figure 4.7. This allowed us to retrieve the results of our manual labeling on the Reddit test set. We then deleted all samples in the Reddit test set where no location entities were tagged, as these samples would be impossible for our model to classify correctly and would provide no added insight.

4.3 Results and Discussion

4.3.1 Results

We commence this section by introducing our chosen evaluation metric, the F1-score. The F1-score was chosen as our evaluation metric because of its ability to return more insightful results regarding the performance of a classification model compared to the simple accuracy score, especially in the context of an entity recognition problem. In entity recognition, positive examples (i.e. actual entities) are generally expected to be distributed in a somewhat sparse manner in the dataset; it is rare to find a NER classification problem where the text is densely populated with entities. Indeed, this is the case with our dataset, where the dataset is written in natural prose and our Person and Location entities are scattered throughout in a sparse manner. Moreover, the data in Section 3.3.1 shows us that no one entity class dominates the statistics, meaning that any binary classification problem regarding these classes would have significantly more negative samples than positive samples. Due to this characteristic of the dataset, it is possible for a model to achieve a very high accuracy score simply by ensuring that the number of true negatives is high. To rectify this issue, we use the F1-score, which is

	Post	Craving	Alone	Mult. friends	1 friend	Family	Partner	Colleagues	Strangers	home
0	Looking for some wisdom I saw here once.\r\nl'...	111	0	0	0	0	0	0	0	0
1	How do you celebrate milestones/goal achieveme...	101	0	0	0	0	0	0	0	0
2	Tomorrow's my birthday and it's triggering mem...	0	110	111	0	0	11	0	0	0
3	I have noticed even my sense of humor has matu...	111	0	0	0	0	0	0	0	0
4	I got a reality check tonight.\r\nl was eating...	101	0	0	111	0	111	111	0	111

Figure 4.7: The dataframe of our manually-labeled Reddit test set. This section of the dataframe primarily shows columns corresponding to Person entities, but the format is identical for the Location entities which we were testing on.

defined as the harmonic mean of the precision and recall. The precision, defined as the number of true positives divided by the number of positives chosen by the model, measures the quality of the positive samples chosen by the model and is low if the number of false positives is high. The recall, defined as the number of true positives divided by the total number of actual positives, measures how well the model is able to extract positive samples and is low if the number of false negatives is high. In this manner, we are able to provide a more robust accuracy measurement that gives us a better idea of how the model is actually performing. We use the F1-score to perform evaluation for both the binary classification models and the multi-class classification models in this chapter. In fact, the F1-score will be the accuracy measure of choice in subsequent chapters as well.

Although the binary classification model introduced in Section 4.2.2 was never going to be our final product using this method, it still yielded results that are worth revealing and analyzing. When trained on an extensive amount of training data (9537 positive reviews and 23,399 negative reviews, as mentioned in Section 4.2.2), our binary classification model trained to distinguish Google Maps reviews of bars and restaurants from Google Maps reviews of other locations performed very well. The full rundown of the evaluation results are given below in Figure 4.3. The F1-score of 0.882 demonstrates our model's strong mastery of the problem it was presented with. It is worth noting as

well that this model was able to return comparable results when trained on a smaller training set.

```
True Positive(TP) = 2042
False Positive(FP) = 279
True Negative(TN) = 4145
False Negative(FN) = 265
Accuracy of the binary classification = 0.919
F1 score = 0.882
```

Figure 4.8: Detailed results of binary classification performed on Google Maps reviews. The model sought to distinguish between reviews on a restaurant or bar and reviews on other locations. Positive = restaurant/bar, negative = other location.

As previously mentioned, though, this problem of distinguishing Google Maps reviews of bars and restaurants from Google Maps reviews of other locations was an easier task to accomplish for a model than subsequent problems that would be more useful for our project. For one, we wanted our binary classifier to be able to work on and distinguish between each of the location classes we defined during our test set labeling, detailed in Section 3.3.1. This would mean that we would need a model to be able to distinguish between a Google Maps review of a bar and a Google Maps review of a restaurant, for example. For another, we wanted our binary classifier to work on the Reddit test set and not just on Google Maps reviews.

As such, we performed one evaluatory run using a similar logistic regression model to perform binary classification on Google Maps reviews on bars only. We wanted to train the model to be able to distinguish between Google Maps reviews on bars and Google Maps reviews on other locations. In particular, the other locations we used this time were the other location classes we defined in Section 3.3.1. This, in turn, would make the classification problem for our model more difficult. The locations used for negative samples for the previous binary classification model were very obviously different from bars and restaurants, which were used for positive samples. The locations used for negative samples here were those that are relevant to our problem and thus had more similarity to bars, making it more difficult to distinguish. Figure 4.9 shows the results of one run of training a model on Google Maps reviews to distinguish between reviews on a bar and reviews on other locations in our pre-defined Locations set.

The next step was to evaluate the model's performance with the Reddit dataset. Since our Reddit test set had already been manually labeled, we were able to use it to evaluate our binary classification model. The results of this experiment were of particular interest, as we would see how the model was able to adjust to data

```
True Positive(TP) = 108
False Positive(FP) = 229
True Negative(TN) = 1284
False Negative(FN) = 73
Accuracy of the binary classification = 0.822
F1 score = 0.417
```

Figure 4.9: Detailed results of binary classification performed on Google Maps reviews. The model sought to distinguish between reviews on a bar and reviews on other locations of interest. Positive = bar, negative = other location.

from a different source. Unfortunately, the results we obtained indicated a very poor performance. The results are given in Figure 4.10 and will be discussed further.

```
True Positive(TP) = 16
False Positive(FP) = 165
True Negative(TN) = 67
False Negative(FN) = 0
Accuracy of the binary classification = 0.335
F1 score = 0.162
```

Figure 4.10: Detailed results of binary classification performed on Google Maps reviews, tested on the Reddit test set. The model sought to distinguish between reviews on a bar and reviews on other locations of interest. Positive = bar, negative = other location.

Multi-class classification is an even more difficult problem to tackle than binary classification is, and the results we obtained seemed to demonstrate this. Below, the results of running our model, trained on Google Maps reviews, on the test set of Google Maps reviews is shown in Figure 4.11. The dataset, which comprises the training data plus the test set, was created by looping through 20 cities for each of ten location entities.

The results we obtained were seen to be satisfactory enough for us to have confidence to proceed with testing the multi-class classification model on the Reddit dataset. As is the case before, this represented a large step up in difficulty for the model because the data samples in the Reddit dataset are from a different domain than the Google Maps reviews data samples. On a test set of 59 valid samples, our multi-class classification model correctly assigned a label to 11 test cases, for an accuracy of 18.64%.

4.3.2 Discussion

In this section, we will analyze the results we obtained in this chapter. In doing so, we will also take a dive into the difficulties encountered, both in regards to the methods

	precision	recall	f1-score	support
0.0	0.48	0.50	0.49	105
1.0	0.07	0.23	0.10	30
2.0	0.73	0.49	0.59	419
3.0	0.45	0.47	0.46	174
4.0	0.05	0.17	0.08	36
5.0	0.24	0.44	0.31	103
6.0	0.52	0.35	0.42	390
7.0	0.24	0.30	0.27	37
8.0	0.35	0.35	0.35	60
accuracy			0.42	1354
macro avg	0.35	0.37	0.34	1354
weighted avg	0.51	0.42	0.45	1354

Figure 4.11: Detailed results of multi-class classification model trained on Google Maps reviews, evaluated on a dataset of Google Maps reviews. The model sought to distinguish between reviews posted on the location categories in Table 4.1.

we implemented and the results we obtained, and the resulting shortcomings. We will also propose further research and methods that could enhance the progress of the work covered in this chapter.

We commence with an analysis of the binary classification models created in this chapter. The first binary classification model, a logistic regression model trained on Google Maps review data to pick out Google Maps reviews that were written on either a restaurant or a bar, performed spectacularly well on our test dataset of more Google Maps reviews. This model laid the foundation for the rest of our models, demonstrating that it was possible for a model to acquire the necessary knowledge to navigate the nuances of Google Maps reviews for a classification task. Although the model for which the results were recorded in this report was trained on large amounts of training data, we were able to achieve similar results with a Google Maps reviews training dataset that was magnitudes smaller. This reinforces the idea stated before that this classification task was not a terribly challenging one for a machine learning model to complete.

When we instead trained a model to specifically extract Google Maps reviews about a bar from among Google Maps reviews about our other pre-defined Location classes, the results were not as excellent. Our second model was a logistic regression model as well, but it was trained on Google Maps reviews data with reviews about bars as the target class. The F1-score achieved by this model was 0.417, due in large part to a poor precision score (108/337). This indicates that the number of false positives was high. Such a predicament most likely arose due to several reasons. The amount of training

data used for this model was significantly less than that for the previous model, for reasons we will expound on at length in our subsequent analysis. Without sufficient training data to fully learn the features that distinguish Google Maps reviews on bars from other Google Maps reviews, our model was not able to identify reviews which were written on similar location classes (e.g. restaurants, supermarkets). Nevertheless, a decent accuracy score of 0.822 gives us reasonable confidence to assume that the model would perform better on this task given more training data.

The same cannot necessarily be said, sadly, of the task where the second model tackled the Reddit dataset as the test set. As can be seen above in Figure 4.10, the results were absolutely atrocious when the second binary classification model was challenged with test data outside of the domain it was trained on. An F1-score of 0.162 emphasizes the wholly unsatisfactory performance of the model, and the precision score of 0.088 (16/181) really hammers it home. The model's poor performance was certainly an effect of the lack of a sufficiently large training dataset. As a result, our model most likely produced an oversimplified feature map for sentences and paragraphs on bars. We hypothesize that too much weight was given to words such as "drinks" and "alcohol" that may show up very frequently in Google Maps reviews on bars. Unfortunately, such words are among the most commonly mentioned words in the Reddit dataset – this would surely help to explain a large part of the very high number of false positives [Kra22]. Our model may have oversimplified the learning process to think that any mention of alcohol must be tied to a bar. This approach may have worked well enough on Google Maps reviews data but fell flat on its face when it was applied to our test data because a lot of our posts are about alcohol even if they do not contain any mention of a bar.

This is a complex issue that could only be solved partially by adding more training data. One idea to remedy this issue would be to fine-tune our binary classification model once it had achieved higher performance benchmarks on a Google Maps reviews test set. In order to prevent our model from overgeneralizing certain correlations or even learning incorrect relationships, we would fine-tune the model by then training it to distinguish between random posts that mention certain keywords and actual Google Maps reviews written on that location class. For example, in training a binary classification model whose target class is restaurants, we would want to prevent the model from overgeneralizing that any mention of food is a sign that the post is about a restaurant. To accomplish this, we could train a binary classification model, which has already been trained to distinguish Google Maps reviews on restaurants from Google Maps reviews on other location classes, to now distinguish Google Maps reviews on restaurants from recipe cards. While we do not expect recipe cards to show up in the Reddit dataset, recipe cards from cooking websites would serve as a useful negative example in this case because they are a real-world example of a list of foods which

does not correspond to a restaurant. In addition, the Internet today boasts countless recipes online from hundreds of cuisines around the world, meaning that our data source would be a vast one.

When evaluating and analyzing the multi-class classification model, the principal difficulty was, once again, not enough data. We will now explain this core issue. Originally, it was the assumption of all the project team members that the free trial of \$300 (300 US Dollars) with the GCP would be sufficient for all of our data needs. This is why we chose this line of development in the first place, as outlined in the beginning of this chapter. Then imagine the great shock we experienced when, with over four months of development time remaining, we ran the \$300 fund dry within a month. In effect, what we had hoped to work on for the better part of four months was rendered impossible after less than four weeks. Over the following month, we attempted to find additional funding through personal and academic connections. In the end, we could only muster up a \$50 credit in partnership with TUM Cloud Computing [Tec21], for which we are grateful.

Admittedly, we were using copious amounts of Google Maps reviews data unabashedly at the start of our development, due to our not having a conceptual understanding of how much \$300 of data represented. Had we managed our resources and funds meticulously from the start, this problem may not have been as detrimental as it ended up being. But finishing our \$300 free trial meant that we had to upgrade to a paid account, meaning that any penny spent over the amount we had linked in the account would have to come out of our own pockets. Due to the scope and funding (none) of this project, going over the spending limit had to be avoided at all costs. This is the reason why our data usage plummeted – not only did our budget shrink (from \$300 to just \$50), we now operated under significantly more pressure to ensure that our spendings were well below the limit. As such, our multi-class classification model was fed far less training data than our initial binary classification model was, despite having to tackle a problem with a much higher degree of difficulty. The results we achieved, then, were predictable yet admirable.

The issue of budget also completely nullified the possibility of a line of research that we were looking to embark on. Our original plan, as discussed in a team meeting, was to track the F1-score as we tweaked a number of parameters – the number of location classes used, the number of data samples we took, etc. There were even plans to integrate the Google Maps geolocation functionality. This line of research would have required magnitudes more data than what we ended up using for what we were eventually able to accomplish. If we were not even able to train our binary classification and multi-class classification models to a satisfactory level with the budget we had, there was no way a task like this stood any chance of succeeding.

Another thing we definitely could have done was to fine-tune the dataset for the

multi-class classifier a lot more. There were probably a substantial number of overly brief reviews like "Very good" that we could have filtered out so that every single review would be lengthy and informative. In fact, Figure 4.3 shows one such review in the fifth row. Cleaning the data in this way would have been a simple step that could yield profitable results. However, our capabilities were already greatly limited at this point – any additional training run of the model would consume more data, so this line of research was not continued.

5 Rule-based Labeling Approach

5.1 Introduction and Background

We have seen in previous sections that the lack of annotated data was a recurring issue for us as we progressed through the stages of this project. Without sufficient training data, learning becomes very difficult for any model. Thus, navigating this issue was a significant motivator in many of the development decisions we made. In Chapter 4, we sought an external data source with potentially noisy labels in an attempt to alleviate the problem. In this chapter, and the next, we seek to bridge the gap by labeling our own training data, albeit in an automated fashion. We used a rule-based approach to extract and label entities in our training data, which was then used to train our model.

We should expound further upon our motivation with this method here. The rules we abided by during our labeling of the test set are explained in Section 3.3.1. A careful examination of the rules there will reveal that the rules described there are somewhat rigid – we often require that an entity be mentioned explicitly in order to tag it, and we shy away from assigning a label when a person or location is implicitly implied through the context specified in the post but not explicitly mentioned. This reason provided the motive for us to train prospective models on manually labeled data. If our rule-based labeling could take care of the majority of cases, then a model trained on our manually labeled data could return strong performance on the test set as well.

5.2 Methods

5.2.1 Data Labeling

Rule-based labeling

We drew inspiration from Section 3.3.1 to set up our rule-based labeling system. We see that our labeling rules generally require the explicit mention of an entity, be it a person or a location or an emotion, or its synonym for us to assign the corresponding label to the post. As such, we defined a short list of synonyms for each entity that we were working with for this section of the project. These became our search terms and keywords for automated data labeling, and they are broken down in Table 5.1.

Table 5.1: A list of the keywords we used for automated manual data labeling, broken down by the corresponding entity.

Test set entity	Keywords
Person:alone	"alone", "by myself"
Person:one_friend	"friend", "buddy"
Person:multiple_friends	"friends", "buddies"
Person:family	"family", "families", " son ", "daughter", "father", "mother", "mom", "dad", "brother", "sister", "bro ", "uncle", "aunt"
Person:partner	"partner", "husband", "wife", "hubby", "boyfriend", "girlfriend", "gf", "bf", "GF", "BF", "SO"
Location:home	"home", "house"
Location:school	"school", "university"
Location:work	"workplace", "job", "from work", "at work", "my work", "my office"
Location:bar	"bar"
Location:supermarket	"market", "store", "supermarket"

There are a few things of note to mention about the keywords we used. First of all, some keywords had to be adjusted in order to ensure that false positives were not returned. This was something we saw very early on in the project when the craving filter for our Reddit dataset searched for the string "urge" and humorously returned posts about burgers. For example, one can see that the search string "bro " has a space at the end of it, and this is to eliminate other words beginning with "bro-" such as "broken" and "brown" and "bronchitis." Similar, the search string " son " has a space affixed at both the beginning and end of the word, since we do not want to select words such as "songwriter" or "reason" or even "resonance." Second of all, some seemingly obvious keywords were omitted, for a similar reason as in the case above. For example, "work" is not a search string for the entity Location:work, and this is owing to the fact that many sentences that use the word "work" (e.g. "Taking painkillers doesn't work on me!") do not actually refer to an actual job or workplace. A sharp-minded reader will realize that something similar is true for the search string "job" (e.g. "Good job getting to ten days sober!"), but a cursory analysis of the Reddit dataset revealed that this was a less common scenario than the above scenario with the search string "work." Third, we would just like to quickly mention that the list of synonyms that made up our search strings list was not designed to be extremely extensive. We were hoping

that our models would perhaps be able to slowly learn the synonyms by themselves during the training process.

With our list of keywords defined, we would then iterate through the text to find these entities and then label them into a dictionary. This became our training data. Below is an example of what a labeled post would look like, rendered with Displacy. While the post in Figure 5.1 is not an actual post from our dataset, it serves to demonstrate the labeling capabilities of our rule-based automated data labeling system.



Figure 5.1: A visualization of entities that our rule-based labeling approach is able to extract and label, provided by Displacy.

Training data format

In order for our model to work, we had to convert our entities to store them in a way that a model could understand and be trained on. The format that we stored the entities was in a dictionary consisting of the original text, the entity label, and the start and end indices of the entity within the text. The accumulation of these dictionaries became a JSON file. Thus, this was the target data format when performing our rule-based automated data labeling system.

One other data format was used in this chapter – a spaCy format that is similar to the JSON format yet subtly different. A Python function was used to convert between the JSON format and the spaCy format.

5.2.2 Machine Learning Model

The general pipeline of the model we employed in this chapter was based off of a model designed to extract custom entities from resumes [Gup18a]. The model utilized a manual data annotation tool named DataTurks. Unfortunately, this tool appears to have been deprecated, as numerous repeated efforts to get the tool working online only resulted in failure. Following this, a Linux application version was trialed, but was also entirely unable to function. In the end, this served as motivation for us to set up the automated manual data labeling pipeline explained in this chapter. In addition, the complete unusability of DataTurks pushed us to find a better, working data annotation system later on in our project. This is how Doccano, introduced in Section 3.3.3, came to be used to create our test set for NER.

Our machine learning model used in this chapter was a simple and more primitive model because this was our first stab at working with manually labeled training data in our exploratory analysis. The backbone of the model was a spaCy blank Language class with a NER pipeline. A now-deprecated spaCy package called GoldParse was used to assist with parsing the test data and obtaining the accuracy statistics.

5.3 Results and Discussion

5.3.1 Results

Below are the results we were able to achieve with this model, which learned our dataset based on rule-based manually-labeled training data. The results listed in Table 5.2 are the results when the model was evaluated on 21 documents which were manually labeled by a human using Doccano. Table 5.3 then provides the compilation of our accuracy statistics across all documents, broken down by entity type.

21 posts from the Reddit dataset were used as the test set, but only Documents 0 through 13 contained entities relevant to our project. The model was successfully able to avoid labeling any false positives in the remaining seven documents. Hence, the document number in Table 5.2 only goes up to 13.

The micro- and macro-averages for the precision, recall, and F1-score were calculated as follows: the micro-average, also known as the weighted average, is calculated by considering each individual item and summing up the numerator and denominator. The macro-average, on the other hand, is found by simply finding the average when each entity type is weighted evenly.

5.3.2 Discussion

We commence the discussion by analyzing the results we obtained in a more detailed way. By observing the totals in Table 5.3, we can see that the overall F1-scores of 0.730 (micro) and 0.610 (macro) are acceptable, but not outstanding.

Upon closer examination, the results of the `Location:work` entity class begin to stand out, and not in a good way. We can see that the majority of the precision and recall numbers in the table are relatively high and close to 1, but entities in the `Location:work` class returned a recall score of 0/9. This means that out of 9 actual appearances of `Location:work` entities in our texts, our model returned 0 of these, resulting in 9 false negatives. This can be explained by the keywords we used as search strings from Table 5.1. As mentioned in the discourse regarding the table and its contents, we chose to completely omit the keyword "work" when labeling the training data, hoping that the other search strings such as "at work" and "from work" would teach

Table 5.2: Detailed evaluation results for the rule-based labeling model, broken down by document then entity type.

Doc. #	Entity type	Precision	Recall	F1-score
0	Location:work	0/0	0/2	0.000
1	Location:work	0/0	0/2	0.000
1	Person:partner	1/1	1/1	1.000
2	Location:work	0/0	0/1	0.000
2	Location:home	1/1	1/1	1.000
2	Person:one_friend	1/1	1/1	1.000
3	Person:partner	0/0	0/1	0.000
4	Person:partner	1/2	1/1	0.667
4	Person:family	2/2	2/2	1.000
4	Location:home	0/1	0/0	0.000
4	Person:one_friend	0/0	0/1	0.000
5	Person:family	0/0	0/1	0.000
5	Location:work	0/0	0/2	0.000
5	Location:home	3/3	3/3	1.000
6	Person:partner	1/1	1/1	1.000
6	Location:home	3/3	3/3	1.000
7	Person:partner	1/1	1/1	1.000
8	Location:home	1/1	1/1	1.000
9	Location:home	1/1	1/1	1.000
9	Person:multiple_friends	1/2	1/1	0.667
10	Location:supermarket	1/1	1/2	0.667
11	Location:work	0/0	0/1	0.000
11	Person:family	1/1	1/1	1.000
11	Person:partner	1/1	1/1	1.000
12	Person:partner	1/1	1/1	1.000
13	Person:family	1/1	1/2	0.667
13	Location:work	0/0	0/1	0.000
13	Person:partner	1/1	1/1	1.000
13	Location:bar	0/1	0/0	0.000

Table 5.3: Detailed evaluation results for the rule-based labeling model, broken down by entity type across all documents. The micro- and macro-averages for precision, recall, and F1-score are also then provided.

Entity type	Precision	Recall	F1-score
Location:work	0/0	0/9	0.000
Person:partner	7/8	7/8	0.875
Location:home	9/10	9/9	0.947
Person:one_friend	1/1	1/2	0.667
Person:family	4/4	4/6	0.800
Person:multiple_friends	1/2	1/1	0.667
Location:supermarket	1/1	1/2	0.667
Location:bar	0/1	0/0	0.000
Micro total:	23/27 = 0.852	23/36 = 0.639	0.730
Macro total:	211/320 = 0.659	109/192 = 0.568	0.610

the model to isolate mentions of "work" that actually correspond to a job or workplace. Unfortunately, through the test set results we can see that this clearly did not happen. With presumably the vast majority of tokens reading "work" not marked as an entity, the model learned not to classify tokens reading "work" as a Location:work entity.

While moving the goalposts is generally frowned upon, it is clear to see that this was a difficult case for the model to master due to the conflicting information it would have received during the training process. Moreover, since our study is an exploratory analysis of sorts, one of our objectives is to figure out what works and what does not work. So if we temporarily disregard this case as an outlier of sorts, then we see that the micro-average recall increases from 0.639 to a strong 0.852, while the macro-average recall increases from 0.568 to 0.649. The micro-average F1-score would then jump up to 0.852, a much stronger figure.

Alternatively, we could have trained the model again with "work" as one of the search strings for the Location:work entity. This would surely bump up the number of false positives come testing time. However, the recall is arguably more important when it comes to an entity recognition problem, where we would like to extract as many entities as possible. In the trade-off between precision and recall, eliminating false negatives at the expense of allowing more false positives may be seen as beneficial in a NER task. We would like a model performing NER to extract as many entities as possible.

Continuing with our analysis of our model's evaluation results, there are two false positives that we would like to discuss that are somewhat relevant to the discussion

we just had on false positives. Document 4 had a mention of a "house plant test" which was not labeled as a `Location:home` entity. Our model, which was trained with the information that the keyword "house" always equates to a `Location:home` entity, logically but incorrectly labeled "house" in the phrase "house plant test" as a `Location:home` entity. This is a reasonable false positive given our rule-based labeled training data set. Similarly, Document 9 concluded with a message addressed to fellow members of the `/r/stopdrinking` forum: "Sober on my friends." We decided early on in our test set labeling process documented in Section 3.3.1 that such general mentions of "friends" as a way of addressing anonymous readers would not be labeled as a `Person:multiple_friends` entity. However, our model again learned based on our training data that all mentions of the keyword "friends" are `Person:multiple_friends` entities, and thus incorrectly extracted this mention of "friends" as such an entity.

Another false positive came about in a similar and humorous way. The author in Document 13 defeated her strong sorrow-driven cravings by instead "eating a huge bar of chocolate." Of course, our model picked up on the keyword "bar" and labeled it as a `Location:bar` entity. This once again demonstrates a potential pitfall of manually labeling data – it is easy to miss the nuances of language when applying general, sweeping rules. In any case, by removing this one mistake, in conjunction with the adjustments we made above surrounding the `Location:work` entity, we are able to boost our macro-average precision and recall to 0.879 and 0.757, respectively, resulting in an improved macro-average F1-score of 0.814.

Two false negatives from the evaluation of our model are worth discussing here as well. In one particularly difficult case, the author mentioned that she was in a new relationship, and immediately proceeded to quote her partner, using the pronoun "he" to indicate that her partner was speaking. However, throughout the entire post, there was never an explicit mention of her partner. As such, this post was labeled as belonging to the `Person:partner` class during our test set labeling process, but our model was unable to find the actual entity corresponding to the `Person:partner` class. One possible remedy for this predicament for us to explore would be co-reference resolution, introduced in brief in Section 1.2.2. This would, however, be a particularly challenging case of co-reference resolution due to the fact that the true antecedent of the pronoun "he" is never mentioned. The second false positive case is a much simpler one: "kid" or "kids" (and other potential search strings like "child" or "children," for that matter) were omitted as keywords for the `Person:family` entity class due to oversight on our part. The simple remedy of adding another few entries to our search strings, documented in Table 5.1, would most likely overcome this problem.

While we see in this section that the results we obtained from our model were quite solid, there are a few things that we must consider as well. The test set consisted merely of 21 Reddit dataset posts with a total of 36 labeled entities between them. With such

a small test set, our confidence that our evaluation results are representative of the model's performance on all data is very low. In fact, more than one entity class did not show up a single time in our test data, meaning that our model's performance on such entities was not tested at all. Still, even with this in mind, our model performed stoutly on our entity recognition problem using rule-based automated manually-labeled training data. This allowed us to move on to more state-of-the-art methods.

6 Spark with BERT

6.1 Introduction and Background

In previous chapters, we have seen that much of our project has boiled down to the pursuit of labeled training data. Indeed, NLP is a field which has been swept up in the deep learning revolution, with Deep NLP playing an integral role in achieving state-of-the-art results in many NLP tasks. This demonstrates that one of the keys to great success in the NLP field is massive quantities of training data. The problem, of course, is that large quantities of labeled training data are not always readily available. As an attempt at overcoming the issue of lack of labeled training data, numerous methods have been devised with the goal of using large quantities of unlabeled data during a pre-training step, followed by fine-tuning on smaller quantities of labeled, task-specific data [Dee21].

BERT, which stands for *Bidirectional Encoder Representations from Transformers*, is one such machine learning technique. It leverages a Transformer mechanism to learn the contextual relations between words in a text. As indicated in its name, BERT differs from other language models due to the fact that it learns bidirectionally – both in a left-to-right direction and a right-to-left direction. In doing so, BERT becomes a language model with a strong sense of language context that can then be fine-tuned for more specific tasks, such as NER [Dee21].

BERT uses the Transformer model architecture, which generally consists of two parts: an encoder and a decoder. The encoder is in charge of reading the text input, while the decoder attempts to output the task prediction. While only the encoder is necessary for BERT to function, it is still useful to consider both the encoder and decoder and their roles. For BERT, the encoder receives as input a sequence of tokens representing natural language, which is then transformed into vector embeddings. The model then attempts a series of two tasks during the training process. The first is a masked language model (MLM), where the model receives input sentences, but with some proportion of the tokens (usually 15%) masked. The model's task is to use the context surrounding the masked tokens to figure out what the masked token should actually read. The second is next sentence prediction (NSP), where the model receives input sentences in pairs and learns to determine if the second one is a logical sentence to come after the first [Dee21].

Four pre-trained BERT models are available. The names of the four BERT models are

`bert_base_uncased`, `bert_base_cased`, `bert_large_uncased`, and `bert_large_cased`. The base and large refer to the size of the model – the base model has 12 layers and about 110 million parameters, while the large model has 24 layers and about 340 million parameters. Whether or not the model is cased or uncased depends on how it formats its data – the uncased models convert the input text into lowercase while the cased models preserve the casing of the input text [Dee21; Koc19].

In this chapter, we also make use of Spark NLP, which is an open-source NLP library built on top of Apache Spark that aims to serve as a comprehensive suite of NLP tools. Spark NLP is supported by John Snow Labs and covers many NLP tasks, such as tokenization, POS tagging, and NER [Joh22b; Koc19].

6.2 Methods

6.2.1 Data Considerations

For this chapter, we repeated the process from Chapter 5 to produce rule-based automated manually-labeled training data for the model. We followed the same procedure with the same rules to extract and label entities from the training data for our model to learn from.

One significant difference with regards to the data for this method was the training data format. Whereas we converted our manually-labeled training data to a spaCy format in Chapter 5 due to the fact that we used a spaCy blank Language class model, our training data here was converted to CoNLL format. CoNLL stands for Computational Natural Language Learning conference, and a 2003 gathering of the conference produced the CoNLL-2003 dataset. The CoNLL-2003 dataset is a NER dataset that was released during the 2003 CoNLL conference, designed to serve as the training dataset for a language-independent NER task. While we do not use the CoNLL-2003 dataset for our project, we used the data format to send input to the model. And while the CoNLL-2003 dataset labeled its tokens with CoNLL-U labels consisting of I, B, O, and POS tags, we modified the pipeline so that our labels were our usual Person and Location tags, along with the corresponding entity classes outlined in Section 3.3.1. After the data was converted into CoNLL format, it had to then be converted into a dataframe. Below, Figure 6.1 shows a partial view of entities extracted for the training data.

6.2.2 Model

The Spark NER pipeline provided the framework for the model we used. There were two main stages to our Spark NER pipeline, which we will detail here. The first stage

text	document
house	[[{document, 0, 4, house, {training -> true}, []}]
store	[[{document, 0, 4, store, {training -> true}, []}]
friends	[[{document, 0, 6, friends, {training -> true}, []}]
at	[[{document, 0, 1, at, {training -> true}, []}]
store	[[{document, 0, 4, store, {training -> true}, []}]
partner	[[{document, 0, 6, partner, {training -> true}, []}]
house	[[{document, 0, 4, house, {training -> true}, []}]
home)	[[{document, 0, 4, home), {training -> true}, []}]
bar	[[{document, 0, 2, bar, {training -> true}, []}]
at	[[{document, 0, 1, at, {training -> true}, []}]
friend	[[{document, 0, 5, friend, {training -> true}, []}]
bar	[[{document, 0, 2, bar, {training -> true}, []}]
friend	[[{document, 0, 5, friend, {training -> true}, []}]
bars	[[{document, 0, 3, bars, {training -> true}, []}]
wife	[[{document, 0, 3, wife, {training -> true}, []}]
daughter's	[[{document, 0, 9, daughter's, {training -> true}, []}]
home	[[{document, 0, 3, home, {training -> true}, []}]
house	[[{document, 0, 4, house, {training -> true}, []}]
wife	[[{document, 0, 3, wife, {training -> true}, []}]
wife	[[{document, 0, 3, wife, {training -> true}, []}]
store	[[{document, 0, 4, store, {training -> true}, []}]
home,	[[{document, 0, 4, home,, {training -> true}, []}]
store	[[{document, 0, 4, store, {training -> true}, []}]
wife	[[{document, 0, 3, wife, {training -> true}, []}]

Figure 6.1: A snapshot of the entities extracted for our Spark with BERT model, presented as a dataframe.

of the pipeline was BERT embeddings. We used the cased version of the BERT Base model in English, `bert_base_cased`. In short, the BERT embeddings stage takes in the input columns from our dataframe and returns an output column of the BERT vector embeddings of the input columns. The vector embeddings have 768 dimensions because we used the `bert_base_cased` model. The second stage of the pipeline was a `NerDLApproach` stage. The `NerDLApproach` allows us to train a generic NER model based on deep learning approaches [Joh22a; Koc20].

We modified many parameters to get the functionality we desired. We were able to modify the number of epochs to train for – for most runs, we chose 5 as the number of epochs as a balance between strong results and training time. The validation split was set to 0.2 to allow for a substantial proportion of the samples to be used as validation during the epochs. We set the test data set to a file we generated by converting the Doccano test set, introduced in Section 3.3.3, into the CoNLL format [Koc20].

6.3 Results and Discussion

6.3.1 Results

As mentioned above, we often trained our model for five epochs. Below, the results obtained by our model after five epochs are shown in Figure 6.2.

```
Epoch 5/5 started, lr: 9.803922E-4, dataset size: 119

Epoch 5/5 - 0.50s - loss: 11.056543 - batches: 15
Quality on validation dataset (20.0%), validation examples = 23
time to finish evaluation: 0.03s
label tp  fp  fn  prec  rec  f1
Person:multiple_friends 3  0  0  1.0  1.0  1.0
Location:home 2  0  0  1.0  1.0  1.0
Person:alone 2  0  0  1.0  1.0  1.0
Location:work 1  1  0  0.5  1.0  0.6666667
Person:partner 6  0  2  1.0  0.75  0.85714287
Person:family 10 1  0  0.90909094 1.0  0.95238096
Person:one_friend 1  0  0  1.0  1.0  1.0
tp: 25 fp: 2 fn: 2 labels: 7
Macro-average prec: 0.91558444, rec: 0.96428573, f1: 0.9393043
Micro-average prec: 0.9259259, rec: 0.9259259, f1: 0.9259259
```

Figure 6.2: Results obtained by our Spark with BERT model after five epochs.

As in the previous chapter, the micro- and macro-averages for precision, recall, and F1-score are given here.

6.3.2 Discussion

We begin by addressing the elephant in the room: the results shown just above are not the result of evaluation on our Doccano test set. Instead, they represent validation results which were calculated at the end of each epoch. While the data was split into training and validation data before the training process, meaning that the validation dataset could serve decently as a test set, such a test set is still from the same domain as the training set and was labeled in an identical manner to the training set. To some extent, this defeats the purpose of our experimentation with this method, since we wanted to test the model's proficiency when dealing with our human-labeled test set. We now move on to explain the overall process for this method, along with some of the difficulties we encountered. Though much effort was expended to coax the proper behavior out of our model, much of our efforts was expended in vain due to two principal reasons.

When working with a model which has its overall structure provided by an external application, there will always be potential risks and pitfalls. In our case with Spark, we were unable to get the testing function to work. Despite our adherence to the examples and instructions set forth and manual tinkering with numerous functions and arguments, the model was unable to be evaluated on the test set. Originally, the model was supposed to be automatically evaluated on the test set parquet after the training process was complete. There was, however, an alternative method to manually evaluate the model on the test set. Regrettably, this also did not seem to work, as we received a strange output that appeared to indicate that every single word in every single sentence of every single post was an entity of some sort, which we knew not to be the case.

Another issue plagued our deployment of this model as well – an extremely strange bug that seemed to manifest itself only when the number of entity classes we used was exactly ten. It later transpired that this was an issue tied to the Spark NLP package that we used. In defining the underlying neural network architecture, the Spark NLP package would try to fit a pre-defined TensorFlow graph containing the following parameters: entity tag classes, number of dimensions for the embedding, and number of chars [Joh22c]. The problem arose when the number of entity tag classes was exactly ten because the package would attempt to fit one of its pre-defined ten-entity-tag graphs, resulting in an obscure error [Joh22c]. This error flummoxed us for an extended period of time because it was difficult to make the logical connections of the steps leading up to this error. In retrospect, it is clear to see that the ten entity classes was somehow causing the problem. However, this did not logically make sense during the debugging process and it made much more sense at the time to examine the implementation of the model and other processes instead. This was a bug that was extremely difficult to diagnose but simple to fix.

In spite of all this, we can see that the validation results we obtained were very strong. After five epochs, our micro-averages (corresponding to the total score when each entity is considered as one) for precision and recall were 0.916 and 0.964, respectively, for a F1-score of 0.939. When considering the macro-averages (corresponding to the average obtained when considering scores by category), the precision and recall were both 0.926, resulting in a F1-score of 0.926. These results indicate a lot of promise for this method. In particular, the high recall values are very encouraging because a NER algorithm should seek to successfully extract as many entities as possible and miss as few entities as possible.

Because the entities from the training set were stored in a dataframe for our Spark with BERT approach, we were able to gain even more of an insight on potential pitfalls with keyword-based entity labeling than we did in the previous chapter. We realized that there were additional issues with our keywords that would have to be fixed if we desired to polish this approach into something that could be deployed. In Chapter 5, we saw that a mention of a "chocolate bar" was erroneously but reasonably classified as an instance of a `Location:bar` entity. In this chapter, another more potentially harmful false positive came up: the word "barely" was classified as an instance of a `Location:bar` entity. This was because our search string for `Location:bar` was simply "bar" due to the fact that we wanted to include words like "bartender." However, this had the unwanted consequence of including a very common English word such as "barely." This would be unacceptable in a final product. Similar situations were unearthed through an examination of the training set dataframe: "momentary" was a `Location:family` entity because it contains the search string "mom," and someone who was "VERY SOBER" got classified as a `Person:partner` entity because "SO" was a search string for `Person:partner`.

In all, the results in this chapter are incomplete yet promising. We have reasonable confidence to conclude that this methodology has great potential to be very successful on our NER test set.

7 Bi-LSTM

7.1 Introduction and Background

When working with large quantities of data, one has to consider the data format in order to most effectively work with it. For example, image data is treated differently from, say, two-dimensional binary data because the location of each pixel in the image is important. If we are not able to know which pixels are adjacent to each other, the image data loses much of its meaning. This is why image data is often fitted with a convolutional neural network (CNN) – a class of neural networks that learns image features based on a series of two-dimensional filters. By using the right tool, a CNN, much of the original data’s structure and meaning can be preserved and learned by a model, which in turn leads to higher accuracy come test time.

In a similar manner, we must consider how best to represent the text data that is found so often in NLP datasets. When considering human written or spoken language, we can clearly see that the words that fill up a dialogue are by no means independent data points. Rather, the words in a text-based dataset create a context, which links the individual words together. This is why we are able to make sense out of a string of words, such as this current sentence, and it is also why we are often able to fill in blanks in a sentence with reasonable prediction words. Words derive their meaning based on the context provided by words that come before and after them in a word sequence. Because of this, it makes sense to use a structure that will take the data’s sequential nature into account when constructing a model to learn about text-based data.

A Bi-LSTM is one such network architecture. Short for *Bidirectional Long Short-Term Memory*, a Bi-LSTM network is a type of recurrent neural network (RNN) architecture that is able to learn order dependence when dealing with sequential data [Bro17]. A RNN is sequential in nature and allows for the creation of cycles between nodes, making one an apt choice when dealing with text-based data which reads in one direction and has words that can affect other words both before and after it. A LSTM network, in particular, is a special breed of RNN that preserves previous information for longer to form long-term dependencies [Ver21]. In a standard LSTM node, there are four neural network layers that are already trained, plus simple mathematical operations and concatenation and copying mechanisms that work on the input of each node. In the meantime, the output from the previous node is preserved and combined with

the new input, allowing for a feeding forward of the information gained from each node [Ver21].

A Bi-LSTM, then, can be likened to a combination of two RNN's put together – one running forwards and one running backwards [Agg19]. With a Bi-LSTM, inputs will run both from the past to the future and from the future to the past. Using the hidden layers of the LSTM, a Bi-LSTM allows us to reference the context of the sentence in both directions when looking at a specific word. With our data, we felt that gaining insight in this way would be vital for a model to successfully predict whether or not a specific word belongs in one of our entity classes defined in Section 3.3.1.

7.2 Methods

7.2.1 Data format

As was the case in Chapters 5 and 6, we used our automated rule-based manual data labeling approach to build up the training dataset for this approach. The data for this approach was converted to a .csv file for the model. The data format was similar to the CoNLL format used in Chapter 6, but each word token that was not any type of entity also had to be labeled. As such, we created a new class, 0, to designate words that did not belong to any of our entity classes.

7.2.2 Model

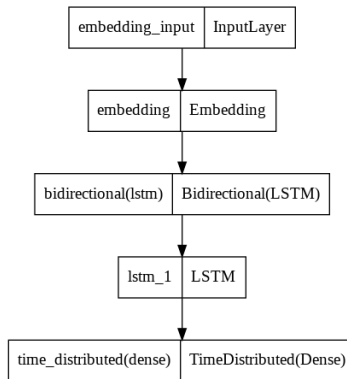


Figure 7.1: Our Bi-LSTM model architecture, generated by Keras.

Based on the work of [Nai20], the model we employed for this chapter had four main layers: an embedding layer, a Bi-LSTM layer, another LSTM layer, and what is called a TimeDistributed layer. The first embedding layer takes in our input data, in the form

of tokenized sentences, and converts them to 64-dimensional vector embeddings. The second layer, the Bi-LSTM, takes in input from both the previous layer (the embedding layer) and the next layer (the LSTM) in a bi-directional manner. The two outputs produced, one from a forwards pass and one from a backwards pass, are concatenated and passed on to the next layer. The third layer is a simple LSTM, while the final layer is a TimeDistributed layer. The TimeDistributed layer is a dense, fully-connected layer that allows each of the timesteps to be connected.

Before the data was fed into the first layer of the model, it had to be processed. This included splitting the data into training and validation examples, but also preparing the data so that it would match the format that the model expected. When the training and fitting process for the model involved mini-batches, each sequence in the input data had to have the same length. There were two solutions to this: data padding or data truncation. In data padding, we would find the length of the longest sentence in the training set, and pad all other sentences at the end with generic empty characters that all have a label of 0, representing a token that does not fall under one of our entity classes. In data truncation, we set a maximum sentence length – all sentences longer than this length are truncated from the end, while all sentences shorter than this length are padded at the end. When training and fitting the model without batch data, neither solution was necessary.

In order to gain more insight as to how our model was performing, we divided our experimentation into two cases with regards to the entity classes. In one case, we used our usual fine-grained Person and Location entity classes, detailed in Section 3.3.1. This was referred to as fine-grained entity tags. In another case, we simplified the entity tags to just two very general classes: Person and Location. The method by which our training data was labeled was still the same – the only difference was that we would not specify which kind of Person or Location entity an extracted token was. This was referred to as simple entity tags.

7.3 Results and Discussion

7.3.1 Results

Table 7.1 documents the results we achieved using the Bi-LSTM method. The evaluation results are divided by the data pre-processing method used, whether or not we fed the data to the model in batches, and which type of entity classes we trained and evaluated the model on.

Table 7.1: Evaluation results for our Bi-LSTM NER model. Models differ based on how the input data was treated, whether the data was fed in batches, and which NER tag set was used.

Data pre-processing method	Batch data	Entity tags	Val. accuracy	Test accuracy
Padding	yes	Fine-grained	0.9993	0.9821
Padding	yes	Simple	0.9993	0.9821
Truncation (maxlen=20)	yes	Fine-grained	0.9944	0.8500
Truncation (maxlen=20)	yes	Simple	0.9944	0.8500
none	no	Fine-grained	0.9944	0.5000

7.3.2 Discussion

At first glance, the results we obtained using a Bi-LSTM model appear to be quite strong. A validation accuracy of 0.9993 is astoundingly good, and the test accuracy of 0.9821 is excellent as well. However, a deeper examination of the results reveals a very concerning fact that nearly renders all the work we did in this chapter useless.

First, we will take a look at the training set we used for this chapter. The label distribution for NER tasks is inherently quite sparse. This makes sense conceptually – if a text were littered with numerous named entities, occurring more frequently than other non-entity words, this somewhat defeats the purpose of NER. NER’s value is in extracting the most important and most relevant pieces of information from a text that is otherwise full of non-entity tokens. The above fact concerning NER label distributions is even more true for our dataset, where many words are spent by post authors on other topics besides Person and Location entities.

For the most extreme case, let us take a look at the case where the data pre-processing method is padding. Our training set has approximately 15,000 tokens, distributed across around 1000 sentences. Among the 15,000, just under 150 were labeled by our automated rule-based manual labeling approach, representing an entity proportion of under 1%. To restate, fewer than 1% of the tokens in our training dataset were extracted as entities that fit in our entity classes – this means over 99% of tokens in our training dataset had the label 0, corresponding to tokens that did not belong to any of our entity classes. A sharp-minded reader may already know where this is headed.

In addition, the longest sentence in our training corpus was 168 words long – perhaps somebody forgot to put periods in his or her writing. As a result, every sentence in

our training dataset was padded to be exactly 168 words long. Each sentence was post-padded with generic non-entity tokens, adding to the number of tokens in our training dataset that have the label 0. If each of 1000 sentences was 168 words long, that means that are now approximately 168,000 tokens in our training dataset, of which under 150 are labeled entities. This now represents an entity proportion of under 0.1%. Less than 1 in 1000 tokens in our training dataset was a labeled Person or Location entity.

As such, our Bi-LSTM model found a way to achieve very high accuracy while expending as little learning effort as possible: guess 0 for every single token in the corpus. Since over 99.9% of tokens in our training dataset had the label 0, this was an approach that yielded results that appeared very strong outwardly. It was a quick and easy approach as well – the training logs reveal that our model converged on this solution after just one epoch.

```
Model: "sequential_10"
=====
Layer (type)                Output Shape                Param #
=====
embedding_10 (Embedding)    (None, 168, 64)            199552
bidirectional_10 (Bidirecti (None, 168, 128)            66048
onal)
lstm_21 (LSTM)              (None, 168, 64)            49408
time_distributed_10 (TimeDi (None, 168, 12)            780
stributed)
=====
Total params: 315,788
Trainable params: 315,788
Non-trainable params: 0
=====
22/22 [=====] - 26s 743ms/step - loss: 0.1835 - accuracy: 0.9592
22/22 [=====] - 15s 706ms/step - loss: 0.0127 - accuracy: 0.9992
22/22 [=====] - 15s 704ms/step - loss: 0.0127 - accuracy: 0.9992
22/22 [=====] - 16s 714ms/step - loss: 0.0127 - accuracy: 0.9992
22/22 [=====] - 16s 719ms/step - loss: 0.0127 - accuracy: 0.9992
```

Figure 7.2: The training log of the Bi-LSTM model – note the high accuracy and the fact that nothing seems to change after the second epoch.

To remedy this problem, we tried approaches without padding and without batch data, and even tried giving the model a simpler task by limiting entity tags to just Person and Location. Unfortunately, the test results for each and every one of these cases revealed that our Bi-LSTM model learned the same approach: tag every token with 0. The reason that the cases with padding as the data pre-processing method yielded the highest test accuracy is because the test samples had to be padded as well, giving the model more free points. In contrast, the case where we did not use batch data yielded the lowest accuracy because there was no padding at all and the test set

thus contained the highest proportion of entities that should have been labeled. In any case, though, the recall for every single one of these models was 0, because there were no true positives. If the F1-score was taken as our evaluation metric, each of our models would have achieved an F1-score of 0.

Unfortunately, given the time constraints of this project, no further fixes were able to be applied to the methods in this chapter. One solution was considered – pad the sequences with labeled entity data, instead of generic non-entity tokens. However, this would somewhat defeat the purpose of using a Bi-LSTM. A Bi-LSTM thrives by analyzing the context around individual tokens, and a meaningless list of labeled entity data (e.g. "I went to the bar with my friends the other day brother supermarket house supermarket brother brother") would not only not provide such a context but also even potentially be detrimental to the model's learning of the context created in the beginning of the sentence. After all, a Bi-LSTM is bidirectional, meaning that the context created when reading the sentence from the end to the beginning is important as well. It is then a possibility that even post-padding the sentence with meaningless tokens had this undesirable effect. However, the fact that the case with no padding and no truncation returned similar results seems to indicate that there was more at play than just this potential issue with padding.

Of course, the disappointing results of this chapter do not necessarily indicate that a Bi-LSTM cannot be made to perform adequately on our test dataset. Rather, it is best to consider the results of this chapter as inconclusive – we were not able to build a model that achieved success on our NER dataset, but we were also not able to definitively prove that a Bi-LSTM is ineffective on our data.

8 Conclusion

The AlcPrIntTUM project is a collective effort to use the computing tools we have at our disposal to aid in the alcohol addiction prevention and intervention processes. Under the AlcPrIntTUM project hierarchy, the project documented in this work originally sought to perform analysis on how entities extracted from natural language posts in self-help-groups could help in creating a more in-depth knowledge model on alcohol cravings. We pivoted to a different task, focusing instead on how to extract the custom entities in an accurate and useful manner. This work documents the insights we obtained during our exploratory analysis on applying NLP techniques to alcohol self-help-group texts.

The contributions made by this work are plentiful. As a team, we found a suitable dataset for our work, termed the Reddit dataset, and figured out how best to process it. We unveiled patterns and relationships within the data by using classic NLP techniques such as word-level clustering, sentence-level clustering, and sentence similarity scores. We definitively concluded that current NER libraries available online are unable to operate on our domain. We also created a large and extensive labeled test set, which we believe to be the only one of its kind. Such a test set will continue to be useful as development continues on the AlcPrIntTUM project.

We also made great strides forward in the evaluation of the feasibility of multiple innovative NER methods. While the approach utilizing Google Maps reviews in Chapter 4 was ultimately cut short by a lack of funding, great insights were gained from the execution of the methods in the chapter. We explored the idea of sentence similarity scores, which could be extended in future research to entire paragraphs or forum posts in a systematic manner. We confirmed the feasibility of Google Maps reviews as a source of labeled data, provided that funding can be secured. We saw that learning on the dataset is possible for classification models, simple and complex alike. Strong results were achieved for easier classification problems, while results attained for more difficult problems such as multi-class classification were palatable as well.

We saw that we were able to achieve a good deal of success with a rule-based data labeling approach to compensate for our lack of labeled training data. The training dataset built up in this manner was useful for multiple models featuring different approaches to learn on. Of course, modifications would have to be made to filter out some unwanted false positives during the labeling process if we were to hone this approach

into something that could achieve state-of-the-art results. But overall, the forward steps made by leveraging this approach showcase the promise for this method.

We explored Spark with BERT and the Bi-LSTM as two state-of-the-art methods to make use of our rule-based manually labeled training data. The Spark with BERT model had some hiccups not necessarily related to the actual performance of the model, but demonstrated great promise with strong validation results. There could definitely be strong results on the horizon if some of the logistical kinks of our model could be ironed out. Meanwhile, the Bi-LSTM model featured an inventive approach to analyzing text-based data but unfortunately fell short in our evaluation of it on our dataset. This method has great potential but our models were unable to showcase any of it during our inconclusive experimentation.

In all, the exploratory analysis in this work covered much breadth and a decent amount of depth. We were able to confirm the feasibility of numerous NLP methods to be applied to our dataset. But as this project was merely a pre-study, there could always be more accomplished and more steps taken.

8.1 Outlook and Future Work

In the context of the overarching project, much remains to be investigated and carried out. The original goal for the project covered in this work involved extracting (location, time, purpose) tuples for analysis on cravings. A significant chunk of our work centered on how to extract this information from text-based data. To take the next step, we must consider how to carry over the insights gained in this work to progress towards such an analysis. This would involve not only extracting entities from text but also gaining an understanding of the context that the entity is part of. Determining a purpose for a person's being at a certain location could give us the context necessary to make a decision on his or her craving state. A time-based analysis would also be useful for gaining insight on which times of the day or times of the week are most likely to see people feeling susceptible to alcoholic cravings.

With regards to our main NER methods, there are several logical next steps to take. The first, for the Google Maps reviews approach, is to get better funding. Primarily, we want to see if we can achieve stronger results on more difficult tasks. We realize that not nearly enough work went down this line because we were hamstrung by our lack of funding, leading to a lack of sufficient training data. Our results on such limited training data show that there is a lot of potential with this method if we expand the size of our training dataset. We would then like to conduct more experimentation regarding the scalability of the learning process, analyzing how the F1-score changes based on the number of entity classes we use and how many training samples we take. Following

on with this line of thinking, we would wonder if this approach could be combined with Deep NLP to obtain even better results. With Google Maps reviews being such a vast and expansive dataset, we may find ourselves able to collect enough noisy data to experiment with state-of-the-art Deep NLP methods. In addition to this, we would also like to consider the feasibility of integrating geolocation data into our study. What about Google’s extensive GPS data? What about the location data collected by smart devices as part of our sister project on physiological data?

For the rule-based labeling approach, the main thing we would like to see as the next step would be to transfer the learning. Transfer learning often requires two steps. The first step can be training the model on a large quantity of noisy data or on a dataset that is outside of the intended target domain. Then the second step would be to fine-tune the model on a smaller set of gold-standard labeled data that is in the target domain. This is a sensible progression following our confirmation of the feasibility of using rule-based manually labeled training data. We could use our rule-based labeled dataset as the noisy data, because there would definitely be mistakes in labeling. Using this, there would be ideas we would want our model to pick up on. For example, will it be able to label synonyms of words that are labeled in the noisy dataset? If not, would the model be able to learn this during the fine-tuning process?

Alternatively, we could also explore training a model on a large dataset in a different domain as the first step of transfer learning. The Google Maps reviews dataset could be used here. In our work, we only trained our model on Google Maps reviews, without ever taking the next step to fine-tune on other data. After training a model that can achieve strong results on the test set, it would make sense to add to it and fine-tune it for the Reddit dataset. Perhaps the Spark with BERT model would be able to step in here as well. BERT is already a massive collection of word embeddings designed for pre-training. Fine-tuning it would be a logical next step.

For the Bi-LSTM model, there remains much to be done due to the inconclusive nature of our results. There is undoubtedly a lot of promise tied to this method, as taking the sequential nature of the data into account can only be a plus. To achieve better learning, we could look into modifying the individual gates within the nodes in order to control the memory and how much context to take in. We could also consider adding or removing layers from the model as we see fit.

There are also a lot of other state-of-the-art NER methods in literature, such as the ones we have documented in this work. If we find them and modify them to work for our task and in our domain, we could achieve even better results than what is outlined in this work.

Finally, we should keep in mind that a mobile application for alcohol addiction prevention and intervention is a final goal for the AlcPrIntTUM project. While this is not a feature we expect to be available in the immediate future, it still makes sense to

proceed with our development with this goal in mind. Our goal is to translate our data analysis into actual results on when users may experience cravings. In doing so, we would be able to warnings or encouragement to users, strengthening their resolve on their path to recovery.

List of Figures

1.1	Co-reference resolution example	7
2.1	AlcPrIntTUM task breakdown	11
3.1	BadgeBot statistics	17
3.2	Word-level clustering data processing pipeline	19
3.3	Word clustering visualization	20
3.4	Word clustering example	20
3.5	Sentence clustering example: spaCy	22
3.6	Sentence clustering example: SentenceBERT	25
3.7	Sentence clustering example: Universal Sentence Encoder	26
3.8	Test set accuracy confidence probabilities	35
3.9	Test set labeling using Doccano	36
4.1	Google Maps API JSON output	38
4.2	Sentence similarity example	40
4.3	Google Maps reviews binary classification data	41
4.4	Google Maps binary classification test example	43
4.5	Google Maps reviews multi-class classification data	44
4.6	Google Maps multi-class classification test example	45
4.7	Reddit test set dataframe	46
4.8	Google Maps binary classification results. Difficulty: easy.	47
4.9	Google Maps binary classification results. Difficulty: hard	48
4.10	Google Maps binary classification results. Difficulty: expert	48
4.11	Google Maps multi-class classification results	49
5.1	Entities labeled by rule-based labeling	55
6.1	Spark with BERT dataframe	63
6.2	Results obtained by Spark with BERT model	64
7.1	Bi-LSTM model architecture	68
7.2	Bi-LSTM model training log	71

List of Tables

1.1	Deaths due to alcohol in 2016	2
1.2	Statistics on alcohol consumption	3
1.3	Statistics on AUD	4
3.1	Reddit API data fields	18
3.2	Test set person entities	30
3.3	Test set location entities	31
3.4	Test set emotions	32
4.1	Multi-class classification model location types	43
5.1	Keywords for automated manual data labeling	54
5.2	Rule-based labeling model evaluation results broken down by document then entity type	57
5.3	Rule-based labeling model evaluation results broken down by entity type across all documents	58
7.1	Bi-LSTM NER model results	70

Bibliography

- [13] *Diagnostic and Statistical Manual of Mental Disorders, Fifth Edition, Text Revision (DSM-5-TRTM)*. American Psychiatric Association, 2013. ISBN: 978-0-89042-576-3.
- [22a] *Alcohol Related Disorders F10-*. 2022. URL: <https://www.icd10data.com/ICD10CM/Codes/F01-F99/F10-F19/F10-> (visited on 10/01/2022).
- [22b] *stopdrinking badge distribution*. 2022. URL: <http://badgebot.me/stopdrinking/> (visited on 10/10/2022).
- [Agg19] R. Aggarwal. *Bi-LSTM*. July 2019. URL: <https://medium.com/@raghavaggarwal0089/bi-lstm-bc3d68da8bd0> (visited on 11/09/2022).
- [alc22] alcoholthinkagain, ed. *Alcohol and the Digestive System*. 2022. URL: <https://alcoholthinkagain.com.au/alcohol-your-health/alcohol-and-long-term-health/alcohol-and-the-digestive-system/> (visited on 10/03/2022).
- [AIR+15] R. Al-Rfou, V. Kulkarni, B. Perozzi, and S. Skiena. "Polyglot-NER: Massive Multilingual Named Entity Recognition." In: *Proceedings of the 2015 SIAM International Conference on Data Mining, Vancouver, British Columbia, Canada, April 30- May 2, 2015* (Apr. 2015).
- [Ame22] American Psychiatric Association, ed. *Diagnostic and Statistical Manual of Mental Disorders (DSM-5-TR)*. 2022. URL: <https://psychiatry.org/psychiatrists/practice/dsm> (visited on 10/08/2022).
- [bae21] baeldung. *Multiclass Classification Using Support Vector Machines*. Aug. 2021. URL: <https://www.baeldung.com/cs/svm-multiclass-classification> (visited on 10/27/2022).
- [Bri20] J. Briggs. *How to Use the Reddit API in Python*. Ed. by Towards Data Science. Dec. 2020. URL: <https://towardsdatascience.com/how-to-use-the-reddit-api-in-python-5e05ddfd1e5c> (visited on 04/16/2022).
- [Bro17] J. Brownlee. *A Gentle Introduction to Long Short-Term Memory Networks by the Experts*. May 2017. URL: <https://machinelearningmastery.com/gentle-introduction-long-short-term-memory-networks-experts/> (visited on 11/09/2022).

- [Car] Caron Staff, ed. *Why Is Alcohol So Addictive?* URL: <https://www.caron.org/blog/why-is-alcohol-so-addictive> (visited on 10/04/2022).
- [Cen20] Centers for Disease Control and Prevention, ed. *Impaired Driving: Get the Facts*. 2020. URL: https://www.cdc.gov/transportationsafety/impaired-driving/impaired-drv_factsheet.html (visited on 10/02/2022).
- [Cer+18] D. Cer, Y. Yang, S. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y. Sung, B. Strope, and R. Kurzweil. "Universal Sentence Encoder." In: *CoRR* abs/1803.11175 (2018). arXiv: 1803.11175.
- [Dee21] Deekshithajp. *BERT for Text Classification*. Ed. by Medium. Dec. 2021. URL: <https://medium.com/@deekshithajp/bert-for-text-classification-b88f93f1685e> (visited on 06/17/2022).
- [Dri22] Drinkaware, ed. *How to prevent alcohol-related accidents*. 2022. URL: <https://www.drinkaware.co.uk/advice/staying-safe-while-drinking/how-to-prevent-alcohol-related-accidents> (visited on 10/02/2022).
- [Duc+13] G. Duck, G. Nenadic, A. Brass, D. Robertson, and R. Stevens. "BioNerDS: Exploring bioinformatics' database and software use through literature mining." In: *BMC bioinformatics* 14 (June 2013), p. 194. DOI: 10.1186/1471-2105-14-194.
- [EA08] P. Edmonds and E. Agirre. *Word Sense Disambiguation*. Ed. by D. R. Mihalcea. May 2008. URL: http://www.scholarpedia.org/article/Word_sense_disambiguation (visited on 10/15/2022).
- [EKK17] T. Eftimov, B. Koroušić Seljak, and P. Korošec. "A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations." In: *PLOS ONE* 12.6 (2017). DOI: 10.1371/journal.pone.0179488.
- [Fan+21] Z. Fang, Y. Cao, T. Li, R. Jia, F. Fang, Y. Shang, and Y. Lu. "TEBNER: Domain Specific Named Entity Recognition with Type Expanded Boundary-aware Network." In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 198–207. DOI: 10.18653/v1/2021.emnlp-main.18.
- [fig16] figshare, ed. *bioNerDS v2 dictionary breakdown*. 2016. URL: https://figshare.com/articles/dataset/bioNerDS_v2_dictionary_breakdown_/3927177/1 (visited on 07/18/2022).

- [Gal16] A. Galea. *American City to State Python Dictionary*. Mar. 2016. URL: <https://galeascience.wordpress.com/2016/03/23/us-city-to-state-python-dictionary/> (visited on 05/12/2022).
- [GAT] GATE, ed. *Named Entity Recognition with ANNIE*. URL: <http://services.gate.ac.uk/annie/> (visited on 05/19/2022).
- [Git] Github, ed. *flairNLP*. URL: <https://github.com/flairNLP/flair> (visited on 05/15/2022).
- [Gom22] R. Gomez. *GeonamesCache*. 2022. URL: <https://pypi.org/project/geonamescache/> (visited on 05/21/2022).
- [Gooa] Google, ed. *universal-sentence-encoder*. URL: <https://tfhub.dev/google/universal-sentence-encoder/4> (visited on 05/02/2022).
- [Goob] Google Maps Platform, ed. *Place Details*. URL: <https://developers.google.com/maps/documentation/places/web-service/details> (visited on 05/23/2022).
- [Goo22] Google Maps Platform, ed. *Place Types*. 2022. URL: https://developers.google.com/maps/documentation/places/web-service/supported_types (visited on 05/20/2022).
- [Gup18a] M. Gupta. *A Review of Named Entity Recognition (NER) Using Automatic Summarization of Resumes*. Ed. by Towards Data Science. July 2018. URL: <https://towardsdatascience.com/a-review-of-named-entity-recognition-ner-using-automatic-summarization-of-resumes-5248a75de175> (visited on 07/16/2022).
- [Gup18b] S. Gupta. *Named Entity Recognition: Applications and Use Cases*. Ed. by Towards Data Science. Feb. 2018. URL: <https://towardsdatascience.com/named-entity-recognition-applications-and-use-cases-acdbf57d595e> (visited on 10/26/2022).
- [Haz20] Hazelden Betty Ford Foundation, ed. *The Facts about Alcohol Cravings and How to Beat Them*. Aug. 2020. URL: <https://www.hazeldenbettyford.org/articles/what-is-craving> (visited on 10/08/2022).
- [Huga] Hugging Face, ed. *bert-base-NER*. URL: <https://huggingface.co/dslim/bert-base-NER> (visited on 05/17/2022).
- [Hugb] Hugging Face, ed. *ner-english-ontonotes-large*. URL: <https://huggingface.co/flair/ner-english-ontonotes-large> (visited on 10/25/2022).
- [IBM20] IBM Cloud Education. *Natural Language Processing (NLP)*. Ed. by IBM. July 2020. URL: <https://www.ibm.com/cloud/learn/natural-language-processing> (visited on 10/14/2022).

- [Joh22a] John Snow Labs, ed. *NerDLApproach*. 2022. URL: <https://nlp.johnsnowlabs.com/api/com/johnsnowlabs/nlp/annotators/ner/dl/NerDLApproach> (visited on 11/06/2022).
- [Joh22b] John Snow Labs, ed. *Spark NLP: State of the Art Natural Language Processing*. 2022. URL: <https://nlp.johnsnowlabs.com/> (visited on 11/06/2022).
- [Joh22c] John Snow Labs, ed. *Tensorflow Graph*. 2022. URL: <https://nlp.johnsnowlabs.com/docs/en/graph> (visited on 09/26/2022).
- [Kan+16] S. Kannan, S. Karuppusamy, A. Nedunchezian, P. Venkateshan, P. Wang, N. Bojja, and A. Kejariwal. "Chapter 3 - Big Data Analytics for Social Media." In: *Big Data*. Ed. by R. Buyya, R. N. Calheiros, and A. V. Dastjerdi. Morgan Kaufmann, 2016, pp. 63–94. ISBN: 978-0-12-805394-2. DOI: <https://doi.org/10.1016/B978-0-12-805394-2.00003-9>.
- [Kar22] D. F. Karabiber. *Binary Classification*. Ed. by B. Martin and R. Lewis. 2022. URL: <https://www.learndatasci.com/glossary/binary-classification/> (visited on 05/13/2022).
- [Kim+21] J. Kim, Y. Kim, S. Kang, and J. Seo. "Adaptive Named Entity Recognition Using Distant Supervision for Contemporary Written Texts." In: *IEEE Access* 9 (2021), pp. 80405–80414. DOI: 10.1109/access.2021.3067315.
- [Koc19] V. Kocaman. *Introduction to Spark NLP: Foundations and Basic Components*. Ed. by Towards Data Science. Sept. 2019. URL: <https://towardsdatascience.com/introduction-to-spark-nlp-foundations-and-basic-components-part-i-c83b7629ed59> (visited on 11/06/2022).
- [Koc20] V. Kocaman. *Named Entity Recognition (NER) with BERT in Spark NLP*. Ed. by Towards Data Science. Mar. 2020. URL: <https://towardsdatascience.com/named-entity-recognition-ner-with-bert-in-spark-nlp-874df20d1d77> (visited on 05/06/2022).
- [Kor21] P. Korab. *Clustering Textual Data with Word2Vec*. Ed. by Python in Plain English. Dec. 2021. URL: <https://python.plainenglish.io/clustering-textual-data-with-word2vec-866dafbd213f> (visited on 04/24/2022).
- [Kra22] T. Kramer. "Simulating and Predicting Addiction Craving Situations through Text Mining and NLP." MA thesis. 2022.
- [Mar19] C. Marshall. *What is named entity recognition (NER) and how can I use it?* Ed. by super.AI. Dec. 2019. URL: <https://medium.com/mysuperaai/what-is-named-entity-recognition-ner-and-how-can-i-use-it-2b68cf6f545d> (visited on 10/17/2022).

- [McC19] C. McCrackin. *New Research Exposes the 15 Most Dangerous Drugs*. Ed. by Addiction Center. 2019. URL: <https://www.addictioncenter.com/news/2019/08/15-most-dangerous-drugs/> (visited on 10/02/2022).
- [MM22] M. Maślankowska and P. Mielniczuk. *Intro to Coreference Resolution in NLP*. Ed. by Neurosys. May 2022. URL: <https://neurosys.com/blog/intro-to-coreference-resolution-in-nlp> (visited on 10/15/2022).
- [Mov22] Movendi International, ed. *Switzerland: Alcohol Has Never Been More Unpopular in Recent History*. 2022. URL: <https://movendi.ngo/news/2022/08/02/switzerland-alcohol-has-never-been-more-unpopular-in-recent-history/> (visited on 10/05/2022).
- [MS20] V. Mikhailov and T. Shavrina. “Domain-Transferable Method for Named Entity Recognition Task.” In: *CoRR* abs/2011.12170 (2020). arXiv: 2011.12170.
- [Nai20] S. Nair. *Named-Entity Recognition (NER) using Keras Bidirectional LSTM*. June 2020. URL: <https://towardsdatascience.com/named-entity-recognition-ner-using-keras-bidirectional-lstm-28cd3f301f54> (visited on 09/09/2022).
- [Nat20a] National Highway Traffic Safety Administration, ed. *Drunk Driving*. 2020. URL: <https://www.nhtsa.gov/risky-driving/drunk-driving> (visited on 10/02/2022).
- [Nat20b] National Institute on Alcohol Abuse and Alcoholism, ed. *Understanding Alcohol Use Disorder*. 2020. URL: <https://www.niaaa.nih.gov/publications/brochures-and-fact-sheets/understanding-alcohol-use-disorder> (visited on 10/04/2022).
- [Pap] Papers with Code, ed. *CoNLL-2003*. URL: <https://paperswithcode.com/dataset/conll-2003> (visited on 10/25/2022).
- [Ped+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. “Scikit-learn: Machine Learning in Python.” In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- [PMD21] K. Pohl, P. Moodley, and A. D. Dhanda. “Alcohol’s Impact on the Gut and Liver.” In: *Nutrients* 13.9 (2021). DOI: 10.3390/nu13093170.
- [PR18] V. Poznyak and D. Rekve. *Global Status Report on Alcohol and Health 2018*. World Health Organization, 2018.

- [Pre22] M. Preston. *Top 10 Most Dangerous Drugs*. Ed. by The Delamere Blog. 2022. URL: <https://delamere.com/blog/top-10-most-dangerous-drugs> (visited on 10/02/2022).
- [Ray21] C. Raypole. *9 Ways to Manage Alcohol Cravings*. Ed. by K. Martinez. Dec. 2021. URL: <https://www.healthline.com/health/alcohol-cravings> (visited on 10/08/2022).
- [Red22] Reddit, ed. *API Documentation*. 2022. URL: <https://www.reddit.com/dev/api/> (visited on 04/16/2022).
- [Rei96] E. Reiter. "Building Natural-Language Generation Systems." In: *CoRR comp-lg/9605002* (1996).
- [Ret] Rethinking Drinking: Alcohol Your Health, ed. *Handling Urges to Drink*. URL: <https://www.rethinkingdrinking.niaaa.nih.gov/tools/Interactive-worksheets-and-more/Stay-in-control/Coping-With-Urges-To-drink.aspx> (visited on 10/08/2022).
- [RG19] N. Reimers and I. Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks." In: *CoRR abs/1908.10084* (2019). arXiv: 1908.10084.
- [sci22a] scikit-learn developers, ed. *Recognizing Hand-written Digits*. 2022. URL: https://scikit-learn.org/stable/auto_examples/classification/plot_digits_classification.html (visited on 05/29/2022).
- [sci22b] scikit-learn developers, ed. *sklearn.cluster.DBSCAN*. 2022. URL: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html> (visited on 04/29/2022).
- [sci22c] scikit-learn developers, ed. *sklearn.svm.SVC*. 2022. URL: <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html> (visited on 05/29/2022).
- [spa] spaCy, ed. *English: Available trained pipelines for English*. URL: <https://spacy.io/models/en> (visited on 05/18/2022).
- [Stu] N. Stuben. *Ohne Alkohol mit Nathalie*. URL: <https://oamn.jetzt/> (visited on 04/15/2022).
- [Tec21] Technische Universität München, ed. *Cloud Computing (IN2107)*. 2021. URL: <https://www.ce.cit.tum.de/caps/lehre/ws21/seminare/cloud-computing/> (visited on 10/29/2022).
- [Tec22] Technische Universität München, ed. *Research Group Social Computing*. 2022. URL: <https://www.soc.cit.tum.de/> (visited on 04/15/2022).

- [The] The Stanford Natural Language Processing Group, ed. *Coreference Resolution*. URL: <https://nlp.stanford.edu/projects/coref.shtml> (visited on 10/15/2022).
- [The20] The Stanford Natural Language Processing Group, ed. *Stanford Named Entity Recognizer (NER)*. 2020. URL: <https://nlp.stanford.edu/software/CRF-NER.html> (visited on 05/13/2022).
- [The22] The Imbalanced-Learn Developers, ed. *SMOTE*. 2022. URL: https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html (visited on 07/29/2022).
- [Ver21] Y. Verma. *Complete Guide To Bidirectional LSTM (With Python Codes)*. July 2021. URL: <https://analyticsindiamag.com/complete-guide-to-bidirectional-lstm-with-python-codes/> (visited on 11/09/2022).
- [Wig21] I. Wigmore. *Natural Language Generation (NLG)*. Ed. by TechTarget. July 2021. URL: <https://www.techtarget.com/searchenterpriseai/definition/natural-language-generation-NLG> (visited on 10/20/2022).
- [Wik21] Wikipedia, ed. *Wikipedia:Contents/Categories*. Apr. 2021. URL: <https://en.wikipedia.org/wiki/Wikipedia:Contents/Categories> (visited on 10/25/2022).
- [Wik22a] Wikipedia, ed. *Hoeffding's Inequality*. 2022. URL: https://en.wikipedia.org/wiki/Hoeffding%27s_inequality (visited on 10/26/2022).
- [Wik22b] Wikipedia, ed. *Natural Language Processing*. 2022. URL: https://en.wikipedia.org/wiki/Natural_language_processing (visited on 10/14/2022).
- [Wor] World Health Organization, ed. *Global Information System on Alcohol and Health*. URL: <https://www.who.int/data/gho/data/themes/global-information-system-on-alcohol-and-health> (visited on 10/03/2022).