LONDON'S GLOBAL UNIVERSITY

UCL

# Using natural language processing to develop a pipeline to analyse media representation of people with disabilities in Web-based news articles

Collection and filtering of Web-based news articles, comparison of open-source sentiment models, and applications of the technology

Bagus Maulana[1]

MEng Computer Science

Catherine Holloway, Nicholas Firth

Submission date: 30th April 2018

[1]**Disclaimer:** This report is submitted as part requirement for the MEng Degree in Computer Science at UCL. It is substantially the result of my own work except where explicitly indicated in the text. *The report may be freely copied and distributed provided the source is explicitly acknowledged*

**Abstract**

**Report Title:** Using natural language processing to develop a pipeline to analyse media representation of people with disabilities in Web-based news articles: Collection and filtering of Web-based news articles, comparison of open-source sentiment models, and applications of the technology

**Authors Name:** Bagus Maulana

**Supervisors Name:** Catherine Holloway, Nicholas Firth

**Date and Year of Submission:** 30$^{th}$ April 2018

# Contents

# Chapter 1

# Introduction

Natural language processing (NLP) encompasses a wide range of computational techniques for machine understanding of human (natural) language that are often used alongside each other. The review article [1] defines natural language processing as 'a theory-motivated range of computational techniques for the automatic analysis and representation of human language.' The techniques that fall under the umbrella natural language processing include word tokenisation, probabilistic language modelling, translation, part-of-speech parsing, sentiment analysis (or opinion mining), text classification/categorisation, and topic modelling, among other things. The computational models used in natural language processing range from simple rule-based models (e.g. splitting a sentence on whitespace to tokenise words) to statistical machine learning and deep learning models. Natural language processing is now applied for various everyday technologies, for example, information retrieval for search engines such as Google and Bing, and categorisation and topic modelling for recommendation engines used to suggest 'similar' articles.

The obvious advantage of natural language processing is that machines can process vast bodies of human-created literature (books, articles, posts, e-mails, messages, etc.) much faster than humans can, processing thousands or millions of documents per second. This allows for high-level quantitative analyses of all documents in a vast corpora for a given domain to be feasible, which can uncover information previously inaccessible by only reading and generalising from a small sample of documents. For example, this level of quantitative analysis can uncover trends and patterns within a given domain (e.g. how does the popularity of the term 'mentally ill' increase or decrease year-on-year in English news media?).

Various other studies have attempted to utilise natural language processing to perform high-level analyses in the domain of news media. The research done in [2] assembled a vast corpus of regional newspapers in the United Kingdom spanning 150 years to detect long-term patterns of cultural change (e.g. increase of female representation in the news, or when trains overtook horses for transportation) by analysing n-gram trends and named entities. More specifically, in the domain of media representation of specified groups of people, studies such as [3] has attempted to use features based on natural language processing (such as n-grams and part-of-speech tags) to classify racist and sexist posts in social media, although there is still a research gap in this area (especially for news articles, and/or relating to specific groups, such as people with (specific) disabilities or mental illnesses).

Applying natural language processing to perform meta-analyses over large text corpora has various interesting potential applications in improving our understanding of the human world - for example, to detect macroscopic cultural shifts as in [2]. In particular, the representation of specific groups, such as people with disabilities, has been a popular research theme for psychologists, sociologists, and others. For example, the paper [4] analysed a sample of 600 print articles relating to mental illnesses in New Zealand and categorised them to positive and negative depictions, and the predominant themes thereof (e.g. criminality (negative), educational accomplishments (positive)). Applying natural language processing to this area of research would allow the possibility of discovering higher-level trends, by computationally analysing a much larger sample of articles and identify trends by varying independent variables such as year of publication and publisher. In this research, a sample of 305,113 news articles (51,177 after filtering off-topic articles) from British online news sources are used. However, challenges remain as syntax-based statistical natural language processing approaches tend to be more limited in scope and is prone to false positives and negatives, and mitigating these factors is currently an open area of research.

The aim of this project is to utilise these natural language processing computational techniques in order to perform a high-level meta-analysis of literature available in the public news media available online. More specifically, to gather online news articles relating to people with disabilities in British media, and perform natural language processing analyses at scale to identify trends such as term popularity (e.g. 'suffer from ...' vs 'with ...') and variation in positive/negative sentiment.

The goal of this project is to develop a computational pipeline capable of performing this analysis of online media end-to-end. Given a list of topic consisting of keywords (or phrases) and query terms, this pipeline covers the task of web crawling and scraping, using public APIs if possible, to collect a dataset of news articles; filtering off-topic articles for the given keywords; matching relevant sentences referring to a keyword; performing sentiment analysis (using publicly available open-source libraries) on these sentences; and producing relevant plots to show applications of this technology. This pipeline will be available open source on GitHub (`https://github.com/bmaulana/nlp-media`)

This project was carried out in a step-by-step approach. The pipeline was developed as four individual components: a web scraper and crawler for data collection given a list of queries, a filter to remove irrelevant articles given a list of key terms, a parser to pattern-match relevant sentences given key terms, a sentiment scorer (and results analysis/comparison of different open-source scorers on this domain), and a script to perform statistical analysis on the results and produce relevant plots. Each component's output is piped to the next component's input by saving its output to a JSON file and having the next component read the previous component's output file, which ensures computation can be 'resumed' without re-running the previous component. A main pipeline script connects these components together by calling them in order for each topic and supported Web source (Daily Mail, Daily Express, Guardian).

The body of this report is subdivided into four sections: context, requirements and analysis, design and implementation, and results evaluation.

# Chapter 2

# Context

## 2.1 Background

Analyses of news media in its various forms (print, online, etc.) has been a consistent research theme. The news media provides a quantifiable depiction of the prevailing society's popular conceptions or views regarding a theme or topic. This is especially applicable with regards to conceptions on particular groups of people, in which the language used in the media reflect on popular views, and has been shown to differ (with statistical significance) in different societies. For example, it was shown that the Canadian press was more likely to name individuals with disability and use appropriate labelling than the Israeli press in 1998 [5]. Furthermore, there is evidence to suggest that news media sources contribute to shape and reinforce beliefs among the society, such as misconceptions and stigma [6].

Public awareness of disability is also a popular research theme. While not related to news media, a review in 2011 [7] found 75 articles and 68 studies that passed a selective inclusion criteria with regards to intellectual disabilities, published in English between 1990 and mid-2011. The topics brought up include the public's knowledge, attitudes, and beliefs about intellectual disability; and varying for socio-demographic characteristics, cross-cultural comparisons, and the effects of interventions.

Analyses of news or other media are primarily carried out by taking a small, statistically representative sample of documents (news articles) from a text corpora (for example, all news articles published in England for a certain period) and analysing them manually. There has also been various such studies within the domain of disability awareness in the media:

- In a 2002 study [4], researchers analysed a sample of 600 print articles relating to mental health or mental illness that was collected by a commercial clipping bureau. The articles were then categorised into positive and negative depictions, then further into sub-samples such as danger to others, criminality, vulnerability, etc. The study found that at the time, in New Zealand, negative themes predominate about 3 to 1 (with 27% being positive). However, given the paper's scope, this conclusion cannot be generalised to learn trends, or how the conclusion varies given certain variables (e.g. time, location).
- A study conducted in 1998 [5] were replicated in 2008 [8] to assess change in representations of disability and persons with disability in the Canadian news media. This study sampled

196 news articles in 1998 and 166 news articles in 2008. It found an increase in the usage of 'person-first' terminology (e.g. person with disabilities) and a decrease in 'disabling language' (e.g. disabled person). This is an attempt to identify trends with regards to media representation of disability, however only provides two data points (1998 and 2008) with relatively small sample size.

- A study in 2005 [9] analysed 1,515 articles relating to autism in Australian news media. All articles were read by two research assistants to ensure they are on-topic and then coded as either 'negative' or 'positive' in overall focus, and then coded into themes (e.g. funding, education, etc.).

By applying natural language processing and computational techniques in analysing text articles, it is possible to develop a computational pipeline that could analyse and extract quantitative information from these articles at a much faster rate, enabling the analyses of a much larger scale of documents within a realistic time frame. While a sample of few hundred or thousand documents is usually enough to provide statistically significant conclusions, by providing an analysis of the full corpora (or a much larger sample), it is possible to uncover additional information from the data set. For example, higher-level trends (such as how the conclusion varies by year, location, publisher, etc.) can be discovered from a quantitative analysis of the larger dataset, by 'splitting' the result set into smaller subsets based on independent variables (e.g. year, location, publisher, etc.) and performing statistical comparisons of dependent variables (e.g. term frequency) between each subset.

Additionally, data collection is much more feasible in scale, cost and time using computational techniques. An automated script can be used to collect news articles published on the Internet at a rate of roughly one article per second, or thousands of articles per hour (varies on Web source, hardware, Internet connection, etc.), a vast improvement over contracting a commercial clipping bureau to provide 600 articles as in [4].

Several studies has taken advantage of this approach to carry a more complete analysis of textual corpora. For example, [2] assembled a corpus of 35.9 million news articles from 120 publishers in the United Kingdom between 1800 and 1950, representing 14% of all news articles published in the United Kingdom over that period. With this approach, the researchers were able to extract quantitative time-series information with regards to cultural trends as present in 35.9 million British news articles over the 150-year period. This amount of large time-series information (represented as n-grams and named entities) allowed them to discover macroscopic cultural trends. By analysing and comparing word (n-gram) trends across various topics, the researchers were able to identify trends that reflect cultural shifts e.g. 'train' increasing in popularity and overtaking 'horse' around 1900, or 'labour party' overtaking 'conservative party' and 'liberal party' in news coverage from the 1920s onward. Additionally, they also used entity recognition to extract named entities from articles and considered trends based on known information about these named entities, such as the proportion of female vs male entities, categories of entities (e.g. politicians, writers, etc.), and age of these entities. They also considered the geographical location of the publication to see how word usage trends (e.g. 'british' vs 'english') differ based on location.

[2]'s study was based on prior discussions and studies on the potential of exploiting large text corpora to detect macroscopic, long-term cultural changes. [10] was one of the first studies to suggest this approach. In this seminal study, a corpus of 5 million digitised English-language books

published over 200 years (or about 4% of all books ever published), provided by Google's effort to digitise books, were analysed to extract how often a given n-gram was used over time. (This data is available on `http://www.culturomics.org/`) This information is then used to analyse trends in language: the size of the English lexicon, regularisation of English verbs (from 'irregular' suffixes to '-ed'), or how quickly years (e.g. '1950') decline in use. Influenced by [10], several other studies has been published adopting a similar approach:

- an analysis of 1.7 million Victorian-era books [11]
- an analysis of 17,094 US Billboard Hot 100 songs between 1960 and 2010 [12]
- an analysis of a 3.9 million news article sample from the Summary of World Broadcasts (SWB) collection [13]
- an analysis of 2.5 million English-language million news articles from 498 online news outlets from 99 countries [14]

However, there were also criticism of this approach, such that it ignored semantics and context, thus for example 'thirteen hundred words of gibberish and the Declaration of Independence are digitally equivalent' [15], issues with OCR quality and duplicate editions [15], or that the selection of digitised books are biased [16].

There has been some progress in applying natural language processing techniques more specifically in the domain of how (specific) groups of people are represented in the media:

- A study has attempted to extract features (such as n-grams and part-of-speech tags) using natural language processing to classify racist and sexist posts in social media, providing annotations to 6,909 tweets [3].
- Another study explored potential linguistic markers of schizophrenia in social media from a dataset of 174 users with self-reported schizophrenia and up to 3,200 tweets per user, and control users in a 50:50 split. They used a support vector machine (SVM) approach over features based on lexicon-based approaches (i.e. a list of mental health related keywords), latent dirichlet allocation (LDA), Brown clustering, character n-grams, and perplexity [17].

However, there is still a visible research gap in this area, for using computational natural language processing specifically for news articles and/or with regards to specific groups, such as people with disabilities (or a specific disability).

To date, advances in natural language processing has made this approach much more feasible, efficient, and effective, even given limited time and hardware constraints. Various free and open-source tools for computational natural language processing and statistical analysis has been developed by the research community, such as nltk [18] and StanfordNLP [19] for natural language processing, scikit-learn [20] for statistical analysis, and matplotlib [21] for plotting. Depending on the type of data analytics performed and the hardware used, these computational tools are able to analyse news articles at a rate of multiple documents per second. The 'Technical Context' section contains a further listing and discussion of these tools, alongside the specific libraries and natural language processing techniques used for this research project.

## 2.2 Research Methodology and Sources

Background research were carried out by investigating papers from public sources (such as Google Scholar). Research articles were gathered from a list of important topics and query terms related

to the technique (natural language processing) and/or the domain (disability and news media): for example, 'natural language processing review', 'news media' AND 'disability', 'natural language processing' AND ('news media' OR 'cultural trends'), and 'natural language processing' AND 'disability'. Highly-cited research articles are prioritised as examples, as they are deemed to be more 'important' papers or studies within its topic. Additionally, the author looked for highly-cited 'key' papers and review articles within each specific topic, and then looked at its list of citations (older papers), and research articles that cites the review article (newer papers), to expand the list of relevant research papers and examples.

Technical research, on the other hand, were carried out as necessary. After the requirements and components for the pipeline has been decided, research were carried out to find relevant techniques, formulae, algorithms, tools, libraries/packages, and existing implementations that would be useful to implement each component (or sub-tasks within a component). The research were carried out in an iterative approach alongside software development, where initial planning and research would show an initial implementation plan, then implementation would uncover feasibility of these approaches and possible alternatives/refinements to be researched, then further research may reveal new options/refinements to be implemented, and so on. Research or work done on other components/tasks may also reveal possible improvements or alternatives for another task, which may require further research and implementation. Again, more popular tools and libraries are prioritised, although several approaches and implementations were considered for most components and sub-tasks, to be compared for suitability, runtime, results, etc.

The sources used for this literature review are:

- Google Scholar, often used to find research articles to act as 'entry points' towards a research topic, to find other studies similar to another research article within a specified topic, or to retrieve citation information of a given research paper, book, or popular Python package.
- GitHub topics, used to find repositories that are relevant to a specific task or component, find similar GitHub repositories, gather information about a given repository, and also as a benchmark for topic popularity and range of solutions in a given programming language. Several GitHub pages also curate a list of repositories within a specific topic [22]–[24].
- Python package repositories such as PyPi [25] and Anaconda Cloud [26], which list all available Python packages and shows general information regarding them, and also provides a search function; useful to find relevant packages for a component/task and gather information about a given Python package.
- Official web sites and documentation of Python packages, which list and define capabilities (functions and parameters) of the package, useful to explore the functionality of a given package and its capacity to solve a specific task, and to understand the technologies/approaches used by the package's implementation (e.g. how is a sentiment model implemented?). Often (especially with more popular packages), citation information regarding the package would also be available in its web site.

## 2.3 Technical Context

As mentioned earlier, computational natural language processing techniques are utilised to extract features from collected news articles (text documents) at scale. For this project, the requirements

for the computational pipeline can be subdivided to five main components: data collection, filtering, sentence matching, sentiment scoring, and statistical analysis and plotting. Among these components, natural language processing techniques are necessary for filtering, sentence matching, and sentiment scoring. On the other hand, collection is performed using established, general-purpose web-scraping tools. Similarly, statistical analysis and plotting is performed using general-purpose statistical tools and metrics. This section will cover the technical tools researched and used for all listed components regardless.

Natural language processing covers three main 'curves' or areas: syntax, semantics, and pragmatics (narratives, understanding). Syntax specifies the way symbols (words, terms, tokens, or n-grams) and groups of symbols are arranged and whether they are well-formed in an expression, whereas semantics specifies what these expressions mean, and pragmatics specifies contextual information [1]. Contemporary (or 'traditional') approaches to natural language processing mainly focus on syntactic analysis, due to the relative ease of extracting syntactic features of text such as term frequency, word co-occurrence, and part-of-speech tags, compared to extracting logical expressions and networks necessary for semantic analysis. However, syntactic analysis is much more limited as it often misses information such as the (semantic) context of a word (e.g. "one" in "there's no one there" (referring to a person) vs "we have only one car" (referring to a quantity)). This paper will focus on mainly syntactic techniques and features, as these are more relevant to this domain of high-level topic matching and sentiment analysis that is feasible with current technology at this scale.

Python was chosen as the main programming language used for this project. The primary reason for this choice is the wide availability and range of existing tools for natural language processing, sentiment analysis, statistical analysis and plotting, web scraping and parsing, etc. in Python. A study in 2016 showed that Python is the most popular language for machine learning and data science [27], which correlates to the amount of available tools developers have created for the language. A GitHub search for the topic 'nlp' as of 18 April 2018 reported 1,397 Python and 470 Jupyter (Python interactive 'notebook') repositories with the tag 'natural language processing', compared to the second most popular programming language being Java with only 251 repositories tagged 'nlp' [28]. Furthermore, Python is also an ideal language for quick experimentation due to relatively high-level and low verbosity of the code, such that it is relatively easier to make small changes on the fly. Additionally, the Anaconda distribution of Python [29] is used for its suitability to set up and manage Python environments and packages for data science projects.

For data collection, general-purpose tools for sending HTTP requests (to 'open' web addresses and store HTML web pages programmatically) and parsing HTML code (to parse article text and metadata from 'raw' HTML code) are sufficient. The Requests library [30] is a popular Python tool (with 400,000+ daily downloads) for sending HTTP/1.1 requests simply. A HTTP GET request will retrieve the HTML code (and other information) associated with a given URL from a web server, similar to opening the page on a web browser, stored as a Python object by Requests. Once the HTML code of a web page (given an article's URL) has been stored, the BeautifulSoup library [31] provides simple methods to navigate and search a parse tree (such as HTML code). Given that web pages from the same source/publisher tend to follow a similar structure, BeautifulSoup can be used to parse article text and relevant metadata (e.g. headline, date of publication, outgoing links in a search page) by searching for specific tags and attributes within the HTML code.

In this project, filtering off-topic articles is done somewhat crudely via ranking term frequency independently for each document. Term frequency is a simple and commonly-used metric for natural language processing, with various existing tools that can compute this metric for thousands of text documents within seconds. To put simply, the term frequency of a term (i.e. a word or token) in a document is the number of times that the term occurs in the document. The popular scikit-learn library [20] provides a tool to measure term frequency of text documents, handling both tokenisation (converting a text document into a list of tokens (terms/words)) and counting word occurrence. It also provides the option to ignore stop words (i.e. common, meaningless words in English, such as 'the' or 'a'), which can often skew the results due to its relative high frequency. Additionally, the nltk library [18] is used for stemming – to reduce all words in the document to its word stem (such that e.g. 'walk', 'walks', 'walked', and 'walking' are equivalent).

Term-frequency ranking is implemented by sorting all terms (words) in the document based on their term frequency. For filtering documents, a constant threshold value is used to determine whether documents are on-topic or not, where a document is on-topic if the keyword's rank is lower than the threshold value (i.e. the keyword is one of the most frequently used terms in the document), and is off-topic if the keyword's rank is higher than the threshold value (i.e. the keyword is rarely used in the document, relative to other terms). As multiple keywords per topic are used, all keywords in the document are first converted to a unique 'keyword token' before term frequency is measured (which will treat all instances of any keyword in the topic as a single term).

In literature related to natural language processing, various more sophisticated approaches has been proposed and used for the task of text classification and filtering off-topic articles. A conventional approach is by calculating term frequency – inverse document frequency (tf-idf) [32], [33], a metric that builds on term frequency by taking into account the relative importance of each word. Inverse document frequency (idf) is calculated by counting the number of documents in a corpus where a term appears – if a term appears more frequently (e.g. common words such as 'the'), it is deemed to be less important and assigned a lower score. However, this approach is unsuitable for this project, given the selective nature of the dataset (where only articles with certain query terms are collected), thus the idf values of these query terms would be flawed (as they would appear in every document).

Another proposed approach is by using a supervised machine learning model to classify documents into pre-defined categories (the text classification problem). Various approaches were proposed to solve text classification, including Support Vector Matrices (SVM), Naïve Bayes (NB), and k-nearest neighbour (kNN) models [34]. However, this approach is infeasible for this project due to a lack of labelled data of articles and categories, and in this case the categories themselves are poorly defined and non-exclusive (multiple different topics may be discussed within one article). Topic models, which compute the proportion of abstract 'topics' in a document, has also been proposed. Latent dirichlet allocation (LDA) [35] represents documents as random mixtures over latent topics, where each topic is characterized by a distribution over words. However, as these topics are abstract and characterized generatively (i.e. each topic's distribution over words are generated by the model, rather than pre-defined), it is not very useful for the task of classifying whether a document matches pre-defined topics/keywords. Also, both of these approaches are significantly more computationally expensive than tf ranking or tf-idf.

Sentiment scoring of articles is sub-divided into two components: a component to find sentences

relevant to a topic (given a list of key terms) in a text document (sentence matching), and another component that performs sentiment analysis on these sentences and transforms it into a real-valued score of each sentence. SpaCy [36] is a popular tool for general natural-language processing tasks, using pre-trained convolutional neural network models for tasks such as tagging, parsing, and entity recognition, and is benchmarked to be the fastest and among the most accurate syntactic parser, able to parse 13,965 words per second in 2015 [37]. Among the information SpaCy extracts from text are lemmas (root words) of terms (e.g. 'mentally' $\rightarrow$ 'mental') and features a rule-based matching engine (retrieve a list of sequences of tokens within a document that matches a given pattern, e.g. tokens with a specified lemma), both which are useful for the task of finding sentences relevant to a topic in a document.

For sentiment scoring of sentences, a variety of open-source tools and pre-trained models dedicated to sentiment analysis were researched for the purpose of comparison. Sentiment analysis, or opinion mining, is defined as "the field of study that analyzes people's opinions, sentiments, evaluations, attitudes, and emotions" from natural (human) language [38]. A GitHub search for the topic 'sentiment-analysis' as of 18 April 2018 reported 450 Python and 222 Jupyter repositories with the tag 'sentiment-analysis' [39]. Furthermore, a community-curated list of sentiment analysis methods and implementations exist [22], which served as a useful starting point to explore open-source sentiment analysis implementations.

The particular sentiment analysis implementations that have been explored in this paper are:

- VADER is a parsimonious rule-based model that scores the sentiment of a given sentence, based on the presence of 'sentiment lexicons' (a list of lexical features common to sentiment expression, such as 'good' and 'bad', including slang words, emoticons, and acronyms), and rules such as negations (e.g. 'not good') and increased sentiment intensity given emphasis such as punctuation, capitalisation, and degree modifiers (e.g. 'very') [40].
- vivekn's 'sentiment' repository implements a supervised machine learning approach to sentiment classification trained on an enhanced Naïve Bayes classifier, trained on a publicly available dataset of 25,000 highly-polar movie reviews from the Internet Movie Database (IMDb) using bigrams and trigrams as features [41].
- xiaohan2012's 'twitter-sent-dnn' repository trains a convolutional neural network model with dynamic k-max pooling (DCNN) for modelling (real-valued sentiment scores of) sentences. The sentence model properties considered are the word and n-gram order, and induced feature graph (generated by the DCNN). It was trained on a dataset of 1.6 million tweets with emoticon-based labels. [42]
- kevincobain2000's 'sentiment_classifier' repository trains a supervised machine learning model based on a Naïve Bayes and Maximum Entropy Classifier to transform a sentence to positive and negative (real-valued) sentiment scores. It considers Word Sense Disambiguation using wordnet and word occurrence statistics from nltk's movie review corpus, and uses bigrams as features. The training data is a mixture of nltk's movie review corpus, Twitter posts, and Amazon customer reviews data. [43]
- OpenAI's 'generating-reviews-discovering-sentiment' repository provides a pre-trained single-layer multiplicative LSTM recurrent neural network model with 4096 units (a relatively simple model optimised for training/convergence time) to generate (real-valued) sentiment scores of input sentences. It was trained on a dataset of over 82 million Amazon product

reviews from May 1996 to July 2014, substantially larger than previous work (and taking one month across four NVIDIA Pascal GPUs to train), and outperforms state-of-the-art models when tested on similar-domain corpora such as Rotten Tomatoes and IMDb reviews. Sentences are represented as a sequence of UTF-8 encoded bytes where for each byte, the model updates its hidden state and predicts a probability distribution over the next possible byte. [44]

- Stanford CoreNLP [19] provides a set of linguistic analysis tools, including sentiment analysis, given input text, while running in a local web server. Its sentiment analysis tool uses a recursive neural network model, representing text as parse trees, and trained on a Sentiment Treebank of fully-labelled parse trees for 215,154 unique phrases and 11,855 sentences from the Rotten Tomatoes movie review corpus; and classifies sentences into five sentiment classes, from 'very negative' to 'very positive' [45]. Although Stanford CoreNLP was written in Java, several packages exist that allow a Stanford CoreNLP local server to be started and queried programmatically in Python [46].
- TextBlob is a general-purpose natural language processing library similar to nltk, SpaCy, or CoreNLP. It provides two sentiment analysis models: PatternAnalyzer, a rule-based classifier based on part-of-speech pattern matching, and NaiveBayesAnalyzer, a Naïve Bayes classifier trained on a dataset of movie reviews. [47]

Several other repositories has also been explored, however deemed unsuitable for this project either due to requiring to be re-trained using labelled training data (which was unavailable for the domain of news articles), or the implementations are broken or infeasible.

For statistical analysis and plotting, the conventionally used libraries in Python are numpy [48], scipy [49], scikit-learn [20], and matplotlib [21]. Numpy provides a powerful and efficient n-dimensional array object (often used as requirement for other libraries), and functions to perform mathematical operations over real values, vectors (1-dimensional arrays), and matrices (2-dimensional arrays) such as scalar/vector/matrix addition, multiplication, extracting columns of a matrix to a vector, and boolean filtering [48]. Scipy is a library that extends numpy to provide additional domain-specific functions, providing tools such as sparse matrices and implementations of statistical equations such as estimating distributions [49]. Scikit-learn provides implementations of algorithms for data analysis, feature extraction, and machine learning, such as the CountVectoriser used to compute term frequencies [20]. Matplotlib is a 2D plotting library that produces visual graphs from lists/arrays [21]. It provides the capability to generate various types of plots, such as scatterplots, line plots, histograms, and box-and-whisker plots; modify the plot parameters (such as colours, labels, and bounds), create a grid of axes and plot multiple graphs in the same axes, generate a legend or colorbar, among other features.

# Chapter 3

# Requirements and Analysis

## 3.1 Problem Statement

## 3.2 Requirements

### 3.2.1 Data Collection

### 3.2.2 Dataset Filtering

### 3.2.3 Rule-based Sentence Matching

### 3.2.4 Sentiment Scoring

### 3.2.5 Statistical Analysis and Plotting

## 3.3 Analysis of Requirements

# Chapter 4

# Design and Implementation

## 4.1 Overall Design

## 4.2 Dataset Description

### 4.2.1 Sources

### 4.2.2 Keywords, key phrases, and query terms

### 4.2.3 Dataset Size

### 4.2.4 Limitations

## 4.3 Components

### 4.3.1 Data Collection

### 4.3.2 Dataset Filtering

### 4.3.3 Rule-based Sentence Matching

### 4.3.4 Sentiment Scoring

#### 4.3.4.1 Comparison of open-source sentiment models

#### 4.3.4.2 Final implementation

### 4.3.5 Statistical Analysis and Plotting

# Chapter 5

# Results Evaluation

# Chapter 6

# Conclusions

## 6.1 Achievements

## 6.2 Evaluation

## 6.3 Future Work

# Bibliography


[1] E. Cambria and B. White, "Jumping NLP curves: A review of natural language processing research," *IEEE Computational intelligence magazine*, vol. 9, no. 2, pp. 48–57, 2014.

[2] T. Lansdall-Welfare, S. Sudhahar, J. Thompson, J. Lewis, F. N. Team, N. Cristianini, A. Gregor, B. Low, T. Atkin-Wright, M. Dobson, *et al.*, "Content analysis of 150 years of British periodicals," *Proceedings of the National Academy of Sciences*, vol. 114, no. 4, E457–E465, 2017.

[3] Z. Waseem, "Are you a racist or am I seeing things? Annotator influence on hate speech detection on Twitter," in *Proceedings of the first workshop on NLP and computational social science*, 2016, pp. 138–142.

[4] J. Coverdale, R. Nairn, and D. Claasen, "Depictions of mental illness in print media: A prospective national sample," *Australian & New Zealand Journal of Psychiatry*, vol. 36, no. 5, pp. 697–700, 2002.

[5] N. Gold and G. K. Auslander, "Media reports on disability: A binational comparison of types and causes of disability as reported in major newspapers," *Disability and rehabilitation*, vol. 21, no. 9, pp. 420–431, 1999.

[6] O. F. Wahl, "Mass media images of mental illness: A review of the literature," *Journal of Community Psychology*, vol. 20, no. 4, pp. 343–352, 1992.

[7] K. Scior, "Public awareness, attitudes and beliefs regarding intellectual disability: A systematic review," *Research in Developmental Disabilities*, vol. 32, no. 6, pp. 2164–2182, 2011.

[8] K. Devotta, R. Wilton, and N. Yiannakoulias, "Representations of disability in the Canadian news media: A decade of change?" *Disability and rehabilitation*, vol. 35, no. 22, pp. 1859–1868, 2013.

[9] S. C. Jones and V. Harwood, "Representations of autism in Australian print media," *Disability & Society*, vol. 24, no. 1, pp. 5–18, 2009.

[10] J.-B. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, *et al.*, "Quantitative analysis of culture using millions of digitized books," *science*, vol. 331, no. 6014, pp. 176–182, 2011.

[11] F. W. Gibbs and D. J. Cohen, "A conversation with data: Prospecting Victorian words and ideas," *Victorian Studies*, vol. 54, no. 1, pp. 69–77, 2011.

[12] M. Mauch, R. M. MacCallum, M. Levy, and A. M. Leroi, "The evolution of popular music: USA 1960–2010," *Royal Society open science*, vol. 2, no. 5, p. 150 081, 2015.

[13] K. Leetaru, "Culturomics 2.0: Forecasting large-scale human behavior using global news media tone in time and space," *First Monday*, vol. 16, no. 9, 2011.

[14] I. Flaounas, O. Ali, T. Lansdall-Welfare, T. De Bie, N. Mosdell, J. Lewis, and N. Cristianini, "Research methods in the age of digital journalism: Massive-scale automated analysis of news-content—topics, style and gender," *Digital Journalism*, vol. 1, no. 1, pp. 102–116, 2013.

[15] P. Gooding, "Mass digitization and the garbage dump: The conflicting needs of quantitative and qualitative methods," *Literary and linguistic computing*, vol. 28, no. 3, pp. 425–431, 2013.

[16] T. Schwartz, "Culturomics: Periodicals gauge culture's pulse," *Science*, vol. 332, no. 6025, pp. 35–36, 2011.

[17] M. Mitchell, K. Hollingshead, and G. Coppersmith, "Quantifying the language of schizophrenia in social media," in *Proceedings of the 2nd workshop on Computational linguistics and clinical psychology: From linguistic signal to clinical reality*, 2015, pp. 11–20.

[18] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[19] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Association for Computational Linguistics (ACL) System Demonstrations*, 2014, pp. 55–60. [Online]. Available: `http://www.aclweb.org/anthology/P/P14/P14-5010`.

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

[21] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing In Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007. DOI: `10.1109/MCSE.2007.55`.

[22] M. Xia. (Oct. 20, 2017). A curated list of sentiment analysis methods, implementations and misc., [Online]. Available: `https://github.com/xiamx/awesome-sentiment-analysis` (visited on 04/18/2018).

[23] Keon. (Apr. 16, 2018). A curated list of resources dedicated to natural language processing (NLP), [Online]. Available: `https://github.com/keon/awesome-nlp` (visited on 04/18/2018).

[24] J. Misiti. (Mar. 26, 2017). A curated list of awesome machine learning frameworks, libraries and software, [Online]. Available: `https://github.com/josephmisiti/awesome-machine-learning` (visited on 04/18/2018).

[25] P. S. Foundation. (2018). PyPi – the python package index, [Online]. Available: `https://pypi.org/`.

[26] Anaconda, Inc. (2018). Anaconda cloud, [Online]. Available: `https://anaconda.org/`.

[27] J. F. Puget. (Dec. 19, 2016). The most popular language for machine learning is ..., [Online]. Available: https://www.ibm.com/developerworks/community/blogs/jfp/entry/What_Language_Is_Best_For_Machine_Learning_And_Data_Science?lang=en.

[28] GitHub, Inc. (Apr. 18, 2018). Topic: nlp, [Online]. Available: https://github.com/topics/nlp (visited on 04/18/2018).

[29] Anaconda, Inc. (2018). What is Anaconda? [Online]. Available: https://www.anaconda.com/what-is-anaconda.

[30] K. Reitz. (2018). Requests: HTTP for humans, [Online]. Available: http://docs.python-requests.org/en/master.

[31] L. Richardson. (Aug. 11, 2017). Beautiful Soup, [Online]. Available: https://www.crummy.com/software/BeautifulSoup.

[32] S. Robertson, "Understanding inverse document frequency: On theoretical arguments for IDF," *Journal of documentation*, vol. 60, no. 5, pp. 503–520, 2004.

[33] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[34] A. Khan, B. Baharudin, L. H. Lee, and K. Khan, "A review of machine learning algorithms for text-documents classification," *Journal of advances in information technology*, vol. 1, no. 1, pp. 4–20, 2010.

[35] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[36] Explosion AI. (2018). SpaCy: Industrial-strength natural language processing in Python, [Online]. Available: https://spacy.io.

[37] J. D. Choi, J. Tetreault, and A. Stent, "It depends: Dependency parser comparison using a web-based evaluation tool," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, vol. 1, 2015, pp. 387–396.

[38] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[39] GitHub, Inc. (Apr. 18, 2018). Topic: sentiment-analysis, [Online]. Available: https://github.com/topics/sentiment-analysis (visited on 04/18/2018).

[40] E. Gilbert and C. Hutto, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," in *Eighth International Conference on Weblogs and Social Media (ICWSM-14)*, 2014. [Online]. Available: http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf.

[41] V. Narayanan, I. Arora, and A. Bhatia, "Fast and accurate sentiment classification using an enhanced naive bayes model," in *International Conference on Intelligent Data Engineering and Automated Learning*, Springer, 2013, pp. 194–201.

[42] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, vol. 1, Baltimore, USA, Jun. 2014.

[43] P. Khaturia. (Jan. 20, 2018). Sentiment classification using word sense disambiguation, [Online]. Available: https://github.com/kevincobain2000/sentiment_classifier (visited on 04/18/2018).

[44] A. Radford, R. Józefowicz, and I. Sutskever, "Learning to generate reviews and discovering sentiment," *CoRR*, vol. abs/1704.01444, 2017. arXiv: 1704.01444. [Online]. Available: http://arxiv.org/abs/1704.01444.

[45] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, 2013, pp. 1631–1642.

[46] Lynten. (Feb. 14, 2018). Python wrapper for Stanford CoreNLP, [Online]. Available: https://github.com/Lynten/stanford-corenlp (visited on 04/18/2018).

[47] S. Loria. (2018). Textblob: Simplified text processing, [Online]. Available: http://textblob.readthedocs.io/en/dev.

[48] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006, vol. 1.

[49] E. Jones, T. Oliphant, and P. Peterson, *SciPy: Open source scientific tools for Python*, http://www.scipy.org/, 2001. [Online]. Available: http://www.scipy.org.

# Appendix A

# Other appendices, e.g., code listing

*Put your appendix sections here*