LONDON'S GLOBAL UNIVERSITY

Using Natural Language Processing (NLP) to develop a pipeline to analyse media representation of people with disabilities in Web-based news articles

Bagus Maulana[1]

MEng Computer Science

Catherine Holloway, Nicholas C. 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 (NLP) to develop a pipeline to analyse media representation of people with disabilities in Web-based news articles

Authors Name: Bagus Maulana

Supervisors Name: Catherine Holloway, Nicholas C. Firth

Date and Year of Submission: 30th April 2018

Research into the representation of groups (e.g. women, youth) in the news media is common across different research fields, from social sciences to computer science. A common approach is to manually analyse a small sample of a few hundred news articles and generalise an overall conclusion from that sample. Computational Natural Language Processing (NLP) could be used to process articles much faster and vastly increase the sample size, which could uncover further information from a text corpus, such as trends i.e. how the conclusion varies over independent variables such as time.

This project explores the feasibility of developing a computational pipeline that performs data collection from online news sources, filtering, parsing (feature extraction), sentiment scoring, statistical analysis, and data visualisation. This pipeline is then used in an experiment to collect articles from three major British online news publishers and show trends regarding how the terms used and sentiment in news articles varies over time and between different publishers when reporting news related to disability.

Results indicated that for some metrics, such as moving average of sentiment score, and on certain keyword categories, minor trends over time and between different publishers are apparent, although inconsistent. While the approach showed promise in performing quantitative analysis upon large bodies of literature, and specifically in the media representation domain, it is highly recommended that future approaches to this media analysis problem improves upon this work by training a domain-specific filter and sentiment scorer with labelled data to improve the accuracy, and thus consistency, of sentiment scoring.

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] defined 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 NLP umbrella include extracting term frequency distributions, text processing (e.g. tokenisation, stemming), part-of-speech tagging, text classification, information extraction (e.g. entity recognition), sentence structure parsing (parse tree), and sentiment analysis (or opinion mining) [2], [3]. The computational models used in NLP range from simple rule-based models (e.g. counting words, or term frequency) to statistical machine learning and neural network models [4].

An advantage of NLP is that machines could process vast bodies of human-created literature (books, articles, posts, e-mails, messages, etc.) much faster than humans can, processing thousands of text documents per second. This allowed for high-level quantitative analyses of thousands or millions of text documents from a vast corpus to be feasible, which could uncover information previously inaccessible from manually reading only a small sample of documents and generalising from the sample. This level of quantitative analysis could uncover trends and patterns from a text corpus, to answer questions such as "How does the popularity of the term 'mentally ill' increase or decrease year-on-year in British news media?"

Applying computational NLP to perform meta-analyses over large text corpora has interesting potential applications in improving our understanding of the human world, such as analysing cultural trends quantitatively [5]. One study assembled a vast corpus of regional newspapers in the United Kingdom spanning 150 years to detect long-term patterns of cultural change, such as the increase of female representation in the news, or the popularity of trains and horses for transportation [5]. This was achieved by analysing trends for n-gram frequency (a count of words or phrases in a text document) and named entities (known persons, organisations, locations, etc.) in text.

More specifically in the domain of media representation of particular groups of people, several researchers have attempted to use features extracted using NLP to perform computational analytics of text, mainly from social media. For example, a tool to classify racist and sexist posts in social media was developed by using NLP to extract n-grams and part-of-speech tags (labels of

words corresponding to its definition and context, such as 'noun' or 'verb') from text posts [6]. However, there is still a research gap in this area, especially for applying NLP for news articles, in the context of specific groups such as people with disabilities.

The representation of specific groups, such as people with disabilities, in media has been a popular research theme in social science. For example, a 2002 study analysed a sample of 600 print articles relating to mental illnesses in New Zealand to measure the proportions of positive and negative depictions and predominant themes (e.g. criminality, educational accomplishments) [7]. There were attempts to discover trends, such as a study conducted in 1998 [9] and replicated in 2008 [12] which assessed change in representations of disability and persons with disability in Canadian news media. However, the study provided only two data points (1998 and 2008) with relatively small sample sizes of 196 news articles in 1998 and 166 news articles in 2008.

Applying computational NLP to this field would allow the possibility of discovering higher-level trends, by computationally analysing a much larger sample of articles, then identifying trends by creating subsets based on independent variables such as year of publication and publisher. In this research, a sample of 305,185 news articles (48,967 after filtering off-topic articles) from British online news sources were used. However, challenges remain as contemporary syntax-based NLP approaches tend to be more limited in scope and are prone to inconsistencies (false positives and negatives), where mitigating these inconsistencies is currently an open area of research.

The aim of this project was to show the feasibility of utilising computational NLP approaches to perform a meta-analysis of literature available in the public online news media. More specifically, to collect news articles relating to people with disabilities in British online media, and perform analyses using NLP at scale to identify trends such as term popularity and variations in positive/negative sentiment over variables such as date published and publisher.

This project's goals were to achieve the stated aim by developing a computational pipeline capable of performing analysis on online news media in full, from data collection to analysis and visualisation. Given a list of topics related to disabilities, each topic consisting of key terms and query terms; this pipeline accomplished the task of web crawling and scraping web sites to collect a dataset of news articles; filtering relevant articles given key terms; extracting 'relevant' sentences that referred to a key term from these articles; performing sentiment analysis on these sentences; and producing relevant visualisations and statistical analyses to show trends. This pipeline is available open source on GitHub (`https://github.com/bmaulana/nlp-media`).

The main NLP techniques that were relevant for this project are: Text processing, to parse text and other relevant information from web news articles, and 'prepare' text for further analyses using tokenisation (splitting text into a list of tokens, or words) and stemming (reducing words to their word stem, e.g. talked $\rightarrow$ talk); term (n-gram) frequency, to quantitatively count the occurrence of words and phrases (sequences of words) in an article; sentiment analysis, to produce a 'sentiment score' of news articles that correspond to its perceived positive/negative view. These techniques were implemented by utilising open-source NLP implementations.

This project was carried out in a modular approach. The pipeline was developed as 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 extract term occurrences and relevant sentences from articles, given a list key terms; a sentiment scorer for sentences and articles; and a script to perform statistical analysis on the results and produce relevant plots. A main pipeline

script connects these components together by calling them in sequential order, performing analyses for each topic and Web news source (Daily Mail, Daily Express, Guardian). Each component's output is saved to a JSON [60] file, and the next component reads the previous component's output file, which ensured that computation can be 'resumed' without recomputing the previous component.

The body of this report is subdivided into four chapters, followed by a conclusion and appendices. Chapter 2 covers related work on the domain of news media analysis (especially in the context of people with disabilities), a background of NLP research, and information regarding the relevant technologies researched for this project. Chapter 3 defines a structured list of requirements, goals, and expectations for this project. Chapter 4 documents the design and implementation of the computational pipeline and its components that were used to carry out the data analytics experiment and achieve the stated goals. Chapter 5 discusses the experiment's results in diagrams, graphs, and tables. Finally, the conclusion, Chapter 6, evaluates the project, summarises key achievements and takeaways, provides recommendation on how this work could be expanded upon, and sets guidelines for further work in this field. Furthermore, the bibliography lists sources that were used as references in this project, and the Appendix section contains the raw results of the experiment, including graphs and data not in Chapter 5.

The performance of several open-source sentiment analysis implementations were compared to judge their suitability for the sentiment scoring task, given the domain of sentences from news articles related to disabilities or people with disabilities. Results indicated that a relatively simpler rule-based model, VADER [8], outperformed more complex supervised machine learning or neural network models, which had been trained on other domains, such as tweets and IMDb/Amazon reviews, and proved to be less generalisable for this domain.

This project proved the feasibility of utilising NLP-based technologies to derive trends from a large corpora of online news articles relating to disabilities or people with disabilities. The results showed that, for example, the perceived sentiment of Guardian articles are, on average, significantly higher than Daily Express and Daily Mail articles for the topic 'disabled'. It also showed an decreasing trend over time on the use of 'invalid' and 'handicap(ped)', and an increasing trend for 'accessibl(e)', within Guardian articles.

These results could have a substantial impact on how research should be conducted for similar studies on the media's representation of groups of people, as it showed that NLP could be used to analyse a much larger sample of text documents than traditional approaches and derive meaningful trends. However, challenges remained due to the sentiment scorer's inaccuracy. The accuracy and consistency of results could be further improved in future work by developing a domain-specific supervised model for filtering and sentiment scoring, trained on an adequately-sized labelled dataset of news articles within the domain.

Chapter 2

Context

2.1 Background

Analyses of news media in its various forms (print, online, etc.) is a popular research method. News media provides an overview of the prevailing society's conceptions or views regarding a theme or topic, which can be analysed to deduce quantitative information. This approach had been commonly used to study representations of particular groups of people, as the language used in the media reflects and shapes prevailing 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 [9]. Furthermore, there is evidence to suggest that news media sources contribute to shape and reinforce beliefs among the society, such as misconceptions and stigma [10].

There has been various studies related to the public awareness of disabilities. A review in 2011 [11] 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. Their inclusion criteria omits studies which were found to be irrelevant, duplicate, or not written in English; and only accepts articles which were published in full in peer-reviewed journals, and the study's subject had to be the general public of working age (instead of particular subgroups). This showed that there is interest within the research community to find new ways to understand and quantify the public's perception towards disabilities.

Analyses of news media are primarily carried out by taking a small 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 [7], researchers analysed a sample of 600 print articles relating to mental health or mental illness that were 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 [9] and replicated in 2008 [12] assessed 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 [13] 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.).

Data collection and processing of text using computational techniques is much more feasible in scale, cost and time relative to manual collection and reading of text. An automated script could be used to collect thousands of news articles published on the Internet per hour (varies on Web source, hardware, Internet connection, etc.), a vast improvement over contracting a commercial clipping bureau to provide 600 articles as done in past studies [7]. By applying NLP-based computational techniques in analysing text, it should be possible to develop a pipeline that could analyse and extract quantitative information from these articles at a much faster rate than manual reading, enabling the analyses of a much larger scale of documents.

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 should be possible to uncover additional information from the data set. For example, higher-level trends, such as how a conclusion varies by year, location, and publisher, may be discovered from a quantitative analysis of the larger dataset, by 'splitting' the result set into smaller subsets based on independent variables such as year, location, and publisher, and comparing these subsets based on dependent variables such as term frequency. Furthermore, computational pipelines for data collection and scoring news articles are more objective and reproducible than manual methods, which may vary due to each researcher's individual biases.

Several attempts have been made to take advantage of this approach to carry a more complete analysis of textual corpora. A team of researchers 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 [5] . With this approach, the researchers were able to extract quantitative time-series information from 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 the researchers 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, such as 'train' overtaking 'horse' in popularity around 1900, or 'labour party' overtaking 'conservative party' and 'liberal party' in news coverage from the 1920s. Additionally, they 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 and male entities, and entity categories such as their age and occupation. They also considered the geographical location of the publication, to see how usage trends of words such as 'british' and 'english' differ based on location.

The British news-media study were inspired by prior discussions and studies on the potential of exploiting large text corpora to detect macroscopic, long-term cultural changes. A seminal study in 2011 [14] started the field of 'culturomics', or performing large-scale quantitative analysis of text corpora. 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) 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 them, several other studies have been published adopting a similar approach:

- an analysis of 1.7 million Victorian-era books [15]
- an analysis of 17,094 US Billboard Hot 100 songs between 1960 and 2010 [16]
- an analysis of a 3.9 million news article sample from the Summary of World Broadcasts (SWB) collection [17]
- an analysis of 2.5 million English-language million news articles from 498 online news outlets from 99 countries [18]

This approach has also been criticised as it ignored semantics and context. For example, critics has noted that "thirteen hundred words of gibberish and the Declaration of Independence are digitally equivalent" [19], issues with OCR quality and duplicate editions [19], or that the selection of digitised books are biased [20].

There has been some progress in applying NLP specifically in the domain of how (specific) groups of people are represented in the media:

- An attempt to extract features (such as n-grams and part-of-speech tags) using NLP to classify racist and sexist posts in social media, providing annotations to 6,909 tweets [6].
- A study that explored potential linguistic markers of schizophrenia in social media; using a dataset of 174 users with self-reported schizophrenia and up to 3,200 tweets per user, and a similarly-sized dataset of 'control' (non-schizophrenic) users. The researchers used a support vector machine (SVM) model, using NLP to extract 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 from tweets [21].

There is still a visible research gap in this area, especially for using computational NLP approaches specifically for news articles, and/or with regards to the representations of specific groups, such as people with disabilities (or a specific disability).

To date, advances in NLP research have made this approach much more feasible, efficient, and effective, even given limited time and hardware constraints. A growing number of free and open-source tools for computational NLP and statistical analysis has been developed by the research community, such as nltk [2], SpaCy [22], and StanfordNLP [23] as general NLP tools, scikit-learn [24] for statistical analysis, and matplotlib [25] 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. Section 2.3 contains a further listing and discussion of these tools, alongside the specific libraries and NLP 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 NLP, specific NLP techniques, disability, and news media: for example, 'natural language processing review', 'news media' AND 'disability', 'natural language processing' AND ('news media' OR 'cultural trends'), 'natural language processing' AND 'disability', and 'sentiment analysis'. 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 was 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 was carried out in an iterative approach alongside software development. Initial planning and research would provide an initial implementation plan, which the implementation of would uncover feasibility of these approaches and possible alternatives/refinements to be researched, and 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 and/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 [4], [26], [27].
- Python package repositories such as PyPi [28] and Anaconda Cloud [29], 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 above, computational implementations of NLP 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 data visualisation. Among these components, NLP 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 data visualisation is performed using general-purpose statistical tools and metrics. This section will cover the technical tools researched and used for all listed components regardless.

NLP 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 approaches to NLP 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, for example, the word '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 NLP, sentiment analysis, statistical analysis, data visualisation, 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 [30], 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 (a Python-based interactive 'notebook' technology) repositories with the tag 'nlp', compared to the second most popular programming language being Java with only 251 repositories tagged with 'nlp' [31]. 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. The Anaconda distribution of Python [32] is used for its suitability to set up and manage Python environments and packages for data science projects.

2.3.1 Data Collection Methods

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 [33] 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 [34] 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.

2.3.2 Filtering Methods

In this project, filtering of off-topic articles were achieved via ranking the term frequency of key terms independently for each document. Term frequency (tf) is a simple and commonly-used metric in NLP, with various existing tools that can compute this metric for thousands of text documents within seconds. To put simply, the frequency of a term (a token, or sequence of tokens) in a document is the number of times that the term occurs in the document. The popular scikit-learn library [24] provides a tool to measure term frequency of text documents, handling both tokenisation (converting a text document into a list of tokens, or terms/words) and counting word occurrence. It also provides the option to ignore stop words (common words in English which do not add topical information, such as 'the' or 'a'), which can often bias results due to their relative high frequency. Additionally, the nltk library [2] 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.

In literature related to NLP, more complex 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) [35], [36], 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, it is deemed to be less important and assigned a lower score. For example, common words such as 'the' are assigned very low scores. However, this approach was not suitable for this project, given the selective nature of the dataset, as only articles containing certain query terms are collected; thus the idf values of query terms were be flawed, as a query term exist 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 [37]. However, this approach was not feasible for this project due to a lack of labelled data (i.e. a dataset of articles and category labels, or in this case 'is this article relevant?' boolean labels).

Topic models, which compute the proportion of abstract 'topics' in a document, has also been proposed. Latent dirichlet allocation (LDA) [38] 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. Additionally, LDA is significantly more computationally expensive than tf or tf-idf.

2.3.3 Text Parsing and Sentiment Analysis Methods

Sentiment scoring of articles is sub-divided into two components: a component to find sentences relevant to a topic, or a list of key terms, in a text document (sentence matching); and another component that performs sentiment analysis on these sentences, and transforms sentences to a real-valued sentiment score. SpaCy [22] 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 [39]. 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 language [3]. 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' [40]. Furthermore, a community-curated list of sentiment analysis methods and implementations exist [4], 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 [8] is a relatively simple parsimonious rule-based model that scores the sentiment of a given sentence, based on rules such as: 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; negations (e.g. 'not good'); and 'emphasis', or increased sentiment intensity due to punctuation, capitalisation, and degree modifiers (e.g. 'very').
- xiaohan2012's 'twitter-sent-dnn' repository provides a trained 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 inferred labels based on emoticons. It is an implementation of [41].
- kevincobain2000's 'sentiment_classifier' repository [43] provides a trained 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 uses bigrams as features, implements Word Sense Disambiguation using wordnet [42] to tranform bigrams to 'senses', and considers word occurrence statistics from nltk's movie review corpus. Its training data is a mixture of nltk's movie review corpus, Twitter posts, and Amazon customer reviews data.
- 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. It is an implementation of [44].

- Stanford CoreNLP [23] 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, represents text as parse trees, and were 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. Unlike other scorers in this list, it classifies sentences into five sentiment classes, from 'very negative' to 'very positive', instead of assigning a real-valued score. [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 [47] is a general-purpose NLP 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 (with no information on features used or dataset size).

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.

2.3.4 Statistical Analysis and Data Visualisation Methods

For statistical analysis and data visualisation, the conventionally used libraries in Python are numpy [48], scipy [49], scikit-learn [24], and matplotlib [25]. 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 [24]. Matplotlib is a 2D plotting library that produces visual graphs from lists/arrays [25]. 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.

The types of plots and statistical analysis metrics that are deemed relevant for this analysis are:

- Scatter plot, used to show the distribution of data within two variables (e.g. year published and sentiment score), and colour could be added to show a third variable (e.g. source/pub-

lisher of article). Available on Matplotlib.

- Histogram (and two-dimensional histogram), used to show the distribution of articles relative to variables such as publisher, year of publication, and sentiment score ranges. Available on Matplotlib.
- Box-and-whiskers [50] and violin plot [51], also used to show and compare the distribution of dependent variables (e.g. sentiment score) within different subsets of the data separated by independent variables (e.g. publisher). Available on Matplotlib.
- Line graph, used to show trends in a dependent variable (e.g. sentiment score) over an independent variable (e.g. year of publication). Available on Matplotlib.
- Mean and standard deviation, to quantify the distribution of articles within different subsets of the data separated by independent variables (e.g. publisher and year of publication) and provide a quantitative measure to compare different subsets. Available on SciPy.
- Mann-Whitney U Test [52], a non-parametric statistical test that measures whether it is true that given a randomly-selected value from a distribution, and another randomly-selected value from another distribution, the first value is equally likely to be less than or greater than the second value (i.e. there are no statistically significant difference between the two distributions), to show if the difference between two subsets are statistically significant. Available on SciPy.

Chapter 3

Requirements and Analysis

3.1 Problem Statement

The primary aim of this project is to utilise available NLP-based technologies in order to perform a high-level meta-analysis of online news articles relating to people with disabilities available in the online British news media, with the goal of revealing trends by varying for independent variables such as source and year published. The solution would need to collect and scrape online news articles from the Internet, filter only relevant articles, use available NLP-based tools to extract information from these articles, perform statistical analyses, and show visualisations of the resulting data to show trends. Of particular interest, as a dependent variable, is a sentiment (or opinion/polarity) index of articles ("how positively does an article view disabilities or people with disabilities?"), and how it varies given independent variables such as source and year published. Thus, the general solution defined by this aim would involve the completion of several sub-problems, primarily data collection, filtering, parsing, sentiment scoring, and statistical analysis and visualisation.

To achieve this aim, the project's goals are to develop a computational pipeline that implements all components required for the general solution. Before the project could be started, the topics relevant to this project had to be defined. Thus, a list of topics relating to the domain of people with disabilities would need to be compiled, each topic consisting of a list of keywords (or key phrases) and query terms related to a specific disability (or disabilities in general). Given this list of topics, this pipeline has to implement these following tasks (goals):

- Web crawling and scraping web pages, using public APIs where possible, to collect a dataset of news articles, given the list of query terms for each topic.
- Filtering relevant articles from the dataset, given the list of keywords for each topic.
- Extracting relevant sentences that refer to a keyword, from the dataset of filtered articles (feature extraction).
- Performing sentiment analysis (using open-source libraries and pre-trained models) on the dataset of relevant sentences.
- Producing relevant statistical analyses and data visualisation to show trends over independent variables such as source and year published.

As the primary advantage of computational NLP is in its speed, and thus analysed sample size,

relative to human reading, the solution must be able to perform these computations quickly and at scale. The total number of documents available, from the sources used in this experiment (section 4.2.1) and given defined query terms (section 4.2.2), is expected to number in the thousands to tens of thousands of documents per topic, or hundreds of thousands of documents in total across all topics. Given this scale, the solution must be able to compute the full pipeline within a feasible timeframe (not more than a few days), given available consumer-grade hardware (Intel i7-6700HQ CPU @ 2.60GHz, NVIDIA GeForce GTX 1060 GPU) to show that the solution is feasible without specialised hardware.

3.2 Requirements

The solutions follows a component-based design, where each sub-problem must be implemented by a component that focuses only on the sub-problem. These components must be linked together via a 'pipeline' script that executes each component in order, and iterates through the list of topics and sources. This is ideal such that changes could be made to a component without affecting (or needing to re-write) code in other components. The list of required components are as follows: data collection, dataset filtering, rule-based sentence matching, sentiment scoring, statistical analysis and visualisation.

3.2.1 Data Collection

The data collection component must be able to find a list of news articles for each supported online source (Daily Mail, Daily Express, and Guardian), given a list of query terms related to each topic. For each article, the only information required at this stage is a working URL pointing to an online resource containing the article text and relevant metadata. Thus, the information that has to be provided by the component after this stage is a list of URLs pointing to relevant articles given a list of search queries.

After the list of URLs pointing to online news articles has been compiled, the component must be able to scrape these articles and extract the full article text and relevant metadata from each URL. Aside from the full article text, the relevant metadata that needs to be extracted are the article's headline, URL, date of publication, and publication source. The information is returned in the form of an array of JSON objects, with each JSON object containing the text and metadata of a single news article. This information must then be saved locally to a file where it will be read by the next component.

The file and directory structure of the output file must be consistent given source and topic, such that the next component can find it programmatically. This requirement holds for all other components in this pipeline.

As data collection is expected to consume the longest time compared to other components, due to the necessity to submit a web request for each article, additional requirements regarding scalability are enforced. The data collection component must run in reasonable time (i.e. less than a day for each topic), given available hardware and a scale of up to tens of thousands of articles per topic. Additionally, it should be possible to resume progress on data collection, such that it is possible to re-run the program at a later date to add new articles without sending web requests

for articles already in the collection. It would also help in cases where the program is interrupted, Internet connection is lost, the machine is shut down, etc. Thus, the component should be able to store already-parsed articles to an external file, and read from the external file upon starting to gather a list of already-parsed URLs, and avoid re-parsing existing URLs.

3.2.2 Dataset Filtering

The next component handles dataset filtering. Initially, this component must be able to load the dataset of parsed articles from the file saved by the data collection component. Each file contains a subset of all parsed articles for a specified topic and a specified source. For each article, it must decide whether it is relevant to the topic defined by the subset, given a list of key terms related to each topic. This decision should be based by the article's full text and headline, and should take all key terms associated to the current topic into account. It should also be able to show/print a sample of an arbitrary number of documents, which would be used to analyse and improve the accuracy of the filter.

The component must save all articles deemed on-topic (relevant to the topic) to a new output file, containing an array of JSON objects in the same format and with all the same information as in the data collection component's output file. Articles deemed off-topic (not relevant to the topic) must not be saved to the output file.

3.2.3 Feature Extraction and Rule-based Sentence Matching

The next component parses the article text using open-source NLP tools to extract relevant information required by the sentiment scoring and statistical analysis components. Given an input file which is the output file of the dataset filtering component, this component must be able to load the article text and metadata of all saved articles. Then, information should be extracted by syntactically parsing each article's text and headline. The sentiment scoring component would require all sentences relating to a keyword or key phrase (containing a keyword, key phrase, or an equivalent term) to be extracted from each article. Additionally, the component should also extract other information as required by the sentiment scoring and statistical analysis components, including the total number of sentences and the number of relevant sentences in the document, and the term frequency of each keyword and key phrase.

After these data have been extracted from each article, it must save the information to a new output file, in the form of an array of JSON objects where each JSON object contains all the information about a single news article (with a key for each 'feature' e.g. relevant sentences). The metadata of each article (headline, date of publication, and source) should also be saved to the new output file, as it would be required as independent variables for the statistical analysis component to analyse trends in the dataset.

3.2.4 Sentiment Scoring

The sentiment scoring component computes a real-valued score for each relevant sentence extracted by the previous component, which must correspond to the perceived 'sentiment' of the sentence towards a disability, disabilities, a person with a disability or disabilities, or people with

disabilit(y/ies), referred by the keyword or key phrase, with sufficient accuracy. Given an input file which is the output file of the dataset filtering component, this component must be able to load the relevant sentences and other data for all saved articles.

Two iterations of the sentiment scorer component must be developed. The first iteration of the sentiment scorer component is used to perform a comparison between several open-source sentiment analysis implementations for this sentiment scoring task. It is not used in the final pipeline. The second iteration of the sentiment scorer component only computes one sentiment score for every sentence, using the best-performing sentiment scorer shown by the first iteration. The second iteration is the one used in the final pipeline.

3.2.4.1 Comparison of Open Source Implementations

The first iteration of the component must select a sample of sentences from the dataset, with a sample size arbitrarily defined by the user. Also, it must implement all open-source sentiment analysis implementations ('sentiment scorer') that were listed in section 2.3.3. All sentences in the sample must be analysed and given a score by each sentiment scorer, and the component should also allow the user to manually label these sentences as positive, neutral, or negative. With this information, the component must be able to determine the accuracy of each sentiment scorer (i.e. show the proportion of true positives and true negatives, and show a confusion matrix), and store the total, per-sentence, and per-article runtime of each sentiment scorer.

This iteration of the component will not be used in the final pipeline, but only used as a tool to compare existing sentiment scorer implementations, and analyse their performance in the domain of sentences in news articles relating to disabilities or people with disabilities.

3.2.4.2 Final Implementation

The second iteration of the sentiment scorer component is used in the final pipeline. This component must perform sentiment analysis for all relevant sentences in the dataset (instead of only a limited sample of sentences). It must only compute one sentiment score using the best-performing sentiment scorer that runs in reasonable time, as shown by the first iteration of this component. It should also compute the average sentiment score for each article, given information of the sentiment scores for all relevant sentences in the article. Furthermore, the component must run in reasonable time (i.e. less than a day for each topic), given available hardware and a scale of up to tens of thousands of articles per topic.

After the sentiment scorer has scored all relevant sentences in the dataset, it must then save information about all articles, with score labels appended to each sentence, to a new output file. The JSON object format of each article in the output file should be equivalent to the input file's format (i.e. no information from previous components are lost), with the exception of an additional 'sentiment score' key-value pair within each sentence's JSON object.

3.2.5 Statistical Analysis and Visualisation

The last component performs statistical analysis and data visualisation. This component must read the sentiment scoring component's output files as input files, to load the dataset of news articles and features extracted by previous components about each article. At this point, information

extracted about each article by previous components must include: source, publication date, and sentiment score, alongside other information. To combine information about articles from different sources (represented by different files), this component should be capable of reading input from several different input files in a single run, and collate the information about every unique article in each file to a single dataset.

This component must be able to show trends, visualisations, statistical metrics that show how dependent variables (e.g. sentiment score) differ relative to the independent variables (e.g. publication year and source). The plot types and statistical metrics relevant to this analysis was defined in the end of section 2.3.4. At a minimum, this component must show how the sentiment score varies relative to publication year and source, and how subsets divided by publication year and source differ in distribution of sentiment scores, using the plots and metrics as defined in section 2.3.4. Additionally, the component should also show visualisations based on other information extracted by previous components as relevant, such as trends in the term frequency of each keyword and key phrase (as a dependent variable) over time (publication year, as an independent variable).

3.3 Analysis of Requirements

The implementation design (and additional requirements) of this project are highly influenced by the core requirements (i.e. data collection, filtering, parsing, sentiment scoring, and statistical analysis and visualisation). The overall design of the pipeline, with isolated components for each sub-problem, stemmed from having largely independent sub-problems that was required in order to perform the data analysis in full, from data collection to analysis and visualisation. Also, initial prototypes (initial 'runs' collecting only a limited number of news articles) showed that data collection took the majority of runtime in the pipeline. For this reason, the requirement where it should be possible to resume progress on data collection (and not repeat the process for existing data) was added.

The JSON object format of articles for each component's output file were largely defined by the requirements of each component (i.e. the information that must be extracted by each component). For example, the JSON object format of an article in the data collection component's output file is as follows:

```
{
        "https://www.express.co.uk/comment/columnists/...": {
                "source": "Daily Express",
                "title": "Will she grow out of her stutter?",
                "datetime": "2008-02-12T00:00:00+00:00",
                "section": "comment",
                "subsection": "columnists",
                "text": "..."
        }
}
```

Where the URL and 'source', 'title', 'datetime', and 'text' fields correspond to requirements defined for the data collection component.

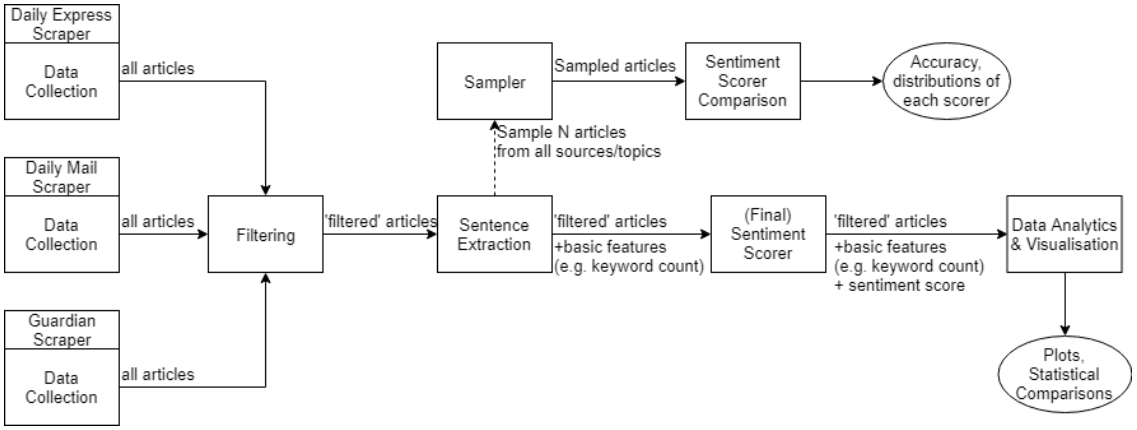
Chapter 4

Design and Implementation

4.1 Overall Design

To fulfil the stated requirements in chapter 3, the solution that was decided is a computational pipeline consisting of five main components. This solution was used to carry out the experiment to collect and analyse online news articles and visualise potential trends in sentiment/opinion. The results of this experiment is shown in chapter 5.

The high-level design of the pipeline and its main components were defined as follows:



In this experiment, the pipeline was ran once for each source and topic, generating a dataset of all articles for the specified topic from the source with all extracted features, stored in a file. The 'plot' component then loads these files and combines the dataset of all articles for every source within the same topic, and produces various plots to show trends within the topic, and statistical metrics for each possible subset within the topic.

Components pass datasets to each other by storing all information it extracts in a JSON format. The output file of an earlier component is read as the input file for the next component, given the same source and topic. The format of the output/input files is a collection of JSON objects, where each line consists of a single JSON object, and each JSON object represents a single article. The JSON object contains a key-value pair for each extracted feature, and each component preserves the previous component's information (key-value pairs) while appending the object with additional key-value pairs for each feature it extracts. This design ensures that in

case computation is interrupted for any reason, the dataset generated by the previous component has already been saved to a file, and computation can restart from the interrupted component (without having to re-run previous components).

4.2 Dataset Description

4.2.1 Sources

The Internet is an increasingly common medium for news publishers to publish news articles and for consumers to read these articles. As of 2017, 64% of British individuals read and/or download online news, newspapers or magazines, a sharp increase from 20% in 2007 [53].

The Daily Mail (`dailymail.co.uk`) and the Guardian (`theguardian.com`) are the two most visited news publisher's web sites as of 2016, with a monthly viewership of 11.85 and 10.05 million respectively [54], and are the main subjects of this experiment. Additonally, The Daily Express (`express.co.uk`), a slightly smaller newspaper with a monthly viewership of 2.67 million in 2016 [54], is also added as a news source in this experiment, to control for source size and explore how the experiment performs with smaller sample sizes. Furthermore, the Guardian provides an API [55] and the Daily Mail and Daily Express provide advanced search tools [56], [57] to query news articles from their websites, which prove useful in their respective data collection component's implementation, although the pipeline would work with any online news source that provides an internal search tool. Thus, the dataset for this experiment contained news articles from these three news publishers or sources; the Daily Mail, the Guardian, and the Daily Express.

4.2.2 Topics, Key Terms, and Query Terms

A list of topics relevant to disabilities, and a list of key terms for each topic, were compiled for this experiment. Furthermore, a list of query terms were compiled based on the list of key terms, with words/phrases that could have other meanings (or combinations of multiple common words) removed (unless the word/phrase is very commonly used to refer to the disability, such as 'blind' and 'mute'). Query terms were used in data collection (where ambiguous terms were removed to reduce off-topic articles), while key terms were used in subsequent components.

The final list of topics relevant to disabilities and people with disabilities, and the key terms and query terms deemed relevant to each topic, were:

Topic	Key Terms	Query Terms
'disabled'	'disabled', 'disability', 'handicapped', 'cripple', 'invalid', 'accessible', 'ablism', 'ableism', 'differently abled'	'disabled', 'disability', 'ablism', 'ableism', 'differently abled'
'autism'	'autism', 'autistic', 'asperger\'s', 'ASD'	'autism', 'autistic', 'asperger\'s', 'ASD'

‘blind’	‘blind’, ‘blindness’, ‘blindism’, ‘visual impairment’, ‘partially sighted’, ‘vision loss’	‘blind’, ‘blindness’, ‘visual impairment’, ‘partially sighted’, ‘visually impaired’
‘cerebral palsy’	‘cerebral palsy’, ‘spastic’	‘cerebral palsy’, ‘spastic’
‘deaf’	‘deaf’, ‘deafness’, ‘hearing impaired’, ‘hard of hearing’, ‘hearing loss’	‘deaf’, ‘deafness’, ‘hearing impairment’, ‘hard of hearing’, ‘hearing impaired’
‘developmental delay’	‘developmental delay’, ‘developmental disability’, ‘developmental disorder’, ‘learning disability’, ‘slow learner’, ‘intellectual disability’	‘developmental delay’, ‘developmental disability’, ‘developmental disorder’, ‘learning disability’
‘dyslexia’	‘dyslexia’, ‘dyslexic’	‘dyslexia’, ‘dyslexic’
‘epilepsy’	‘epilepsy’, ‘epileptic’, ‘seizure’	‘epilepsy’, ‘epileptic’
‘mental illness’	‘mental illness’, ‘mental health’, ‘mental disability’, ‘mental disorder’, ‘mental issue’, ‘brain injured’, ‘brain injury’, ‘brain damaged’, ‘psychological’, ‘psychiatric’, ‘emotional disorder’, ‘behavioural disorder’, ‘retardation’, ‘intellectual disability’, ‘mentally ill’, ‘mentally disabled’, ‘mentally handicapped’	‘mental illness’, ‘mental health’, ‘mental disorder’, ‘mental disability’, ‘mentally ill’, ‘mentally disabled’, ‘mentally handicapped’
‘mute’	‘mute’, ‘muteness’, ‘mutism’, ‘cannot speak’, ‘difficulty speaking’, ‘synthetic speech’, ‘non-vocal’, ‘non-verbal’	‘mute’, ‘muteness’, ‘mutism’, ‘non-verbal’
‘paralysis’	‘paraplegic’, ‘quadriplegic’, ‘spinal cord’, ‘paraplegia’, ‘quadriplegia’, ‘paralysed’, ‘paralyzed’, ‘paralysis’, ‘crippled’, ‘leg braces’, ‘wheelchair’	‘paraplegic’, ‘quadriplegic’, ‘paraplegia’, ‘quadriplegia’, ‘paralysis’

'speech impairment'	'speech impairment', 'stutter', 'speech disability', 'speech disorder', 'communication disability', 'difficulty speaking', 'language impairment', 'language disorder', 'language disability', 'speech impediment', 'stammer'	'speech impairment', 'stutter', 'speech disorder', 'speech impediment'

This list of key terms was roughly based on guidelines from the Californian [58] and UK [59] governments (ignoring whether the term is considered 'appropriate' or 'inappropriate', as terms labelled 'inappropriate' are often still commonly used in the news, and relevant to consider when deciding whether an article is on-topic), with a few additions based on other commonly-used terms found on sampled articles from the Daily Express, the Daily Mail, and the Guardian.

4.2.3 Dataset Size

The dataset is comprised of all articles found online given the query terms defined in section 4.2.2, published between 2000 to (approximately) end of March 2018. The size of the initial collected dataset (i.e. all articles collected by the Data Collection component, prior to any further processing) in number of articles, for each source and topic, were:

Topic	Daily Express	Daily Mail	Guardian	Total
Disabled	16,818	24,768	30,598	72,184
Autism	988	6,035	5,780	12,803
Blind	9,467	23,616	32,307	65,390
Cerebral Palsy	509	1,995	1,569	4,073
Deaf	6,795	20,686	8,163	35,644
Developmental Delay	965	3,529	1,517	6,011
Dyslexia	283	938	1,980	3,201
Epilepsy	700	2,924	2,368	5,992
Mental Illness	8,102	38,273	29,831	76,206
Mute	1,541	2,312	4,764	8,617
Paralysis	711	4,346	3,879	8,936
Speech Impairment	1,777	2,994	1,357	6,128
Total	48,656	132,416	124,113	305,185

After filtering, the size of the dataset that was plotted, in number of articles for each source and topic, were:

Topic	Daily Express	Daily Mail	Guardian	Total
Disabled	1,852	6,035	8,524	16,411
Autism	128	1,755	1,278	3,161
Blind	775	3,008	3,894	7,677
Cerebral Palsy	25	253	75	353
Deaf	114	747	901	1,762
Developmental Delay	5	130	447	582
Dyslexia	19	110	281	410
Epilepsy	58	740	374	1,172
Mental Illness	398	6,794	8,137	15,329
Mute	24	147	285	456
Paralysis	53	1,003	405	1,461
Speech Impairment	57	73	85	215
Total	3,508	20,813	24,646	48,967

For the results evaluation, the focus will be on the 'disabled' topic, as it has the highest amount of news articles within its subset, and is the most generalisable topic on disabilities (as it refers to the general theme of disabilities and people with disabilities, rather than a specific topic).

4.2.4 Limitations

As shown in section 4.2.3, the Daily Express has much fewer articles for any given topic than the Daily Mail or the Guardian. Some topics, such as cerebral palsy, developmental delay, dyslexia, mute, and speech impairment, are also severely lacking in sample size of articles (post-filter). In particular, cases where there are less than ~200 articles from a source within a topic were problematic to plot or form statistically-significant conclusions regarding trends (e.g. to compare with other sources), as the distribution of the data is too varied. Cross-referencing the results in section 5.4 with the size of each subset showed that it was difficult to obtain statistically significant conclusions when comparing to subsets with less than ~200 articles. This means that it is difficult to analyse or compare the Daily Express's subset for topics other than general 'disabled', 'blind', and 'mental illness', due to its relative lack of sample size.

Another limitation with this experiment is the length of time that each news source retain articles for in their online archive. Our dataset indicates that by the end of March 2018 (when the data was collected for this experiment), the Daily Express only retains articles from after ~2007, the Daily Mail only retains articles from after ~2010, and the Guardian only retains articles from after ~2000. This raises an issue when analysing year-on-year trends, as the proportion of articles' sources within a topic are different in each year, and year-on-year differences may be better explained due to this difference in proportion, rather than an actual trend (see also: section 5.3). For this reason, trends over time are only considered to be significant when the trend is consistently repeated for each source's subset (instead of the full dataset for that topic), for all sources with statistically-significant sample size in that topic.

4.3 Components

4.3.1 Data Collection

The data collection component is composed of two sub-parts. The first part is a scraper object unique to each supported source (see section 4.2.1) that defines two functions unique to each source:

- A function to return a list of URLs pertaining to news articles related to a specified query term. If there are multiple pages, the scraper needs to parse how many search pages exist for the query from the first page, and request each search page.
- A function that, given a URL pertaining to an article, parses the article to return the article text and metadata: headline, publication date/time, and publisher.

These two functions require separate implementations for each source, as each website has a different HTML structure (which tends to be consistent within the same source). Thus, the implementations required to parse a list of URLs from the search page and parse text from the article page are different for each source. The scraper object creates a wrapper for the implementations of these two functions to be injected into the main data collection script, such that the main script will work for any source, with the source-specific implementations of these functions abstracted away.

This design allows the use of a single data collection script (the second part of this component) for multiple sources, by having the methods that require separate implementations for each source (finding articles, scraping text and metadata) injected as a dependency to the script. Furthermore, as defined in the requirements (section 3.2.1), it should be possible to re-run the program at a later date to add new articles without sending web requests for articles already in the collection, such that it is possible to 'update' the collection and recover collected information in case computation is interrupted.

The data collection script first loads the existing output file for the source and category (if it exists) and loads a list of already-saved URLs. Then, for each query term in the category, it queries the search page(s) to gather a list of URLs. It then combines the URLs for all search terms in the category into one list. For each URL, if the URL is not saved already in the output file, it queries the URL for the article text and metadata, and stores it as a JSON object to an appended line in the output file. This design allows data collection to be resumed in case computation is interrupted, but with the drawback of technically violating the JSON standard for a valid JSON text to have only one top-level object per file [60].

Among all components in the final pipeline (i.e. not including sentiment-scorer comparison), data collection took the longest to compute. Results from preliminary runs of the experiments indicate that data collection took approximately 0.76 seconds per article for the Daily Express, 0.69 seconds per article for the Daily Mail, and 0.20 seconds per article for the Guardian. The full dataset of 305,185 articles took roughly a week to collect using a single general-purpose computer (Intel i7-6700HQ CPU @ 2.60GHz, NVIDIA GeForce GTX 1060 GPU) on a 200 Mbit/s (download speed) internet connection.

4.3.2 Dataset Filtering

Dataset filtering was implemented by measuring 'term frequency ranking', which measures how often does a key term (i.e. keyword or key phrase) appears in a text document relative to other terms. To compute this metric, the term frequency of every token (word) in the document has to be measured first. This is implemented using scikit-learn's [24] CountVectoriser, which tokenises a document and converts it into an array of term-frequency values for each token in the document. This array is then sorted in descending order, and then the position of the keyword in the sorted array (plus one) is recorded as the keyword rank of the document. A keyword rank of 5, for example, shows that the keyword is the 5^{th} most popular word in the document.

To use keyword rank for filtering documents, a constant threshold value is used to determine whether documents are on-topic or not. 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 the term frequency of the keyword is measured relative to other terms in the document, this metric also normalises for document length.

As in this experiment, each category can contain multiple key terms, slight modifications has to be made during pre-processing (before tokenisation). All mentions of any keyword or key term in the article are replaced by 'KEYWORD_TOKEN', and the filter measures the rank of 'KEYWORD_TOKEN' instead of any specific term, such that it doesn't matter which term the article uses. Additionally, stemming is also performed for every word in the document to reduce all words to its root forms, regardless of its word form or tense (e.g. 'disability' and 'disabilities' are both treated as the same word) using the nltk [2] library. Furthermore, CountVectoriser also ignores 'stop words', or common words in English which do not add topical information, such as 'the' and 'a'.

To evaluate the results of this filtering, the filtering component can be set to print an arbitrarily-sized sample of N articles alongside its predicted label (on-topic/off-topic) and keyword rank. A test run of 10 articles per topic indicated that for most topics, this simple metric works well in distinguishing between on-topic and off-topic articles. Manual reading of sampled articles indicated that:

- There exists a good of correlation between keyword rank and on-topic/off-topic articles, although the sample size is too low to make statistically-significant conclusions, as increasing the sample size would require manually reading more articles time-consumingly.
- In most cases, articles predicted to be 'on-topic' are on-topic, and articles predicted to be 'off-topic' are off-topic.

There were, however, topics where this filtering mechanism does not perform well. As a result, many off-topic articles remain in the filtered dataset. These topics were 'blind', 'mute', 'paralysis' ('paralysed'), and 'speech impairment' ('stutter'), where the keywords can have other meanings irrelevant to the context of people with disabilities. For example, consider these sampled sentences, taken from articles that were mislabelled on-topic:

- "When Harry met Meghan: How Prince Charles's family friend set up blind date."
- "Kate Garraway gets flustered as she struggles to mute her ringing phone live on air."
- "Snooker: Higgins stutters then stages another late comeback."

As term frequency does not distinguish between multiple meanings of words, it is unlikely that this limitation could be solved using any term frequency based approach, or a similar syntactic approach (such as tf-idf and word vectors).

Several other alternatives were considered to term-frequency ranking. The conventional approach is by calculating term frequency — inverse document frequency (tf-idf) [35], [36], which builds on term frequency by weighing more 'common' words with lower scores and vice versa. Using tf-idf, terms that appear more frequently in the corpus of all documents are deemed to be 'less important' and assigned a lower weight, and vice versa. However, this approach is unsuitable for this project, due to the selective nature of the dataset (where only articles containing certain query terms are collected). This will cause the idf weights of these query terms to be much lower than it should be, because at least one of the query terms would appear on every document in the corpus, highly skewing the document frequency of all query terms in this dataset (relative to within all Daily Mail / Guardian / Daily Express articles). For this reason, limited evaluation showed that tf-idf ranking performed worse than term-frequency ranking in this filtering task.

A possible improvement to the term-frequency ranking model is by using a supervised machine learning model to classify documents to 'relevant' and 'irrelevant' (a boolean classification problem). There are many supervised approaches that are popular for text classification, including Support Vector Matrices (SVM), Naïve Bayes (NB), and k-nearest neighbour (kNN) models [37]. However, this approach would require labelled data (i.e. a collection of articles with 'relevant' and 'irrelevant' labels for each topic).

4.3.3 Feature Extraction and Rule-based Sentence Matching

The next component parses the article text using SpaCy [22] to extract relevant information required by the sentiment scoring and statistical analysis components. Initially, the headline is prepended to the article text to create the text document. SpaCy performs a syntactic analysis of the document, returning a list of 'enhanced' tokens (words) which contain additional information parsed by SpaCy, such as part-of-speech tags, syntactic parent/children, known entities, sentence start and end, and root word (lemma). The component iterates over this list of 'enhanced' tokens to extract information that would be useful for statistical analysis:

1. Sentences containing a word relating to a keyword or key phrase (using sentence matching), and the number of keywords / key phrases in each sentence.
2. The frequency of the token or sequence of tokens, for each token that has the same lemma as a keyword, or a sequence of tokens that has the same sequence of lemmas as a key phrase.
3. Number of relevant sentences in the document (number of occurrences of point 1).
4. Total number of sentences in the document.
5. Number of keyword occurrences in the document (number of occurrences of point 2).
6. Total number of tokens in the document.

Points 3-6 were used to measure a relevance score of each article, however that score is currently unused by the statistical analysis component. Future work could expand on this concept.

The sentiment scoring component requires all sentences relating to a keyword or key phrase to be extracted from each article. SpaCy provides a rule-based Matcher tool that returns the indices of tokens (or sequences of tokens) that fulfils a specfic definition ('rule'). This component

uses the Matcher tool to find all indices of tokens that has the same lemma as a keyword, or sequences of tokens that has the same sequence of lemmas as a key phrase. Then, for each token, the component retrieves the full sentence that contains the token index and stores them into a list of relevant sentences, along with the number of keywords / key phrases in each sentence. These relevant sentences are stored in an array of JSON objects, with each JSON object representing one sentence, such as:

```
"Don't be disabled in spirit as well as physically.\"": {
        "keyword_count": 1
}
```

This array is stored as a value within the parent article's JSON object.

Results from running the experiment indicate that this component took approximately 0.37 seconds per article, using a single general-purpose computer (Intel i7-6700HQ CPU @ 2.60GHz, NVIDIA GeForce GTX 1060 GPU).

4.3.4 Sentiment Scoring

The sentiment scoring component implements open-source libraries to measure the 'sentiment' (ideally, perceived view of the sentence towards a disability, disabilities, a person with disabilit(y/ies), or people with disabilit(y/ies) as referred by the keyword or key phrase) of news articles. Sentiment scores are real-valued scores (i.e. a score of 0.0 indicates that a sentence is neutral, a highly positive or highly negative score indicates the sentence has strong positive or negative opinions), capped between -1.0 and 1.0.

Two implementations of the component were developed:

- The first iteration of the sentiment scorer component is used to perform a comparison between several open-source sentiment analysis implementations for this sentiment scoring task. This iteration computes sentiment scores of all implemented scorers for every article, and is ran only on an arbitrarily-sized sample of the full dataset, as running all sentiment scorers without optimisation takes roughly ~1 minute per article which is infeasible for the full dataset.
- The second iteration of the sentiment scorer component is used in the final pipeline. This iteration only computes one sentiment score per each article, and is highly optimised. Two iterations of this component has been tried: one implementing OpenAI's [44] model, and another implementing VADER [8]. In both cases, the runtime of is lower than 0.2 seconds per article.

4.3.4.1 Comparison of Open Source Implementations

This component consist of two scripts. The first script is the sampler, which loads a small sample of articles For each topic and source in the parsed dataset. It then loads all relevant sentences from the sample of articles to a combined list of relevant sentences from each source and topic. From these sentences, it selects an arbitrarily-sized sample of relevant sentences for each source and topic (the experiment used 5 sentences per source and topic * 12 topics * 3 sources = 180 sentences).

Every sentence in the sampled are then scored using the 7 open-source sentiment analysis implementations that were explored in section 2.3.3:

- VADER [8],
- xiaohan2012's 'twitter-sent-dnn' repository [41],
- kevincobain2000's 'sentiment_classifier' repository [43],
- OpenAI's 'generating-reviews-discovering-sentiment' repository [44],
- Stanford CoreNLP [23], using the stanfordcorenlp package [46] was used to start and query a Stanford CoreNLP local server in Python,
- TextBlob's PatternAnalyzer [47],
- TextBlob's NaiveBayesAnalyzer [47].

The scores of every sentence in the sample (of 180 sentences in the experiment) is stored in a JSON object similar to:


```
{
        "sentence": "The autism gender trap.",
        "label": "-",
        "sentiment_score_openai": -0.24175840616226196,
        "sentiment_score_vader": -0.3182,
        "sentiment_score_xiaohan": -0.9157504061450769,
        "sentiment_score_kcobain": -0.5,
        "sentiment_score_stanford": -0.5,
        "sentiment_score_textblob": 0.0,
        "sentiment_score_textblob_bayes": -0.9139802175212899
}
```


The JSON objects of the sample is then output to a text file, where a user can manually change the 'label' fields to either '+' (positive), '-' (negative), 'n' (neutral), or 'o' (irrelevant/off-topic) to be read by the second script. To ensure there is no bias in manual labelling, a second output file which contains only sentences and labels (without sentiment scores) were used.

The second script is the analyser. Given a dataset of labels and sentiment scores (with the format shown above), the analyser plots sentiment score distributions of positive, negative, and neutral sentences in seven histograms (one for each sentiment scorer). Additionally, it also computes these statistics for each sentiment scorer:

- Mean positive: mean sentiment score for all positive-labelled sentences
- Mean neutral: mean sentiment score for all neutral-labelled sentences
- Mean negative: mean sentiment score for all negative-labelled sentences
- True positive: count of positive-labelled sentences with sentiment score > 0.0
- False positive: count of negative-labelled sentences with sentiment score > 0.0
- True negative: count of negative-labelled sentences with sentiment score ≤ 0.0
- False negative: count of positive-labelled sentences with sentiment score ≤ 0.0
- Accuracy: (True positive + false positive) / (count of all positive or negative sentences)

The results of this sentiment scorer comparison is documented in section 5.2.

4.3.4.2 Final Implementation

The results of the sentiment scorer comparison, shown in section 5.2, show VADER [8] and OpenAI's [44] models as the two best sentiment scoring tools for this domain of measuring the perceived 'sentiment' of a sentence towards a disability, disabilities, a person with disabilit(y/ies), or people with disabilit(y/ies).

Unlike the iteration used for sentiment scorer comparison, this iteration only computes one sentiment score per each article, and is highly optimised. Two versions of this iteration were developed: one implementing OpenAI's model, and another implementing VADER, to measure sentiment scores. In the final pipeline used for the experiment, only the version using VADER was used, as VADER's scores was shown to be better at displaying trends separating different subsets than the OpenAI model's scores (refer to section 5.2).

Within the pipeline, this component's task is to compute sentiment score information for every relevant sentence, and append that information on each sentence's JSON object:

```
"Don't be disabled in spirit as well as physically.\"": {
        "keyword_count": 1,
        "sentiment_score": 0.4215
}
```

Once the scores of all sentences has been measured, this component also computes the sentiment score of each article, which is defined as the weighted average of sentiment scores for all relevant sentences contained within the article:

$$Article's\ sentiment\ score = \frac{\Sigma(sentiment\ score * number\ of\ key\ terms) for\ each\ sentence}{Total\ number\ of\ keyword\ occurrences\ in\ the\ article}$$

This iteration is highly optimised for runtime: it only computes one sentiment score for each sentence, using one sentiment model, instead of loading and analysing every sentence with all seven sentiment models as in section 5.2. For the version that implements OpenAI's model, it is further optimised by 'batching' the call to the sentiment model: instead of calling 'openai_model.transform()' for every sentence, it builds a corpus (list) of all sentences in all articles from the loaded file (where each file represents all articles from a source for a topic) and calls 'openai_model.transform()' only once on the full corpus. Results from running the experiment indicate that sentiment scoring took approximately 0.16 seconds per article using the OpenAI model, and 0.0031 seconds using VADER, using a single general-purpose computer (Intel i7-6700HQ CPU @ 2.60GHz, NVIDIA GeForce GTX 1060 GPU).

4.3.5 Statistical Analysis and Visualisation

This component is responsible for statistical analysis and data visualisation, plotting graphs and saving statistical information for each topic. At this point, each article's JSON representation includes information about its source, publication date, sentiment score, and other information extracted by previous components. However, up to this point, the dataset of news articles belonging to each source and each topic were kept in separate files. Instead of loading just one file per execution as in previous components, this component had to load all files that relates to a specified topic from every source, and combines the datasets of all articles from every source that relates to

the specified topic. This is necessary to compare different sources in a plot and perform statistical comparisons. (Refer to the diagram in section 4.1 for more details on the pipeline design for each topic.)

Most of this component is concerned with plotting and measuring how sentiment score varies when the publication source and date/year of publication is varied. For easier data processing, it loads the publication date, sentiment score, and source information of each article as rows in a numpy array matrix, sorted by publication time.

The component then uses the matrix to produce the following plots using matplotlib [25]:

- A scatter plot of publication date (X) vs sentiment (Y), coloured based on source (Z).
- A regular histogram showing the number of news articles in the dataset for each year.
- Two-dimensional histograms showing:
  - The number of news articles for each year (X) and source (Y) in the dataset.
  - The number of news articles for each year (X) and 'sentiment range' (Y) in the dataset. ('sentiment range': news articles are grouped together based on sentiment score with intervals of 0.1; thus, for example, -0.1–0.0, 0.0–0.1, and 0.5–0.6 are sentiment ranges)
  - The number of news articles for each source (X) and 'sentiment range' (Y) in the dataset.
- A line graph of the moving average of sentiment score (Y) over time (X), with separate lines for each source. The moving average is defined as the mean of sentiment scores for W previous articles up to the current article, where W is the moving average window size. In this experiment, W = (no. of articles in the topic) / 10, with a lower cap of 50 and an upper cap of 500.
- A violin plot [51] and box-and-whiskers plot [50] showing the sentiment score distributions (including mean, upper and lower quartiles, and density plots) for each data subset, separated by source.
- A violin plot showing the sentiment score distributions for each data subset separated by source and publication year, with intervals of 2 years, separated to two plots: one for 2000–2009 and one for 2010–2019.

These plots are arranged in a 3x3 grid and saved to an output file, unique to each topic. Refer to section 2.3.4 for a further description of each plots' usage.

To quantify the distribution of each data subset, the component computes the mean, standard deviation, and total count of articles for the full dataset and each possible subset of the data (i.e. a set of all articles for every source, a set of all articles for every year, and a set of all articles for every year and every source). These metrics are saved to an output text file, unique to each topic.

It also attempts to compare whether the sentiment scores of articles published by a source is significantly higher than the sentiment scores of articles published by another source. The Mann-Whitney U Test [52] is a non-parametric statistical test that measures the null hypothesis that "given a randomly-selected value from a distribution, and another randomly-selected value from another distribution, the first value is equally likely to be less or greater than the second value." If the null hypothesis holds true, then there are no statistically significant difference between the two distributions. The Mann-Whitney U Test returns the U statistic and a p-value between 0 and 1, which corresponds to how likely the null hypothesis is to be true; the null hypothesis is rejected if the p-value is lower than 0.05. This test is performed to compare the subset of all

articles published by each source to the subset of all articles published by each other source; for all articles within a topic, and for all articles published in each year within a topic. These test results are saved to the same output text file.

As well as information relating to sentiment score, this component also attempts to plot trends regarding the usage of key terms (used to describe disabilities or people with disabilities) over time. This information is collected from the frequencies of tokens with the same lemma as a key term, as mentioned in point 2 of section 4.3.3. The tokens are first stemmed with a custom stemmer: unlike NLTK's stemmer or SpaCy's lemmatiser as used in previous components, this stemmer only removes plurals (-s, -es) and tenses (-ed, -ing), but does not fully reduce words to its base word form (e.g. 'illness' and 'illnesses' are equivalent, but 'mental' and 'mentally' are still separate words). Then, the dataset is split into smaller subsets based on publication year and source. For each subset, the term frequencies for all articles within the subset are averaged, to compute a measure of average term occurrence (per article):

$$Average\ term\ occurrence = \frac{\Sigma(term\ frequency\ in\ article)\ for\ each\ article}{number\ of\ articles\ in\ subset}$$

For each term, the annual average term occurrences are plotted in a line graph of average term occurrences (Y) over publication year (X). Each unique term has its own plot, with separate lines for each source. These plots are then saved to an output file, separate from the sentiment score plots, also unique to each topic.

Chapter 5

Results Evaluation

5.1 Focused Topic

Although the pipeline was run on a list of twelve disability-related topics (as defined in section 4.2.2), this results evaluation will mainly focus on the 'disabled' topic. The 'disabled' topic consist of articles that relates to the key terms: 'disabled', 'disability', 'handicapped', 'cripple', 'invalid', 'accessible', 'ablism', 'ableism'. This topic was chosen as it had the largest sample of news articles within the dataset (see also: section 4.2.3), and it is the most generalisable topic (as it refers to the general theme of disabilities and people with disabilities, rather than a specific topic). This focus on one topic for the whole of Chapter 5 helps keep the numbers and results being discussed consistent throughout Chapter 5. Furthermore, the full result plots for all topics will be available in the Appendix.

At several points in this chapter, results from other topics will also be discussed where it would add to the discussion. Information from other topics will be explicitly mentioned (e.g. "For topics other than 'disabled',") such that the reader understands where the data does not refer to the 'disabled' topic.

Within the 'disabled' topic, the (post-filtering) sample size that was obtained for each year between 2000 and 2018 is as follows:

Year	Daily Express	Daily Mail	Guardian	Total
2000	0	0	504*	504
2001	0	0	294	294
2002	0	0	334	334
2003	0	2	330	332
2004	0	2	387	389
2005	0	0	381	381
2006	0	0	338	338
2007	51	0	424	475
2008	86	0	424	510
2009	175	0	352	527
2010	157	173	365	695

2011	166	370	607	1,143
2012	238	516	822	1,576
2013	133	473	638	1,244
2014	140	846	544	1,530
2015	177	1,095	526	1,798
2016	260	1,128	640	2,028
2017	220	1,094	506	1,820
2018**	49	336	108	493
Total	1,852	6,035	8,524	16,411

* 2000 data also includes a small amount of articles published before the year 2000.

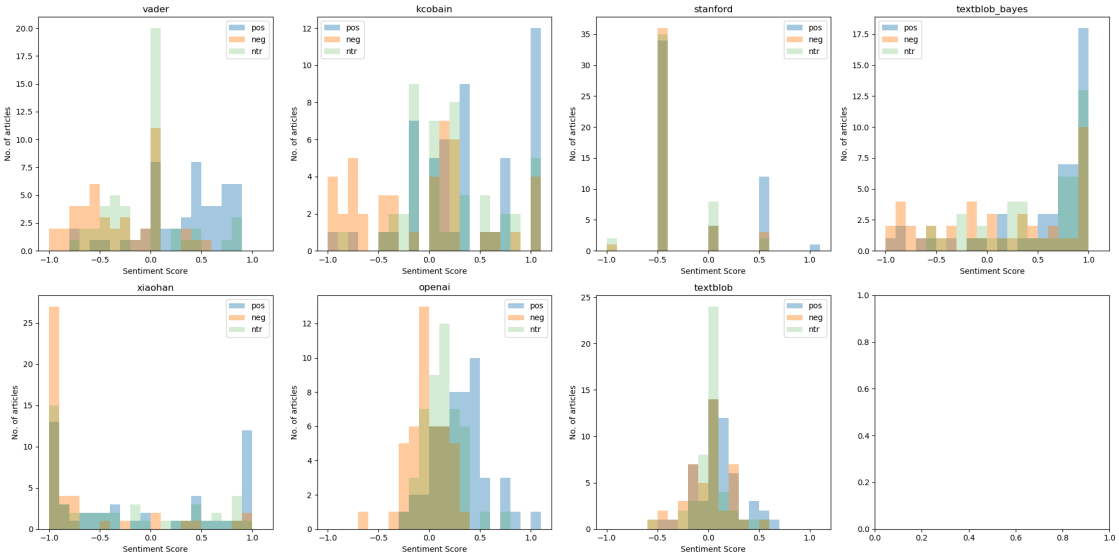
** 2018 data is incomplete and would only include articles up to (approximately) end of March.

5.2 Comparison of Sentiment Scorers

For this comparison, an evenly-distributed sample of relevant sentences (i.e. sentences referring to disabilities or people with disabilities) were taken from the filtered dataset of every topic. The sample contains 5 sentences from each source (Daily Express, Daily Mail, Guardian) and each of the 12 topics (as defined in section 4.2.2). The sample contains 180 sentences in total.

Each sentence in the sample are then manually labelled to: positive, negative, neutral, or irrelevant. These labels are measured by the perceived 'sentiment' of a sentence towards a disability, disabilities, a person with disabilit(y/ies), or people with disabilit(y/ies), or 'irrelevant' if it did not refer to any disability-related topic.

The sentiment score distributions of positive (blue), negative (red), and neutral (green) sentences for each topic were plotted as follows:



From these plots, it is clear that 'vader' [8] and 'openai' [44] are the two most encouraging sentiment scorers on this domain, as they show a clear distinction between the distributions of 'positive' and 'negative' labels (although with some overlap near the centre), while other scorers

produce plots where the values are all over the place.

The means of these distributions, the count of 'positive'-labelled sentences with a positive (>0, true positive) and negative (≤0, false negative) sentiment scores, the count of 'negative'-labelled sentences with a positive (>0, false positive) and negative (≤0, true negative) sentiment scores, and accuracy (defined in section 4.3.4.1 as (true positive + true negative) / (all positive + all negative)) were also measured. Refer to section 4.3.4.1 for a formal definition of these metrics.

The values of these metrics for each sentiment scorer are shown below:

Implementation	Mean Positive	Mean Neutral	Mean Negative	Accuracy*
VADER [8]	0.327	-0.048	-0.313	0.789
XiaoHan [41]	-0.071	-0.306	-0.681	0.621
Kevin Cobain's [43]	0.348	0.210	-0.116	0.621
OpenAI's [44]	0.301	0.128	-0.025	0.758
Stanford CoreNLP [23]	-0.196	-0.394	-0.398	0.568
TextBlob [47] (Pattern)	0.086	0.007	-0.030	0.663
TextBlob [47] (Naïve Bayes)	0.572	0.508	0.123	0.653

*Binary classification accuracy (accuracy of scores for 'positive' and 'negative' labels in the sample, disregarding 'neutral' or 'irrelevant' labels)

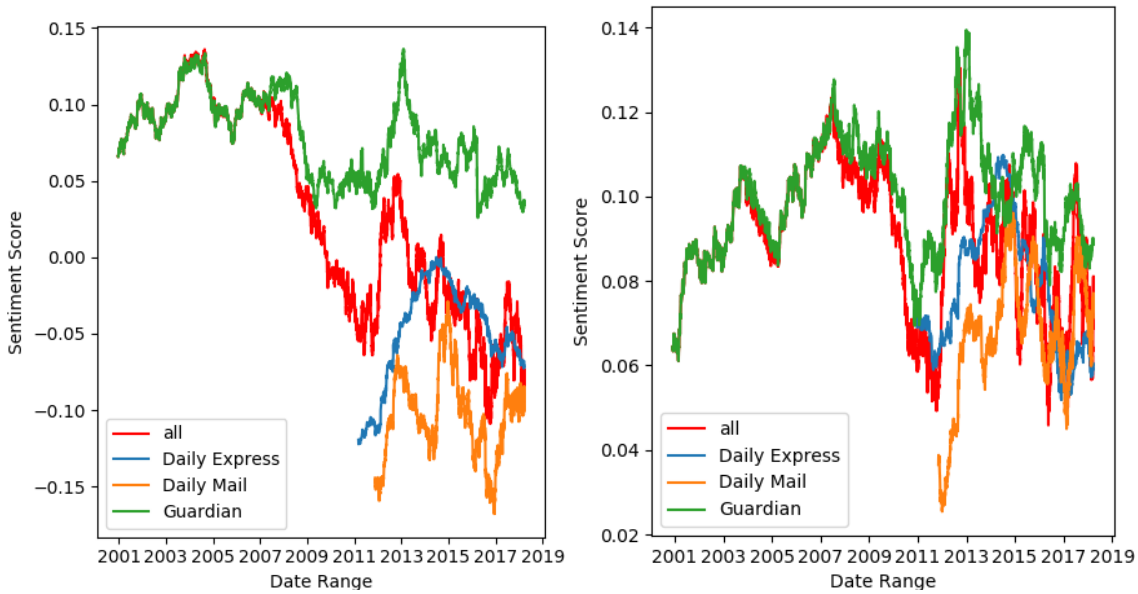
'neutral' or 'irrelevant' labels were disregarded in the accuracy measurement as they do not have an expected value (whereas 'positive' is accurate if value >0, and 'negative' is accurate if value ≤0) For the following measurements, 'neutral' and 'irrelevant' labels were also disregarded:

Implementation	True Positive	False Positive	True Negative	False Negative
VADER [8]	35	4	40	16
XiaoHan [41]	22	7	37	29
Kevin Cobain's [43]	35	20	24	16
OpenAI's [44]	46	18	26	5
Stanford CoreNLP [23]	13	3	41	38
TextBlob [47] (Pattern)	32	13	31	19
TextBlob [47] (Naïve Bayes)	43	25	19	8

These results from the 180-article sample indicate that VADER [8], followed by OpenAI's model [44], as the two best-performing open-source sentiment scoring tools for this domain, as shown by the accuracy metric. Although these results are not conclusive (given the small sample size of sentences, as it was necessary to manually label each sentence in the sample), it is a sufficient indicator of which sentiment scorers would perform better in predicting correct labels, and therefore should be chosen for the final pipeline and experiment. (Additionally, it shows

that the sentiment score distribution generated by OpenAI is slightly skewed positive, while the distribution generated by VADER is slightly skewed negative, based on the mean values and ratio between false positives and false negatives).

To prove whether these accuracy metrics are relevant for the experiment, both VADER and OpenAI's sentiment scorer implementations were implemented in the final pipeline. Then, the 500-article moving average sentiment score of both scorers were plotted for the topic 'disabled':



(Left = VADER, Right = OpenAI's model)

These plots show that the version implementing VADER is more consistent in distinguishing between different sources, with a smoother trend line; while the line produced by the version implementing OpenAI's model has higher randomness; despite using the same moving average window size (500 articles) in both plots.

These results came as a surprise, due to the simplicity of VADER's rule-based model, relative to other implemented models (refer to section 2.3.3 or 4.3.4.1 for a list and short description of these sentiment scorer implementations) based on supervised machine learning or neural network approaches. The suspected reason behind this is that because these supervised machine learning and neural network models were trained on data from other domains (mainly tweets, movie reviews, or Amazon reviews), and the parameters learned by the model does not necessarily translate well to this domain (sentences from news articles referring to disabilities or people with disabilities). Meanwhile, VADER's simpler rule-based model is more generalisable, as it simply checks for a set of rules to determine general positive/negative text (instead of using parameters learned from training data).

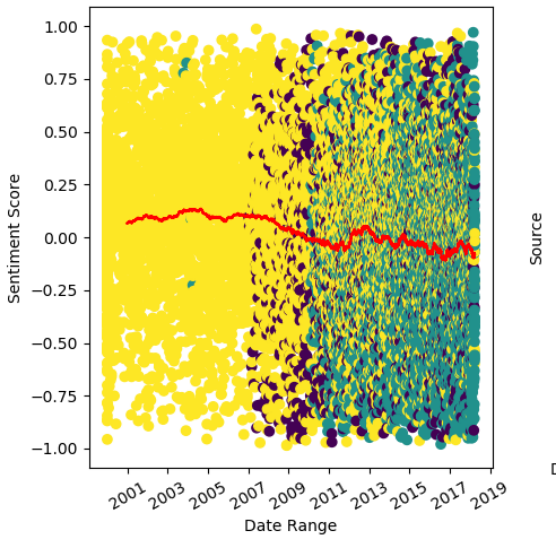
It is likely that a supervised machine learning or neural network model, trained on an adequately large labelled dataset (likely at least tens or hundreds of thousands of sentences) from this domain, could strongly out-perform VADER, given that the accuracy of sentiment scores from OpenAI's implementation (based on a neural network model) was very close to VADER's, despite the model being trained on a different domain (82 million Amazon product reviews). However, such a large labelled dataset was infeasible for the scope of this experiment.

For the following sections, all sentiment score results mentioned are those scored by VADER's

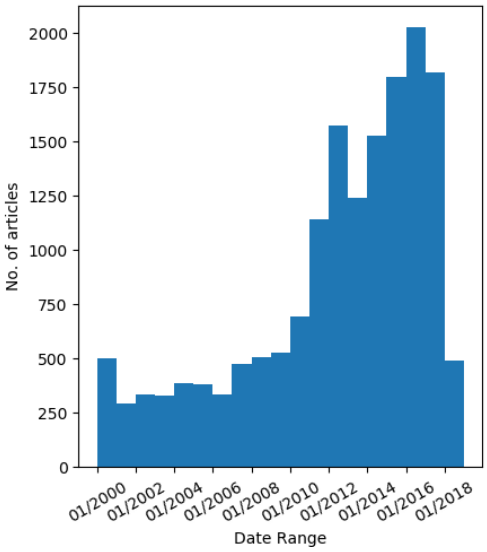
implementation [8]. Additionally, news articles with a VADER sentiment score of 0.0 were excluded from the plotted dataset, as they are presumed to be non-opinionated and thus irrelevant in sentiment measurement.

5.3 Sentiment Score: Plots and Trends

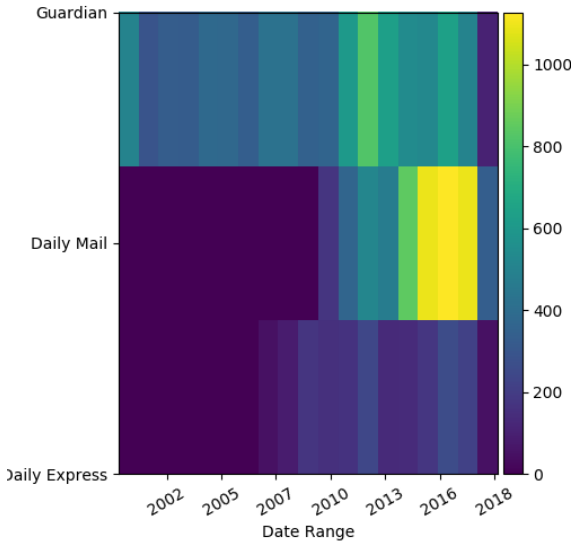
For the 'disabled' topic, the VADER sentiment scores of all articles within the dataset (16,411 articles in total within the topic) were plotted with regards to their publication date and source of article (Daily Express, Daily Mail, Guardian) in several visual representations (as defined in section 4.3.5).

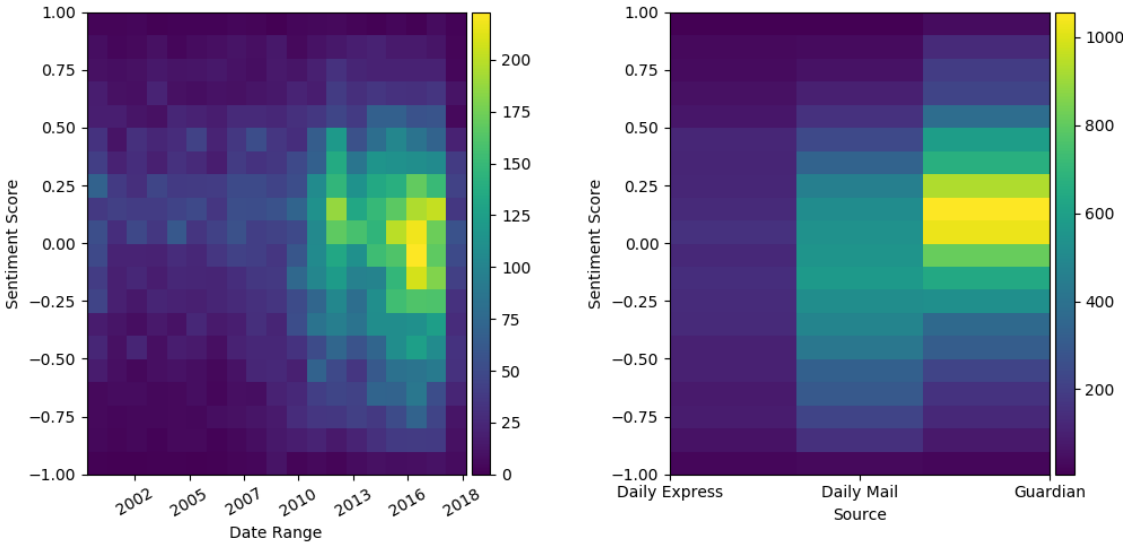


The first plot is a scatter plot of publication date (X) vs sentiment scores (Y), coloured based on publication source (Z). Due to the large dataset size, the scatter plot does not provide much information with regards to distributions and trends, although it shows how the Daily Express (blue-green) only retains articles from after ~2007, the Daily Mail (dark blue) only retains articles from after ~2010, and the Guardian (yellow) only retains articles from after ~2000, as mentioned in section 4.2.4. A moving average line (with a window size of 500) of sentiment scores over time of all articles (regardless of source) is drawn over the scatter plot.



A histogram of the number of articles in each year was plotted to show the distribution of articles over time in the dataset. This do not necessarily reflect how many articles was published regarding the topic for every year, as news publishers often do not retain all historical articles and the amount of retained articles tend to be higher in more recent years, and vice versa. That said, a spike in the number of articles in the dataset was visible in 2012, which correspond to increased media coverage of disabilities and people with disabilities around the 2012 London Paralympics.





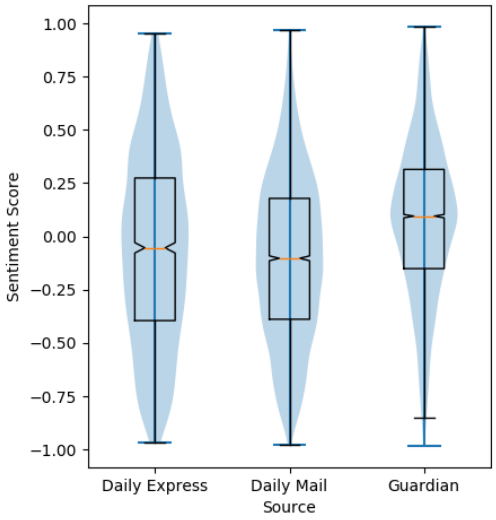
Three two-dimensional histograms that show the distribution of articles for:

- The number of news articles for each year (X) and source (Y) in the dataset.
- The distribution of sentiment scores (Y) for each year (X).
- The distribution of sentiment scores (Y) for each source (X).

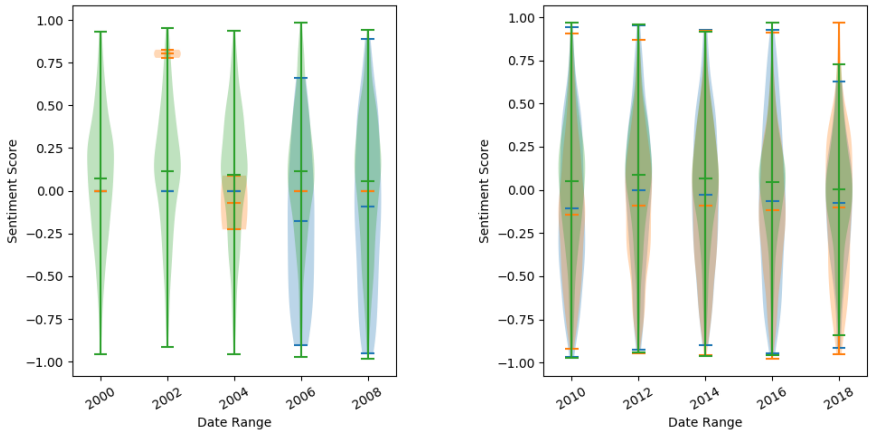
were plotted. The year-source plot show that the dataset consist almost entirely of Guardian articles for the years 2000–2006. In 2007 Daily Express articles and in 2010 Daily Mail articles starts to appear, although Guardian articles still predominate between 2007–2013, until in 2014 where Daily Mail articles start to outnumber Guardian articles by roughly 2:1.

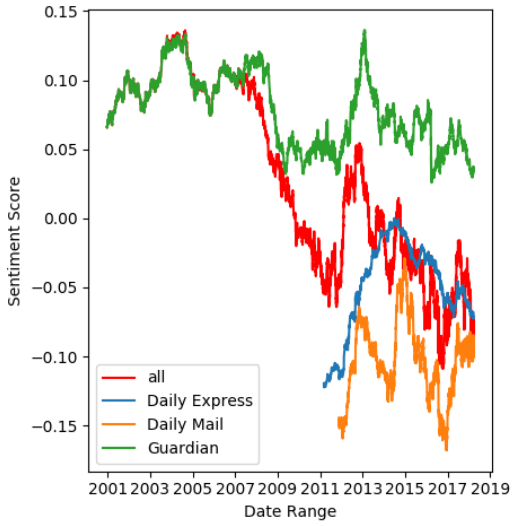
The year-sentiment plot show that the sentiment distribution of most news articles lie somewhere between 0.0 and 0.3 from 2000–2009, and between -0.3 and 0.3 (but with more variance/outliers) from 2010–2018. It also showed a slight 'drop' in the sentiment distribution (or an increase of news articles with negative sentiment scores) around 2016.

The source-sentiment plot show (roughly) that the Guardian has a higher mean and less variance than the Daily Mail, and the Daily Express is barely visible due to the lower sample size of Daily Express articles in the dataset. However, the uneven sample size (i.e. more Guardian articles vs Daily Mail or Daily Express articles over the full dataset) makes this visual representation hard to compare. The violin [51] and box-and-whiskers plot [50] show a better representation of this source-sentiment data:



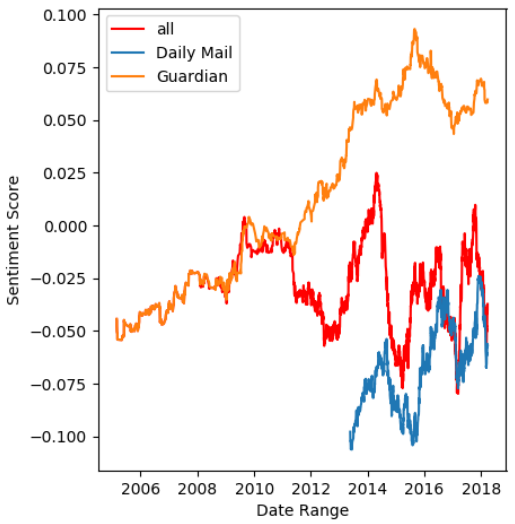
The violin and box-and-whiskers plot show that the Guardian's mean sentiment score (0.074) is higher than the Daily Mail's (-0.104) or the Daily Express's (-0.060). Furthermore, the violin plot component show that the Guardian's distribution is more 'compact' around the mean (std. dev. = 0.363), the Mail has a slightly higher variance (std. dev. = 0.386), and the Express has the highest variance (std. dev. = 0.447). A version that shows violin plots for each 2-year period were also plotted (green = Guardian, blue = Daily Express, orange = Daily Mail):





From the year-sentiment plot showed that there is an apparent trend of declining sentiment scores over time. However, the moving average line plot (where each data point shows the mean sentiment score of 500 consecutive articles, and the last article's date) show that this is not necessarily the case. When only taking into account articles from the same source, the moving average sentiment score stays roughly consistent. However, the decreasing trend in 'all' is likely better attributed to the (gradually) decreasing proportion of Guardian articles and the (gradually) increasing proportion of Daily Mail articles in the dataset.

For topics other than 'disabled', the majority show a similar constant trend for articles within the same source, although this is not always the case. The topic 'autism', for instance, show a positive trend in sentiment scores year-on-year:



Note also that the Daily Express is not plotted for this topic, due to the sample size being too small (128 articles), lower than the moving average window (316 articles on this topic); a common occurrence for most other topics (as defined in 4.2.2).

5.4 Sentiment Score: Statistical Comparison of Sources

To test whether the differences between the distributions of each source are statistically significant, the Mann-Whitney U Test [52] was invoked. The Mann-Whitney U Test is a mathematical function that tests the null hypothesis that given a randomly-selected value from a distribution, and a second randomly-selected value from a second distribution, the first value is equally likely to be less than or greater than the second value. If the null hypothesis holds true, then there is no statistically significant difference between the two distributions. The Mann-Whitney U Test function returns a p-value score, which is a measure of the probability that the null hypothesis is correct. In this experiment, two distributions are considered significantly different if the p-value returned by the Mann-Whitney U Test is lower than 0.05.

Below is a table of p-values from Mann-Whitney U Test results for every source combination in the dataset, for every topic (including topics other than 'disabled'):

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
Disabled	**$2.66 * 10^{-167}$**	**$2.29 * 10^{-35}$**	**0.000108**
Autism	**$3.00 * 10^{-12}$**	0.137	0.184
Blind	0.335 **	**$1.62 * 10^{-9}$**	**$6.57 * 10^{-9}$ ****
Cerebral Palsy	0.469	0.156 **	0.169
Deaf	**$6.30 * 10^{-10}$**	0.141	0.0833
Developmental Delay	0.383	**0.00179**	**0.00326 ****
Dyslexia	0.0740	0.362	0.464 **
Epilepsy	**$3.39 * 10^{-5}$**	0.0579	0.322 **
Mental Illness	**$3.54 * 10^{-76}$**	**$1.51 * 10^{-9}$**	0.102 **
Mute	**$6.78 * 10^{-5}$**	**0.0311**	0.346 **
Paralysis	**$2.73 * 10^{-5}$**	**0.00909**	0.0827 **
Speech Impairment	0.235	**0.0200**	0.0546 **

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

This data shows that a statistically-significant distinction can be proven between the sentiment score distributions of two sources for 17 / 36 of cases. (Cases where there exists a distinction between the two distributions are highlighted in bold) In particular, Guardian > Daily Mail is true for 7 / 12 cases, Guardian > Daily Express is true for 7 / 12 cases, Daily Express > Daily Mail is true for 1 / 12 cases, and Daily Mail > Daily Express is true for 2 / 12 cases. Comparing these values to the size of each dataset (section 4.2.3), Lower p-values tends to correlate well with larger sample sizes (i.e. the higher the sample size of both distributions, the higher the chance that there exist a statistically-significant difference).

For the 'disabled' topic, the Mann-Whitney U Test was also performed to compare between Daily Mail, Daily Express, and The Guardian for each year's subset between 2007 and 2018 (Before 2007, there were not enough non-Guardian articles in the dataset to make a comparison). The p-values of these comparisons are shown below:

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
2007	N/A	**5.24 * 10^{-6}**	N/A
2008	N/A	0.0739	N/A
2009	N/A	**2.24 * 10^{-5}**	N/A
2010	**6.00 * 10^{-7}**	**6.89 * 10^{-6}**	0.360 **
2011	**5.73 * 10^{-16}**	**9.73 * 10^{-5}**	**0.0355**
2012	**4.44 * 10^{-17}**	**0.00371**	**0.000739**
2013	**5.03 * 10^{-16}**	**0.00351**	0.0827
2014	**1.27 * 10^{-9}**	**0.00208**	0.321
2015	**6.87 * 10^{-17}**	**0.00855**	**0.00718**
2016	**4.63 * 10^{-26}**	**4.74 * 10^{-6}**	0.0576
2017	**3.39 * 10^{-13}**	**0.00244**	0.0867
2018*	0.122	0.120	0.328

* 2018 data is incomplete and would only include articles up to (approximately) end of March.

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

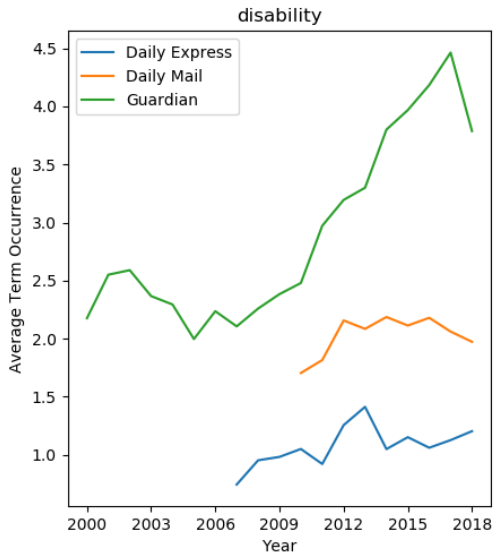
5.5 Key Terms: Plots and Trends

Apart from sentiment score analyses, NLP approaches can also be applied to extract various other syntax-based features and analyse trends based on them. Features based on counting words (term frequency) are a mainstay of NLP research. Besides sentiment scores, the pipeline also extracts the term frequency of all terms that matches a key term's lemma from each news article (refer to section 4.3.3 for details). This information is used to plot trends on keyword usage over time.

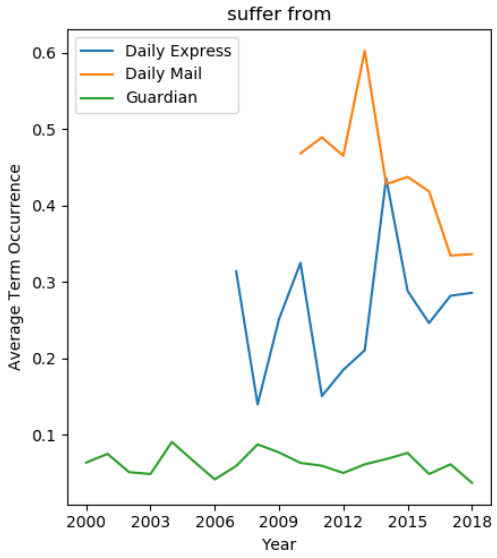
A weak custom stemmer that only stems plural and past/future tense forms (but not other suffixes) allows the grouping of equivalent terms together regardless of context, but without losing additional meaning from affixes. For example, 'illness' and 'illnesses' are equivalent, but 'ill' and 'illness' are still separate words.

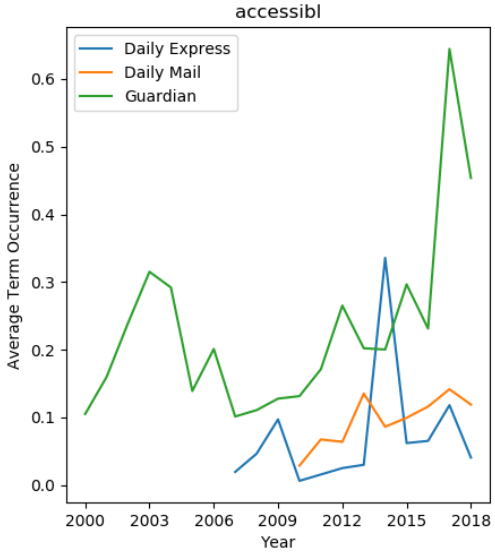
For each term, a measure of average term occurrence, which measures the expected number of occurrences of a term in an article for a given year, is computed for each year between 2000 and 2018. Refer to section 4.3.5 for a formal definition of this metric.

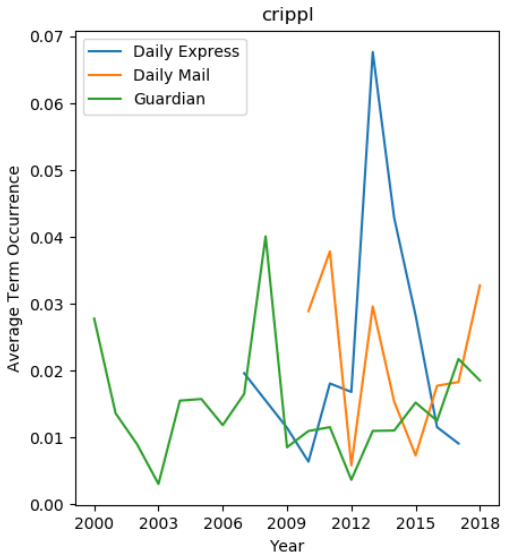
With this approach, we measured the year-on-year average term occurrence trends of each key term in the 'disabled' topic, and plotted the results in line graphs (one for each key term):

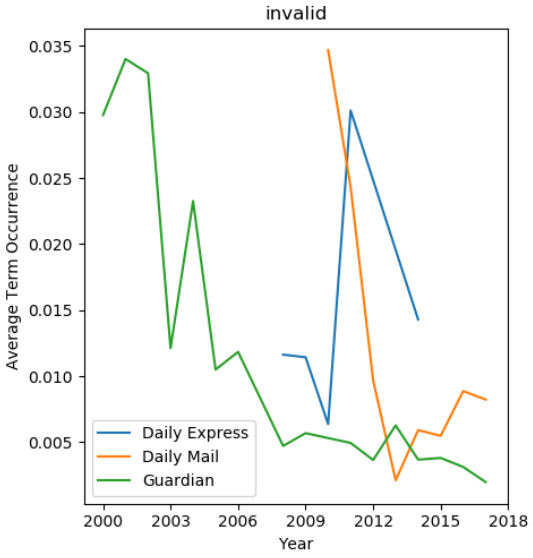

disability
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year

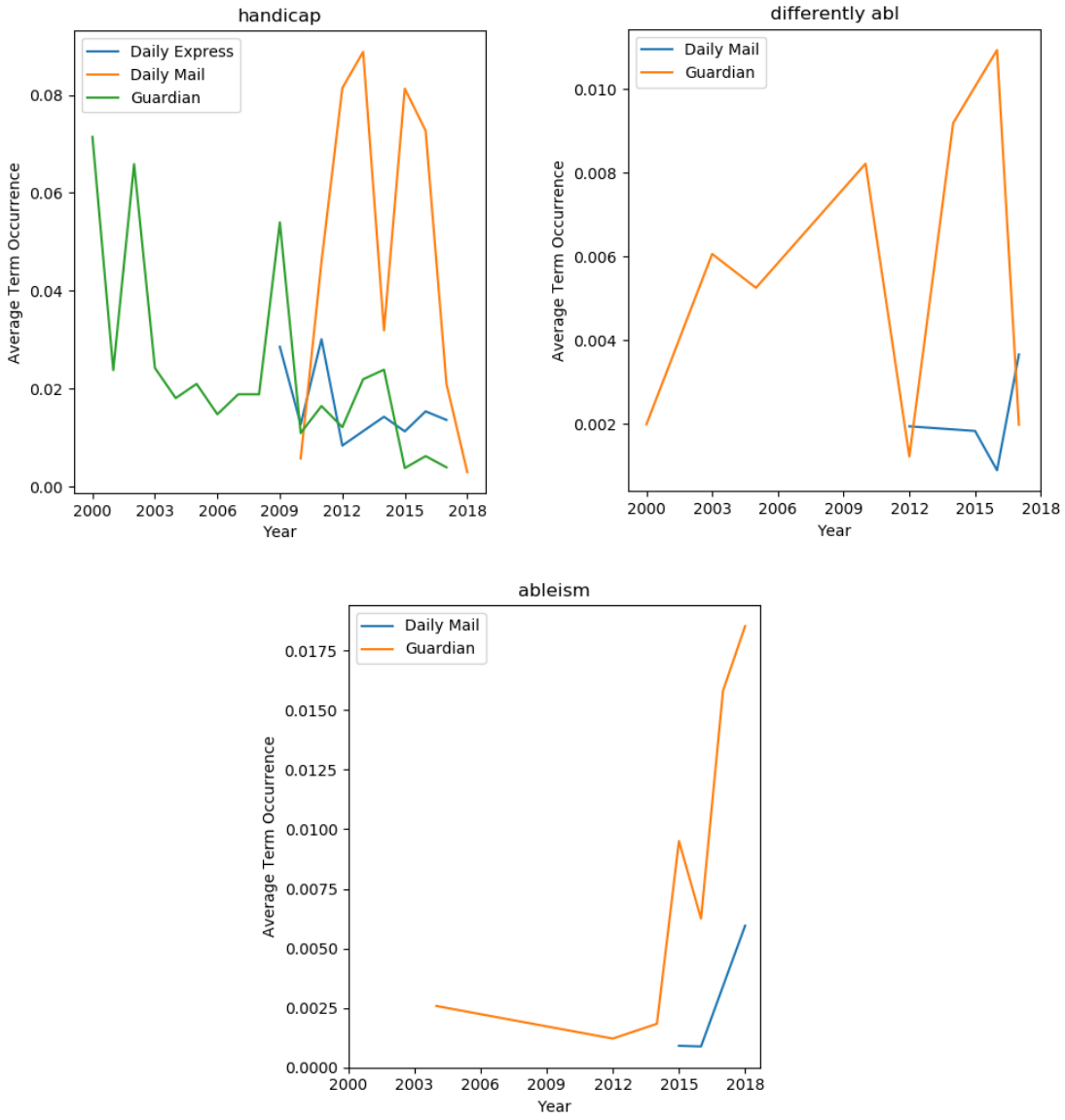

disabl
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year


suffer from
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year


accessibl
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year


crippl
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year


invalid
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year



These line plots show that, for most key terms, the average term occurrence trends mostly stay constant year-on-year for the same publisher. While this exact approach has been successfully used to detect cultural changes in British media [5], in this case the sampled time period is much shorter for significant linguistic changes to have occurred (18 vs 150 years).

In this case, increasing/decreasing year-on-year trends are only visible for a select few terms. The usage of the terms 'invalid' and 'handicap(ped)', for example, show a rapidly decreasing trend between 2000 and 2018 on the Guardian. 'accessible', on the other hand, show an increasing trend on the Guardian. While the data is less consistent for the Daily Express and Daily Mail, this was likely caused by a lack of data for these sources before 2007 and 2010 respectively, and a smaller overall sample size (especially for the Daily Express).

These plots also show variations in term usage between different publishers. For example, the Guardian refers to 'disability' or 'disabl(ed)' by name consistently more often than the Daily Mail or Daily Express, and uses the terms 'suffers(s) from' consistently less often.

Chapter 6

Conclusions

6.1 Achievements

The aim of this project was to explore the feasibility of exploiting NLP technologies to discover trends with regards to specific topics, with regards to the representation of disabilities and people with disabilities in British online news media. The results to this experiment showed that this is feasible. By analysing a dataset of 16,411 news articles related to the key terms 'disabled', 'disability', 'handicapped', 'cripple', 'invalid', 'accessible', 'ablism', and 'ableism'; the results in section 5.3 plotted trends in the variation of modelled sentiment scores across three different news publishers (Daily Express, Daily Mail, and Guardian) and over time. The results of Mann-Whitney U statistical test in section 5.4 showed that the differences in sentiment score distributions between the three publishers are statistically significant for the 'disabled' topic, and showed that 'Guardian > Daily Mail' and 'Guardian > Daily Express' is true for every year between 2010 and 2017. Furthermore, the results of analysing average term occurrences of key terms, as shown in section 5.5, identified increasing or decreasing trends for the terms 'invalid', 'handicap(ped)', and 'accessibl(e)' for the Guardian; and showed variations in term usage/popularity between different publishers.

This experiment was repeated across 11 other topics (as defined in section 4.2.2), with varying degrees of success. The lower sample size of news articles related to other topics in the dataset, and decreased effectiveness of the filter with regards to the selection of topics with more ambiguous keywords (e.g. 'blind', 'mute', 'paralysed', and 'stutter'), were major factors in the variability of results. The full results of the experiment for all 12 topics are available in the Appendix.

The experiment also showed that it was feasible to collect and analyse a large dataset of 305,185 news articles, reduced to 48,967 articles after filtering, within a reasonable time frame using consumer-grade hardware (Intel i7-6700HQ CPU @ 2.60GHz, NVIDIA GeForce GTX 1060 GPU, 200 Mbit/s download speed). Additionally, the vast majority of the time was spent on data collection from online sources, and an analysis of existing news articles corpora would take less time (approximately 0.37 seconds per article in extracting text features, with negligible sentiment scoring runtime using VADER). With more powerful hardware available to institutions and large corporations, this approach should scale well to analyse corpora consisting of millions of news articles in reasonable time.

6.2 Evaluation

This experiment showed that it was feasible to apply existing open-source NLP technologies to discover trends on a large dataset of news articles in this domain, which achieved the primary aim of this project. The solution developed to perform this experiment delivered in performing data collection and filtering of a dataset of 305,185 news articles; and feature extraction (collecting relevant sentences and key term frequencies), sentiment scoring, statistical analysis, and data visualisation of 48,967 news articles after filtering. The modular approach to solution design, with five separate components for each sub-task (data collection, filtering, feature extraction, sentiment scoring, and data analysis/visualisation), was ideal in this research, enabling experimentation in various components (e.g. trying multiple sentiment scorer implementations and data visualisation plots) without having to change or re-run code for other components. This solution achieved the project's goals sufficiently as a proof of concept, and delivered meaningful results.

The choice of NLP techniques and technologies, extracted features, statistical metrics, and visualisation tools are sufficient, and achieved results, but leaves room for further improvement. The main limitation in this research was a lack of labelled data for both filtering on-topic/off-topic articles and sentiment scoring, which led to inconsistencies in both aspects. In particular, the filter had issues with ambiguous terms such as 'blind', 'mute', 'paralysed', and 'stutter'; a supervised filter may be able to overcome this issue by taking the word's context in the sentence into account (for example, by also evaluating other words in the sentence), and also provide better evaluation of filter accuracy. Given this constraint, the filter worked adequately well to prepare the dataset for the experiment, and limited sampling of on-topic/off-topic articles showed that the approach was sufficiently accurate for the majority of topics.

Without labelled data for sentiment scoring, the only possible options were to use generalised supervised models trained on other domains, or to use a less complex model based on rule matching. It was found that a less complex model based on rule matching (VADER [8]) outperformed advanced supervised models based on statistical classifiers or neural networks trained on different domains. However, VADER is still highly inconsistent in this domain (with an accuracy of 0.789), and although it was sufficient to discover trends in this experiment, it is likely that domain-specific supervised model would outperform it. With a more consistent sentiment scorer, it is likely that the variance of sentiment scores within subsets would be lower, and the trend lines generated in data visualisation would be more consistent.

The usage of different NLP libraries in separate components (for example, using NLTK's stemmer in filtering and SpaCy's lemmatiser in feature extraction) also left occasional inconsistencies, such as rare cases of the feature extractor not finding any relevant sentences in articles the filter has deemed on-topic.

6.3 Future Work

The results showed that this computational NLP-based approach is effective in analysing news media with regards to the media representation of disabilities and people with disabilities. This could have a substantial impact on how research will be conducted for similar studies. Past studies (such as [7], [9], [12], [13]) could be revisited using computational approaches to analyse a much

larger sample size of articles, to improve the certainty and representativeness of the conclusion and possibly identify trends by varying for independent variables such as the publisher and date published.

As mentioned above, the lack of labelled domain-specific dataset (of sentences from news articles relating to disabilities, labelled with 'positive' or 'negative') for sentiment scoring limited the sentiment scores' accuracy in this experiment. While exploring general-purpose open-source sentiment scorer implementations, it was found that a simple rule-based model outperformed the more sophisticated statistical classification or neural network models for the sentiment scoring task, as they were trained with labelled dataset from other domains. However, one of the neural network based models (OpenAI [44]) came close to the selected model (VADER [8]), despite being trained on a completely separate domain (Amazon reviews), which suggest that a domain-specific supervised model would strongly out-perform VADER given sufficient, high-quality training data. Thus, if an adequately large dataset of labelled sentences (likely >10,000 sentences would be required) for this domain could be compiled, future work could use this dataset to train a supervised model to improve the accuracy of the sentiment scorer. With a more accurate and more consistent sentiment scorer, it is likely that the variance of sentiment scorers within subsets would be lower, and it would be possible to derive clearer trends and obtain statistically-significant comparisons on subsets with lower sample sizes (e.g. smaller time intervals).

Similarly, a supervised filter, trained on a labelled dataset of on-topic/off-topic news articles, would improve filtering accuracy and reduce off-topic articles in the filtered dataset, especially for trickier topics with ambiguous terms such as 'blind' or 'mute'.

Another potential application is to develop a public interface that allow users to retrieve the results of analyses performed in this experiment for other domains, given a list of topics (key terms and query terms) and/or an existing text corpora provided by the user. The solution described in this project would have to be generalised with an interface, where users can provide their own scrapers that implement a well-documented set of functions for other news websites/sources, define their own list of topics, keywords, and query terms, and possibly even change the independent variables and subset intervals for the analysis. It would also need to provide clear documentation on its usage and expected input structure/format. Such a solution would provide researchers with the tools to perform similar research with ease, and possibly build on the tool as part of other projects using it as a component.

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

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

[4] 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).

[5] 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.

[6] 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.

[7] 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.

[8] 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`.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

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

[21] 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.

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

[23] 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`.

[24] 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.

[25] 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`.

[26] 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).

[27] 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).

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

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

[30] 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.

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

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

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

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

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

[36] 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.

[37] 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.

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

[39] 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.

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

[41] 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, 2014.

[42] S. Banerjee and T. Pedersen, "An adapted lesk algorithm for word sense disambiguation using wordnet," in *International Conference on Intelligent Text Processing and Computational Linguistics*, Springer, 2002, pp. 136–145.

[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`.

[50] J. W. Tukey, *Exploratory data analysis*. 1977, vol. 2.

[51] J. L. Hintze and R. D. Nelson, "Violin plots: A box plot-density trace synergism," *The American Statistician*, vol. 52, no. 2, pp. 181–184, 1998.

[52] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, vol. 18, pp. 50–60, 1 1947.

[53] Statista. (2018). Share of individuals reading or downloading online news, newspapers or magazines in Great Britain from 2007 to 2017, [Online]. Available: `https://www.statista.com/statistics/286210/online-news-newspapers-and-magazine-consumption-in-great-britain/`.

[54] ——, (2018). Newspaper websites ranked by monthly visitors in the United Kingdom (UK) from 2013 to 2016 (in million visitors), [Online]. Available: `https://www.statista.com/statistics/288763/newspaper-websites-ranked-by-monthly-visitors-united-kingdom-uk/`.

[55] Guardian News and Media Limited. (2016). The Guardian – open platform, [Online]. Available: `http://open-platform.theguardian.com/`.

[56] Associated Newspapers Ltd. (). Search tips — Daily Mail Online, [Online]. Available: `http://www.dailymail.co.uk/home/article-10612/Search-Tips.html`.

[57] Express Newspapers. (). Search For " — Page 1 — Express.co.uk, [Online]. Available: `https://www.express.co.uk/search`.

[58] Judicial Council of California. (). Disability terminology chart, [Online]. Available: http://www.courts.ca.gov/partners/documents/7-terminology.pdf.

[59] Department for Work & Pensions: Office for Disability Issues. (Aug. 14, 2014). Inclusive language: Words to use and avoid when writing about disability, [Online]. Available: https://www.gov.uk/government/publications/inclusive-communication/inclusive-language-words-to-use-and-avoid-when-writing-about-disability.

[60] T. Bray, "The JavaScript Object Notation (JSON) Data Interchange Format," RFC Editor, RFC 8259, Dec. 2017, pp. 1–16. [Online]. Available: http://www.rfc-editor.org/rfc/rfc8259.txt.

Appendix A

Appendix: All results

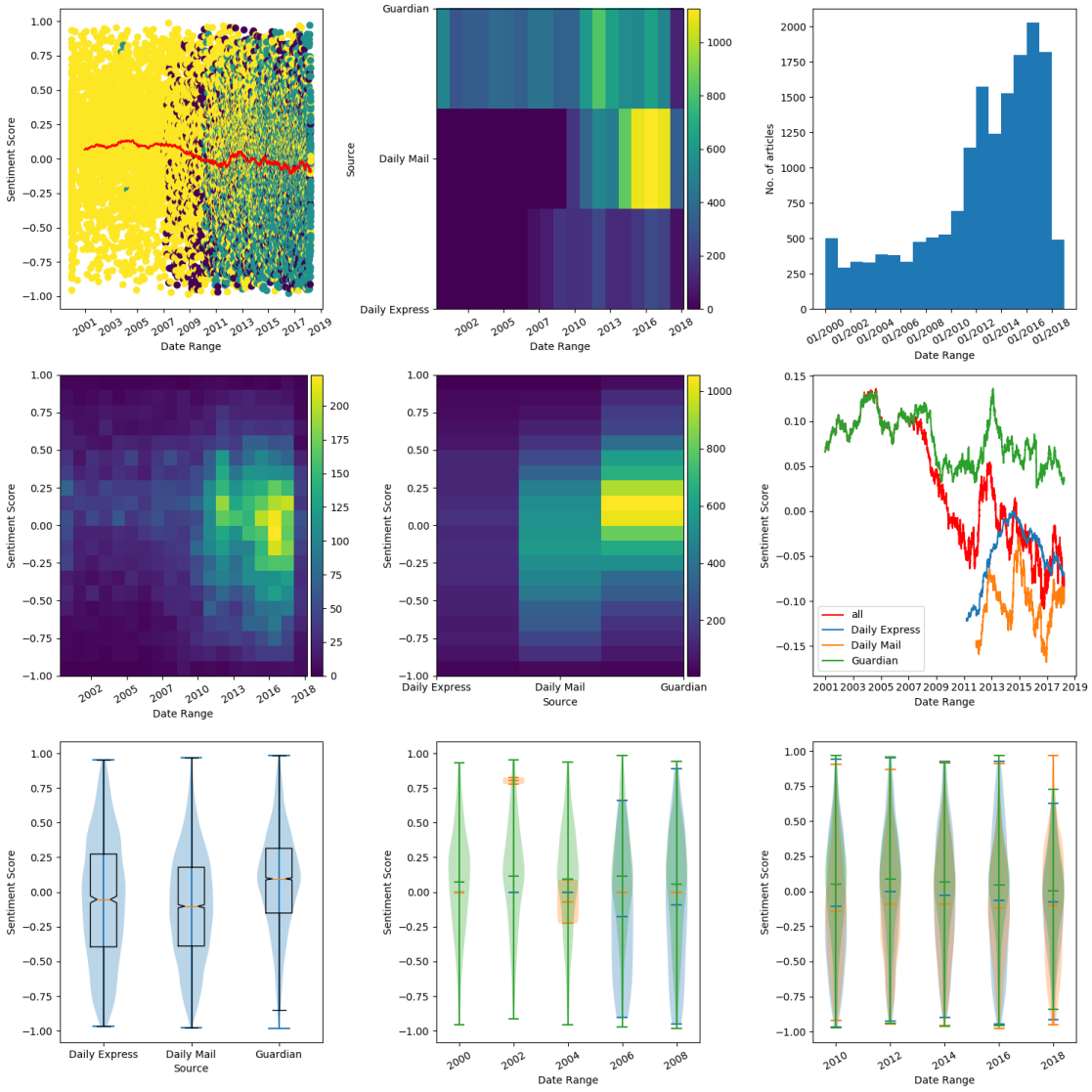
A.1 Topic: ‘disabled’

Key Terms: ‘disabled’, ‘disability’, ‘handicapped’, ‘cripple’, ‘invalid’, ‘accessible’, ‘ablism’, ‘ableism’, ‘differently abled’

Query Terms: ‘disabled’, ‘disability’, ‘ablism’, ‘ableism’, ‘differently abled’

Sample size, n = 16,411

A.1.1 Sentiment Score Plots



A.1.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$2.66 * 10^{-167}$	$2.29 * 10^{-35}$	0.000108
2007	N/A	$5.24 * 10^{-6}$	N/A
2008	N/A	0.0739	N/A
2009	N/A	$2.24 * 10^{-5}$	N/A
2010	$6.00 * 10^{-7}$	$6.89 * 10^{-6}$	0.360 **
2011	$5.73 * 10^{-16}$	$9.73 * 10^{-5}$	0.0355
2012	$4.44 * 10^{-17}$	0.00371	0.000739
2013	$5.03 * 10^{-16}$	0.00351	0.0827
2014	$1.27 * 10^{-9}$	0.00208	0.321
2015	$6.87 * 10^{-17}$	0.00855	0.00718
2016	$4.63 * 10^{-26}$	$4.74 * 10^{-6}$	0.0576
2017	$3.39 * 10^{-13}$	0.00244	0.0867
2018*	0.122	0.120	0.328

* 2018 data is incomplete and would only include articles up to (approximately) end of March.

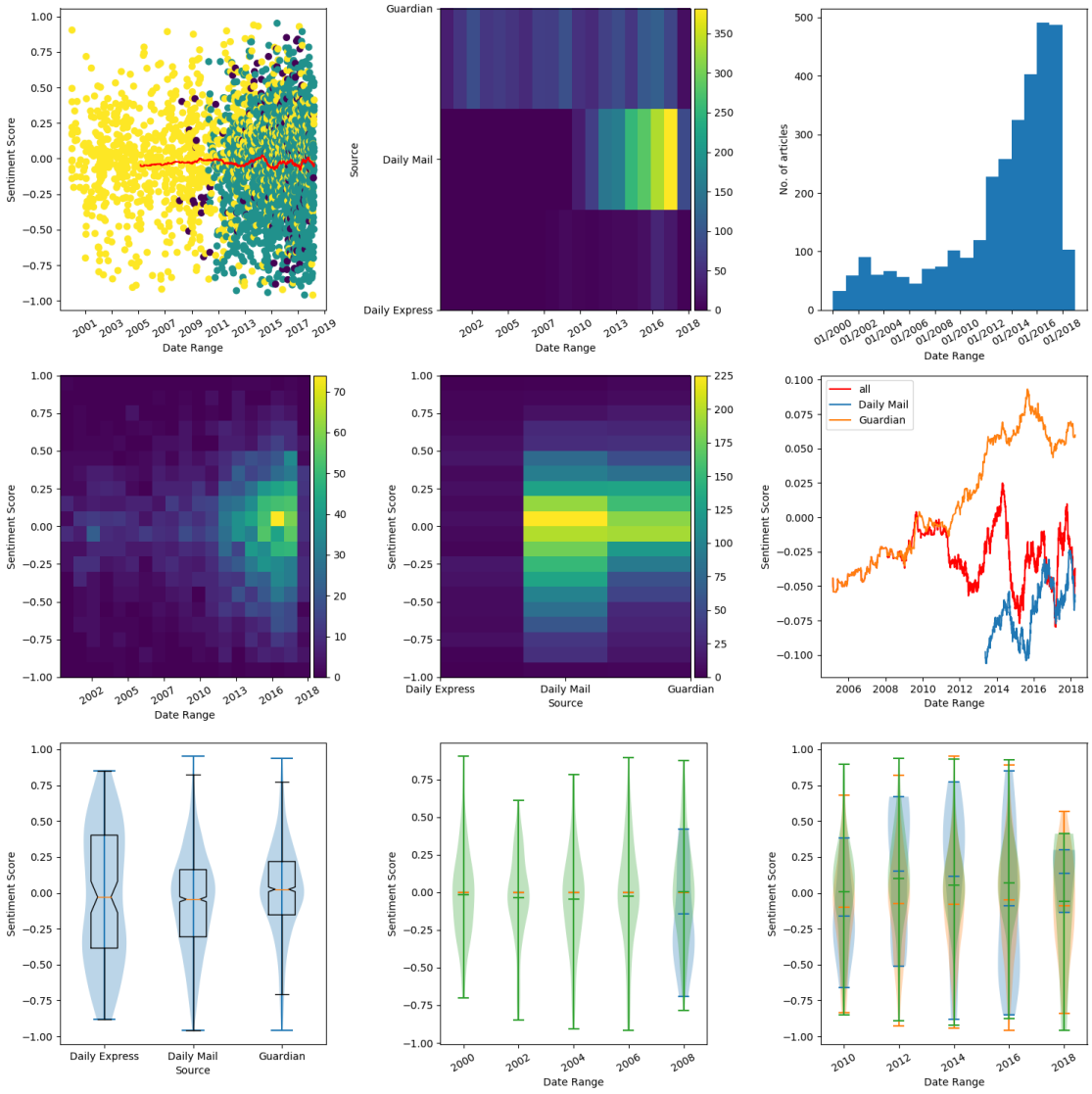
** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

A.1.3 Keyword Trend Plots

A.2 Topic: ‘autism’

Key Terms: ‘autism’, ‘autistic’, ‘asperger\’s’, ‘ASD’
Query Terms: ‘autism’, ‘autistic’, ‘asperger\’s’, ‘ASD’
Sample size, n = 3,161

A.2.1 Sentiment Score Plots

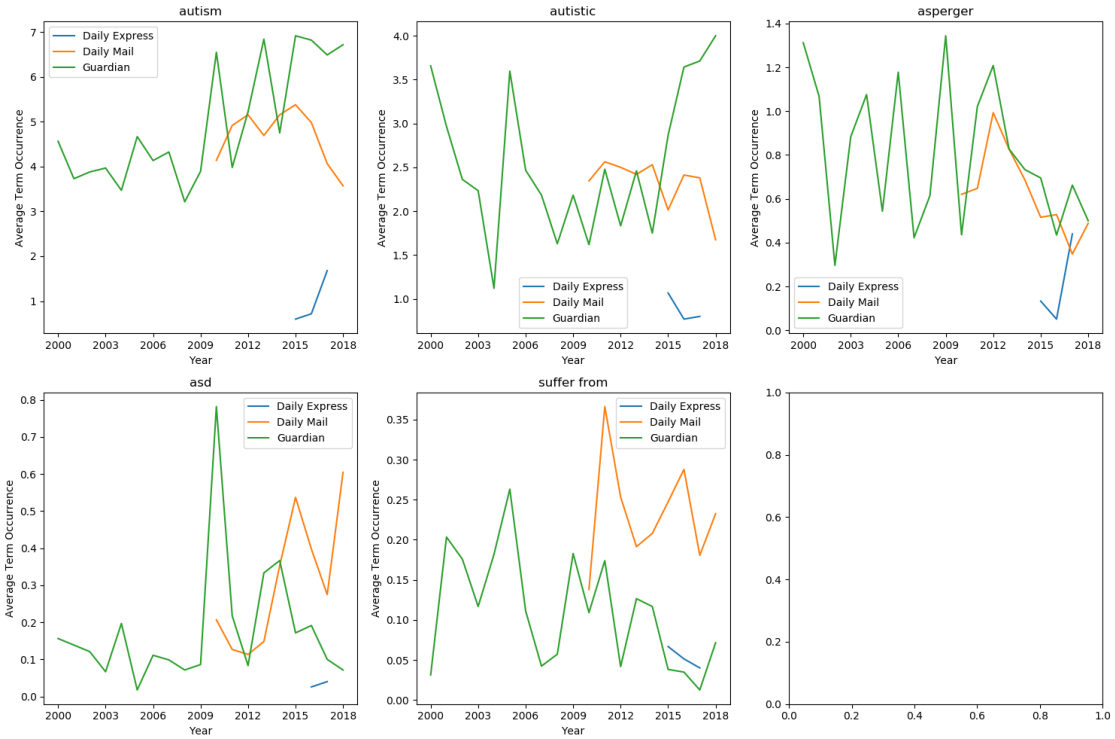


A.2.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$3.00 * 10^{-12}$	0.137	0.184
2010	0.100	N/A	N/A
2011	0.0649	N/A	N/A
2012	0.000150	N/A	N/A
2013	$1.52 * 10^{-5}$	N/A	N/A
2014	0.00162	N/A	N/A
2015	0.000508	N/A	N/A
2016	0.00760	0.0121	0.0432 **
2017	0.000114	0.0667	0.352

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

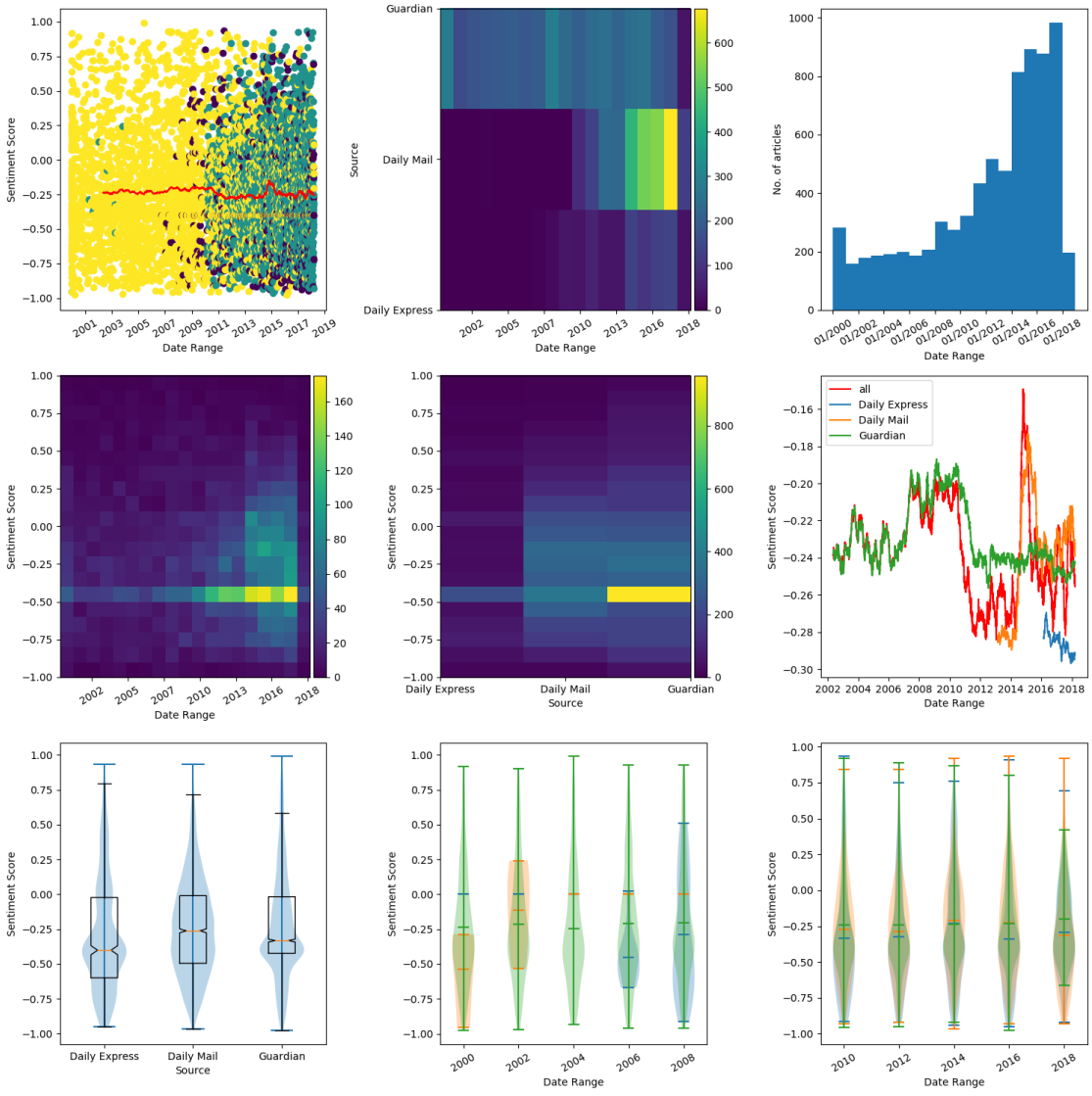
A.2.3 Keyword Trend Plots



A.3 Topic: ‘blind’

Key Terms: ‘blind’, ‘blindness’, ‘blindism’, ‘visual impairment’, ‘partially sighted’, ‘vision loss’
Query Terms: ‘blind’, ‘blindness’, ‘visual impairment’, ‘partially sighted’, ‘visually impaired’
Sample size, n = 7,677

A.3.1 Sentiment Score Plots



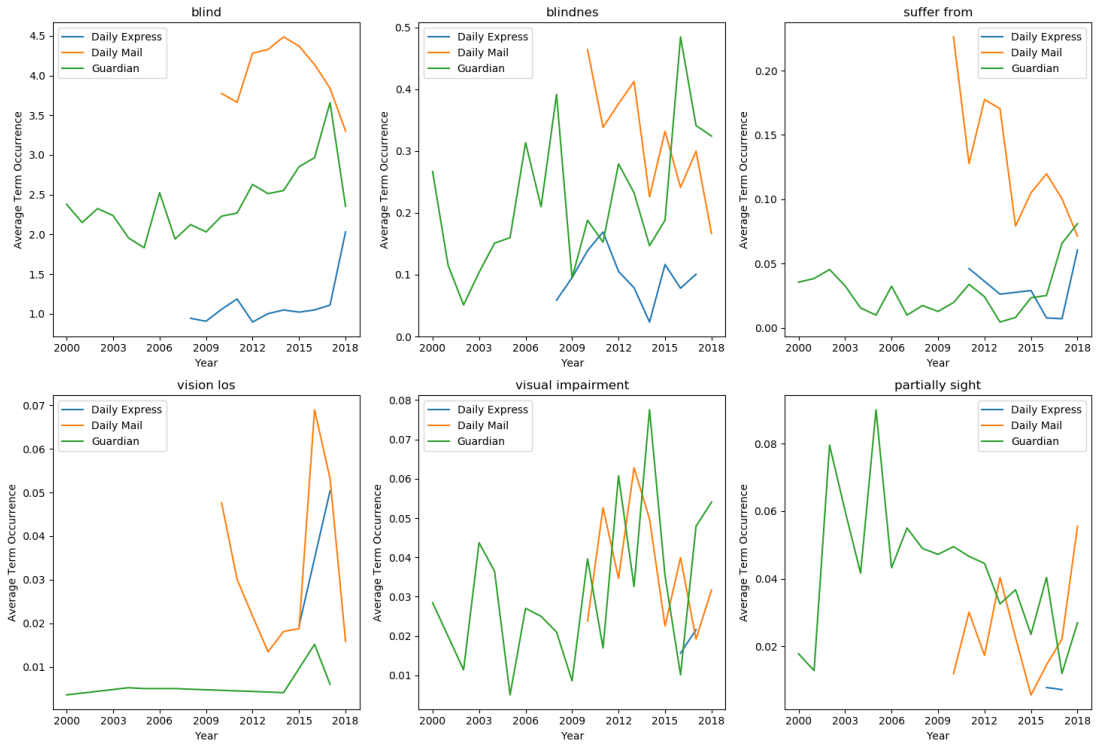
A.3.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	0.335 **	$1.62 * 10^{-9}$	$6.57 * 10^{-9}$ **
2009	N/A	0.339	N/A
2010	0.388	0.0331	0.0720 **
2011	0.493	0.0101	0.0149 **
2012	0.00585	0.00586	0.220 **
2013	0.330 **	0.163	0.158 **
2014	0.0670 **	0.263	0.0857 **
2015	0.181 **	0.214	0.125 **
2016	0.455	0.00119	0.000966 **
2017	0.421	$3.16 * 10^{-5}$	$1.46 * 10^{-5}$ **
2018*	0.0118	0.0919	0.246

* 2018 data is incomplete and would only include articles up to (approximately) end of March.

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

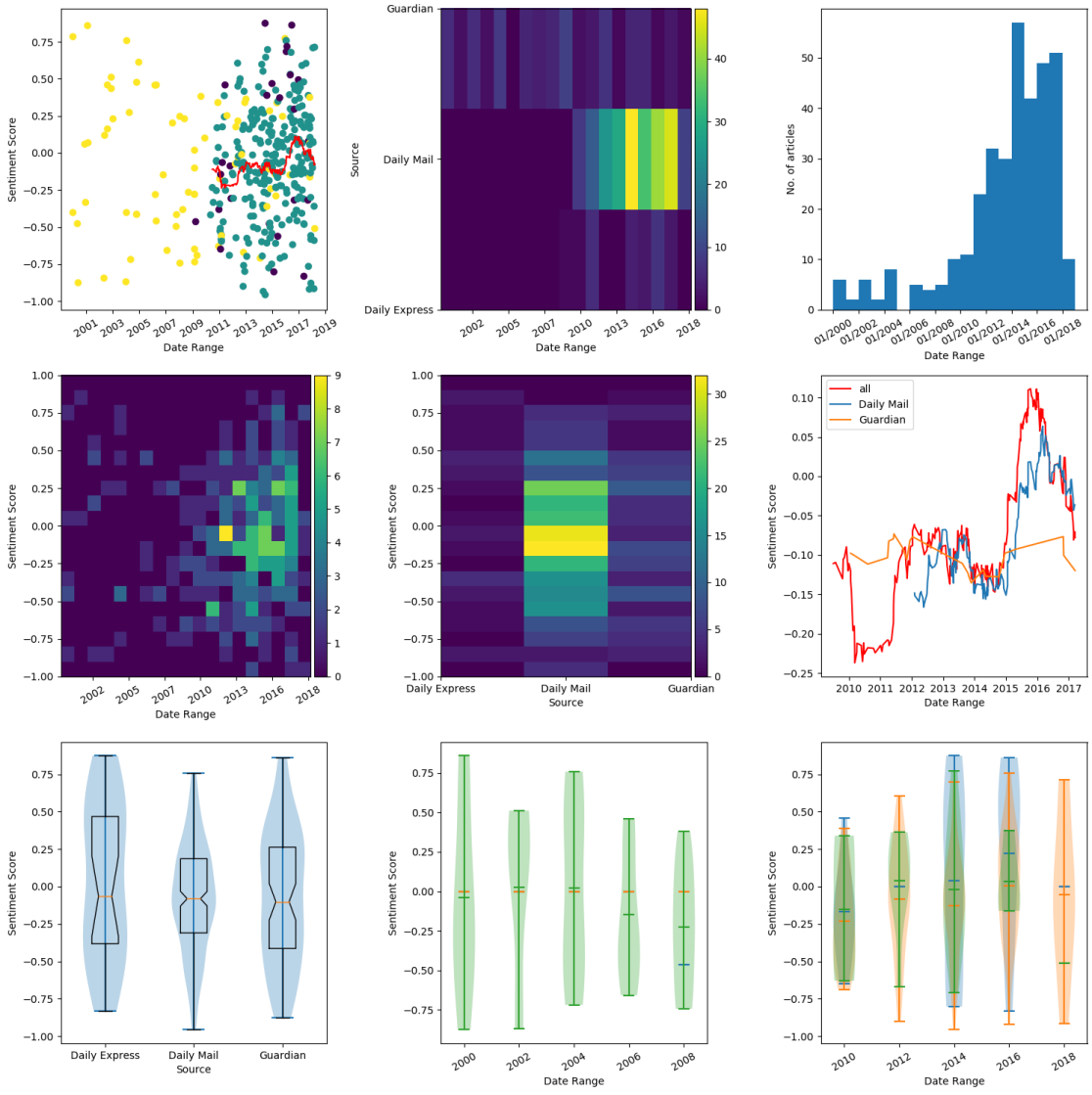
A.3.3 Keyword Trend Plots



A.4 Topic: ‘cerebral palsy’

Key Terms: ‘cerebral palsy’, ‘spastic’
Query Terms: ‘cerebral palsy’, ‘spastic’
Sample size, n = 353

A.4.1 Sentiment Score Plots



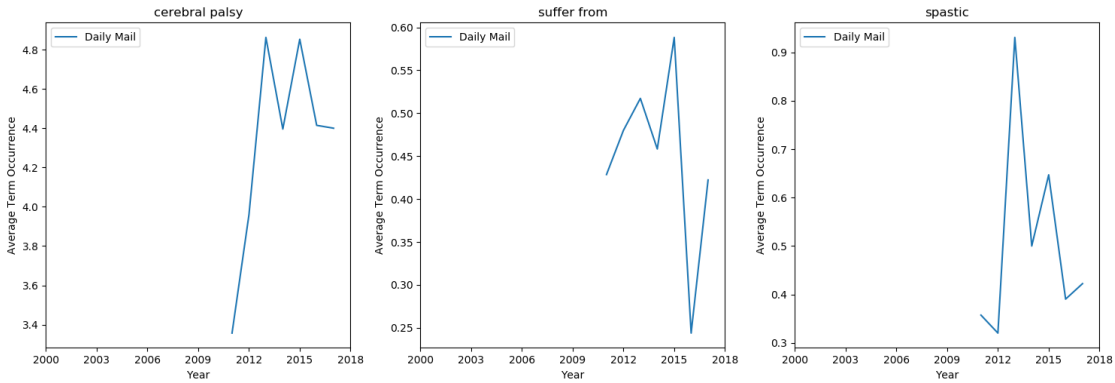
A.4.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	0.469	0.156 **	0.169

Insufficient sample size for year-by-year comparisons. (n=353)

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

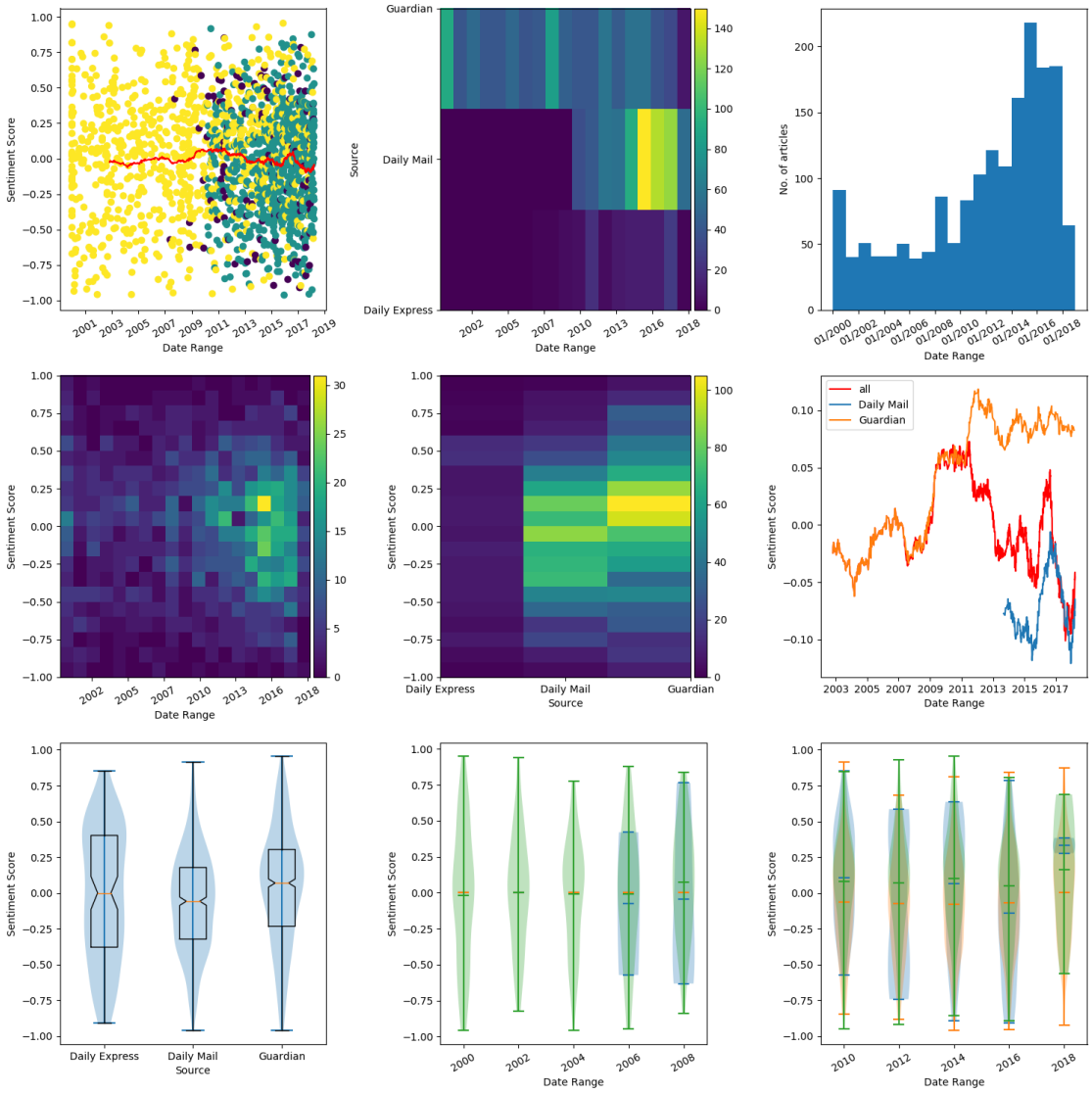
A.4.3 Keyword Trend Plots



A.5 Topic: ‘deaf’

Key Terms: ‘deaf’, ‘deafness’, ‘hearing impaired’, ‘hard of hearing’, ‘hearing loss’
Query Terms: ‘deaf’, ‘deafness’, ‘hearing impairment’, ‘hard of hearing’, ‘hearing impaired’
Sample size, n = 1,762

A.5.1 Sentiment Score Plots

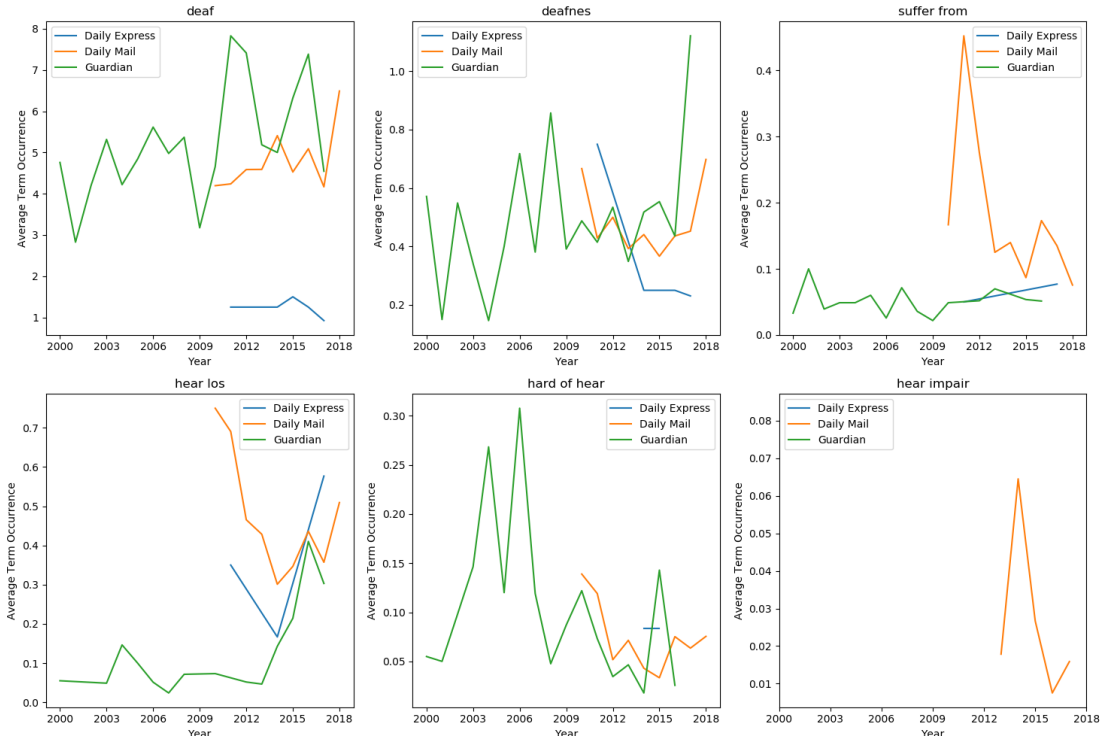


A.5.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$6.30 * 10^{-10}$	0.141	0.0833
2010	0.263	N/A	N/A
2011	0.00262	N/A	N/A
2012	0.00349	N/A	N/A
2013	0.0403	N/A	N/A
2014	0.000131	N/A	N/A
2015	0.00260	N/A	N/A
2016	0.0437	N/A	N/A
2017	0.0307	0.0480	0.233 **

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

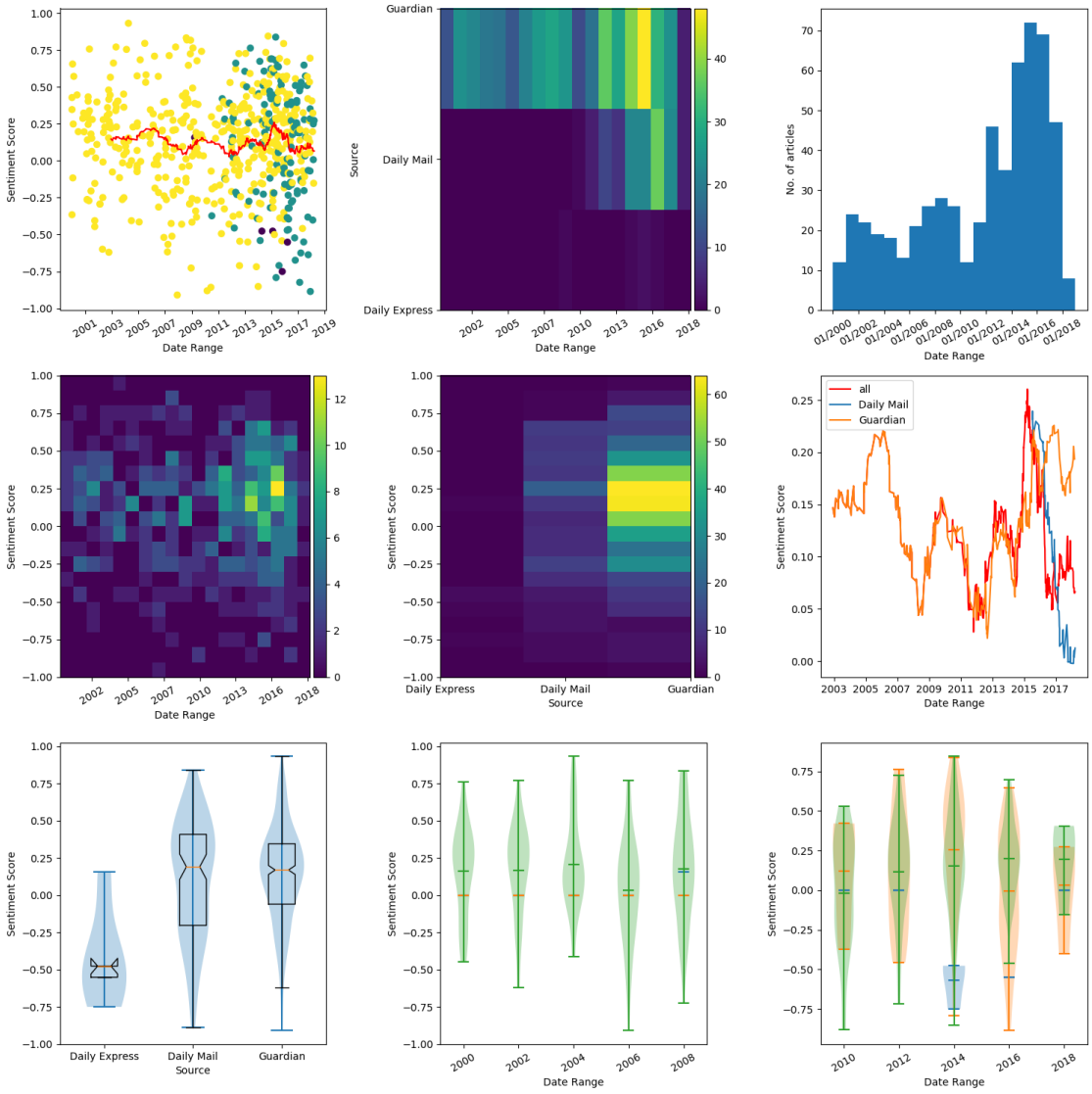
A.5.3 Keyword Trend Plots



A.6 Topic: 'developmental delay'

Key Terms: 'developmental delay', 'developmental disability', 'developmental disorder', 'learning disability', 'slow learner', 'intellectual disability'
Query Terms: 'developmental delay', 'developmental disability', 'developmental disorder', 'learning disability'

A.6.1 Sentiment Score Plots



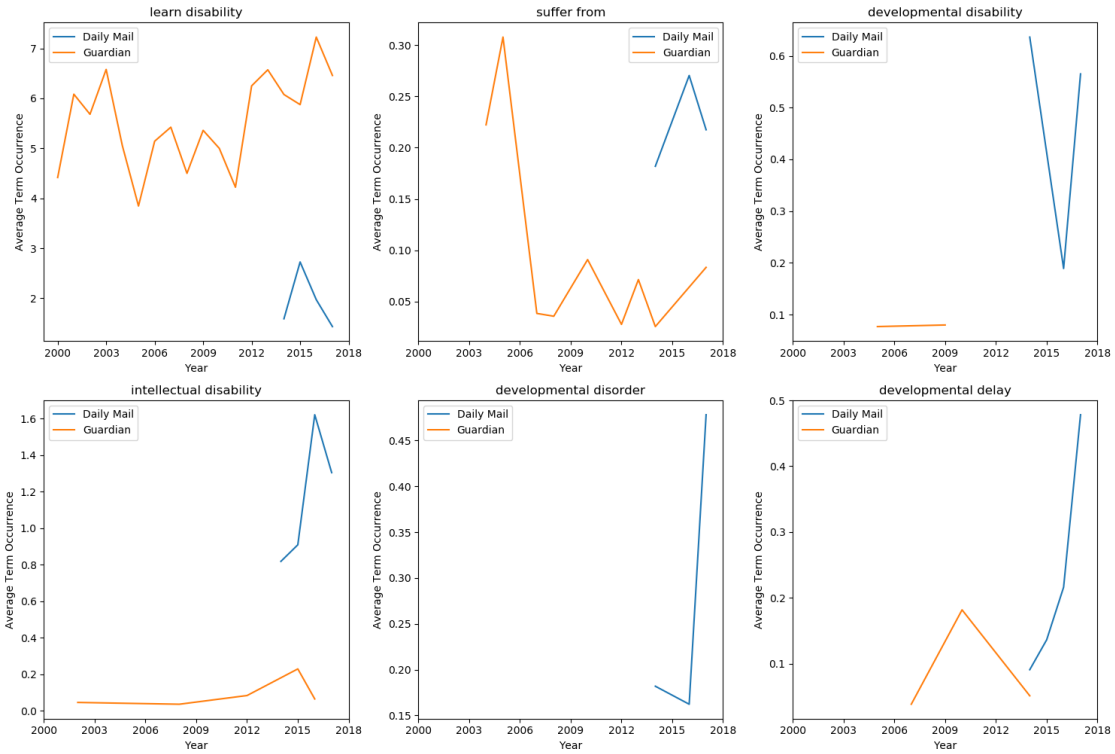
Sample size, n = 582

A.6.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	0.383	0.00179	0.00326 **
2014	0.0695 **	N/A	N/A
2015	0.106 **	N/A	N/A
2016	0.000845	N/A	N/A
2017	0.251	N/A	N/A

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

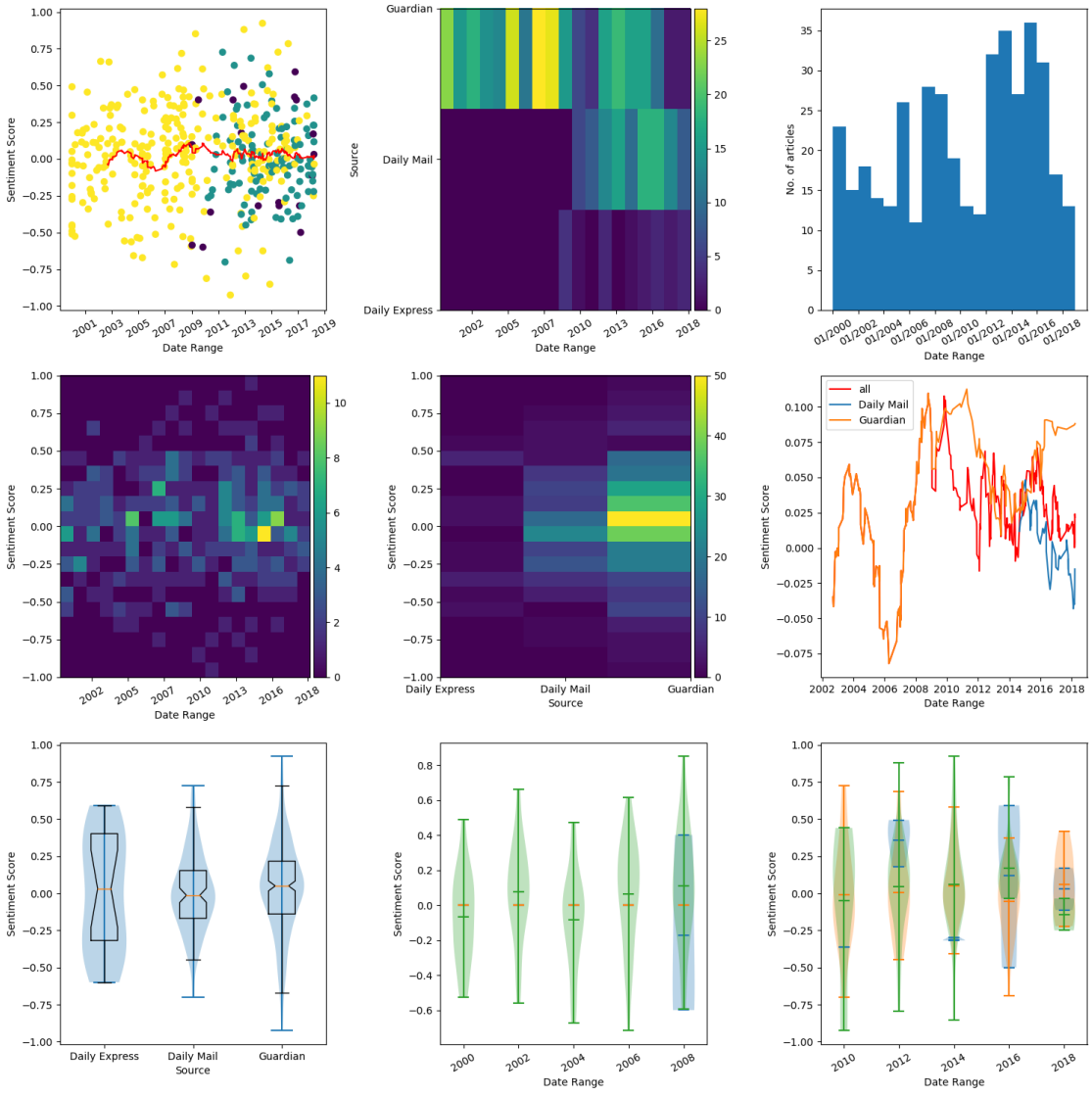
A.6.3 Keyword Trend Plots



A.7 Topic: 'dyslexia'

Key Terms: 'dyslexia', 'dyslexic'
Query Terms: 'dyslexia', 'dyslexic'
Sample size, n = 410

A.7.1 Sentiment Score Plots



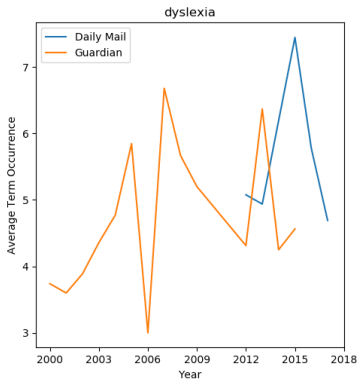
A.7.2 Mann-Whitney U Test Results (p-values)

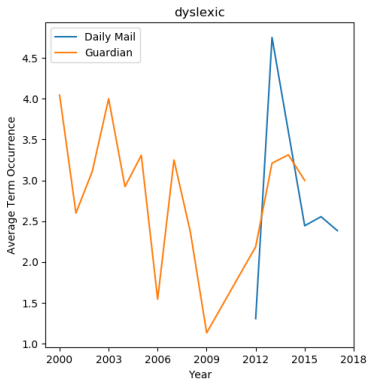
Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	0.0740	0.362	0.464 **

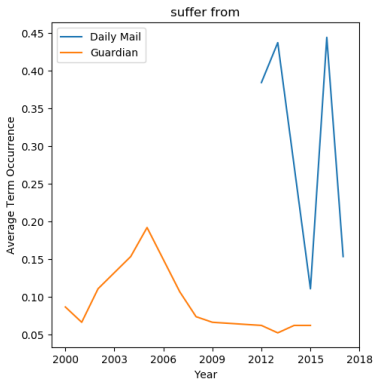
Insufficient sample size for year-by-year comparisons. (n=410).

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

A.7.3 Keyword Trend Plots



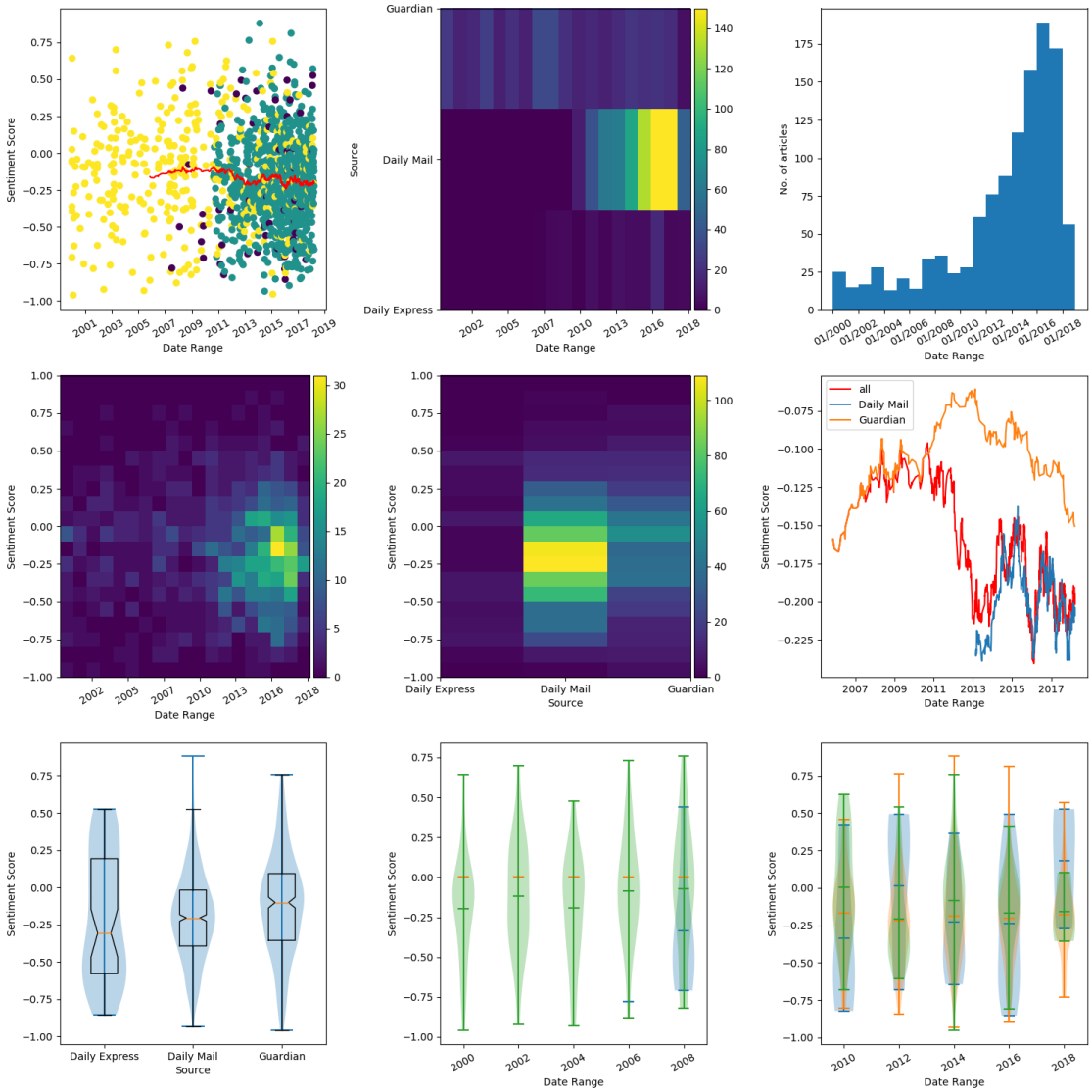




A.8 Topic: ‘epilepsy’

Key Terms: ‘epilepsy’, ‘epileptic’, ‘seizure’
Query Terms: ‘epilepsy’, ‘epileptic’

A.8.1 Sentiment Score Plots



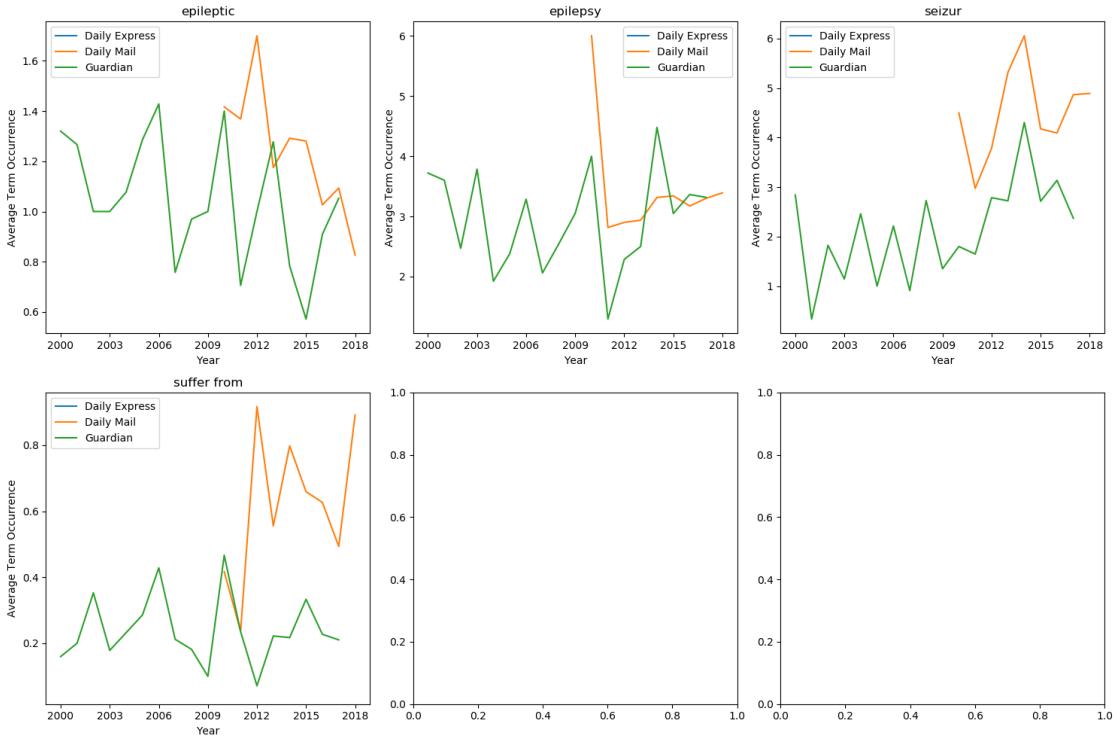
Sample size, n = 1,172

A.8.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$3.39 * 10^{-5}$	0.0579	0.322 **
2014	0.0130	N/A	N/A
2015	0.193	N/A	N/A
2016	0.398 **	N/A	N/A

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

A.8.3 Keyword Trend Plots



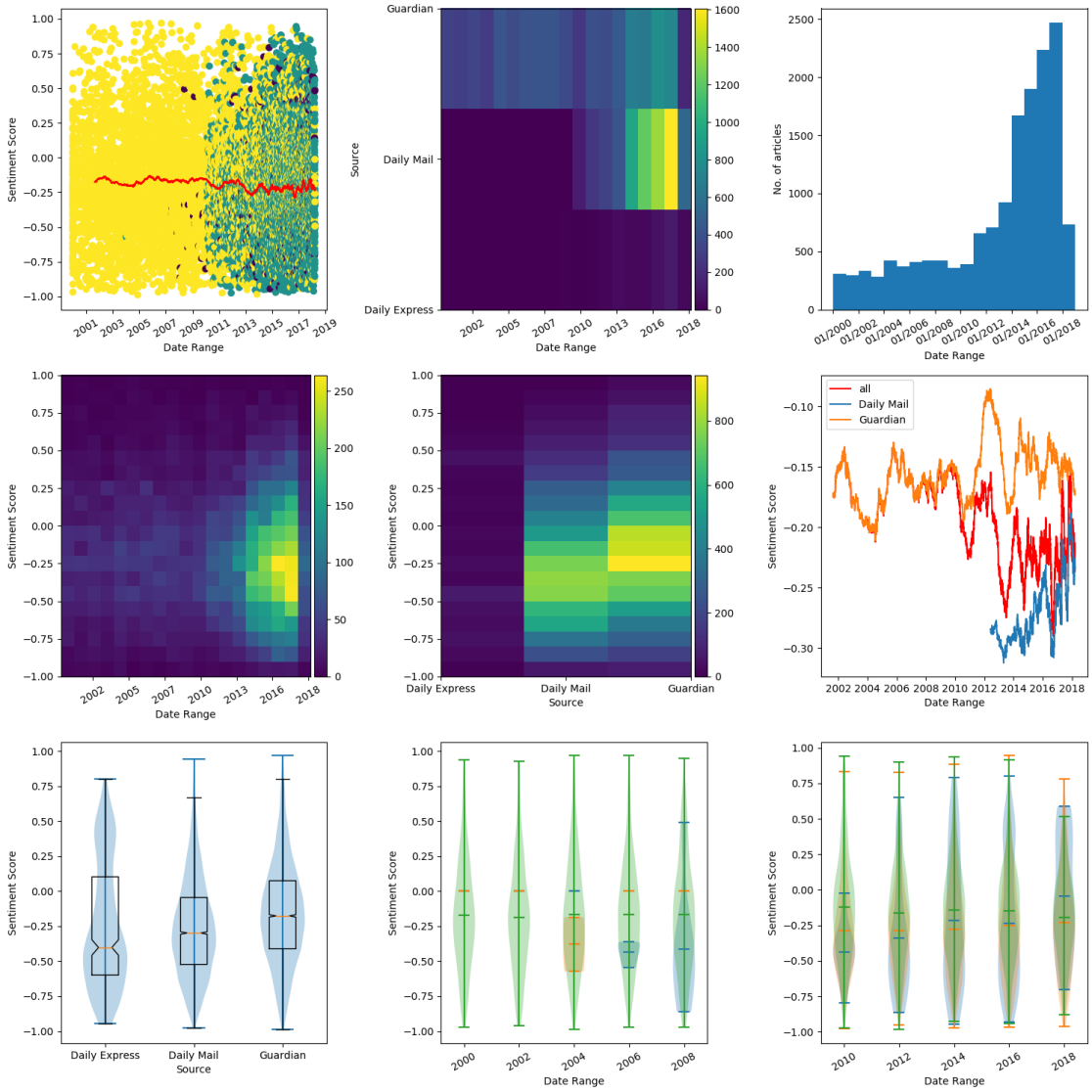
A.9 Topic: ‘mental illness’

Key Terms: ‘mental illness’, ‘mental health’, ‘mental disability’, ‘mental disorder’, ‘mental issue’, ‘brain injured’, ‘brain injury’, ‘brain damaged’, ‘psychological’, ‘psychiatric’, ‘emotional disorder’, ‘behavioural disorder’, ‘retardation’, ‘intellectual disability’, ‘mentally ill’, ‘mentally disabled’, ‘mentally handicapped’

Query Terms: ‘mental illness’, ‘mental health’, ‘mental disorder’, ‘mental disability’, ‘mentally ill’, ‘mentally disabled’, ‘mentally handicapped’

Sample size, n = 15,329

A.9.1 Sentiment Score Plots

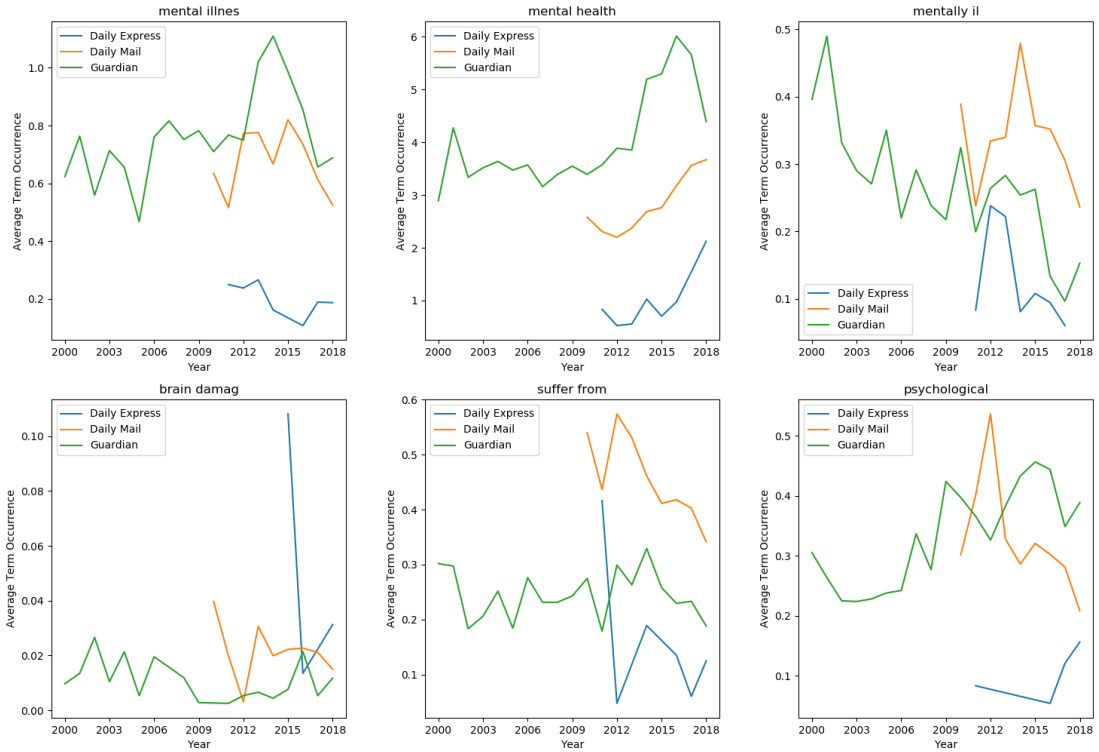


A.9.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$3.54 * 10^{-76}$	$1.51 * 10^{-9}$	0.102 **
2010	0.000255	N/A	N/A
2011	$8.79 * 10^{-11}$	N/A	N/A
2012	$5.70 * 10^{-8}$	0.377	0.112
2013	$1.13 * 10^{-7}$	$1.43 * 10^{-7}$	0.000132 **
2014	$1.19 * 10^{-22}$	0.108	0.178
2015	$2.77 * 10^{-10}$	0.0588	0.401 **
2016	$1.83 * 10^{-19}$	$2.56 * 10^{-11}$	$2.03 * 10^{-5}$ **
2017	$5.55 * 10^{-10}$	0.389 **	0.00826
2018*	0.0531	0.0413 **	0.00878

* 2018 data is incomplete and would only include articles up to (approximately) end of March.
** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

A.9.3 Keyword Trend Plots




psychiatric
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year
mental disability
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year
mentally disabl
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year
mental disorder
Daily Express
Daily Mail
Guardian
Average Term Occurrence
Year
brain injury
Daily Mail
Guardian
Average Term Occurrence
Year
retardation
Daily Mail
Guardian
Average Term Occurrence
Year
mentally handicap
Daily Mail
Guardian
Average Term Occurrence
Year
mental issu
Daily Mail
Guardian
Average Term Occurrence
Year
intellectual disability
Daily Mail
Guardian
Average Term Occurrence
Year
behavioural disorder
Daily Mail
Guardian
Average Term Occurrence
emotional disorder
Daily Mail
Guardian
Average Term Occurrence
brain injur
Daily Mail
Average Term Occurrence

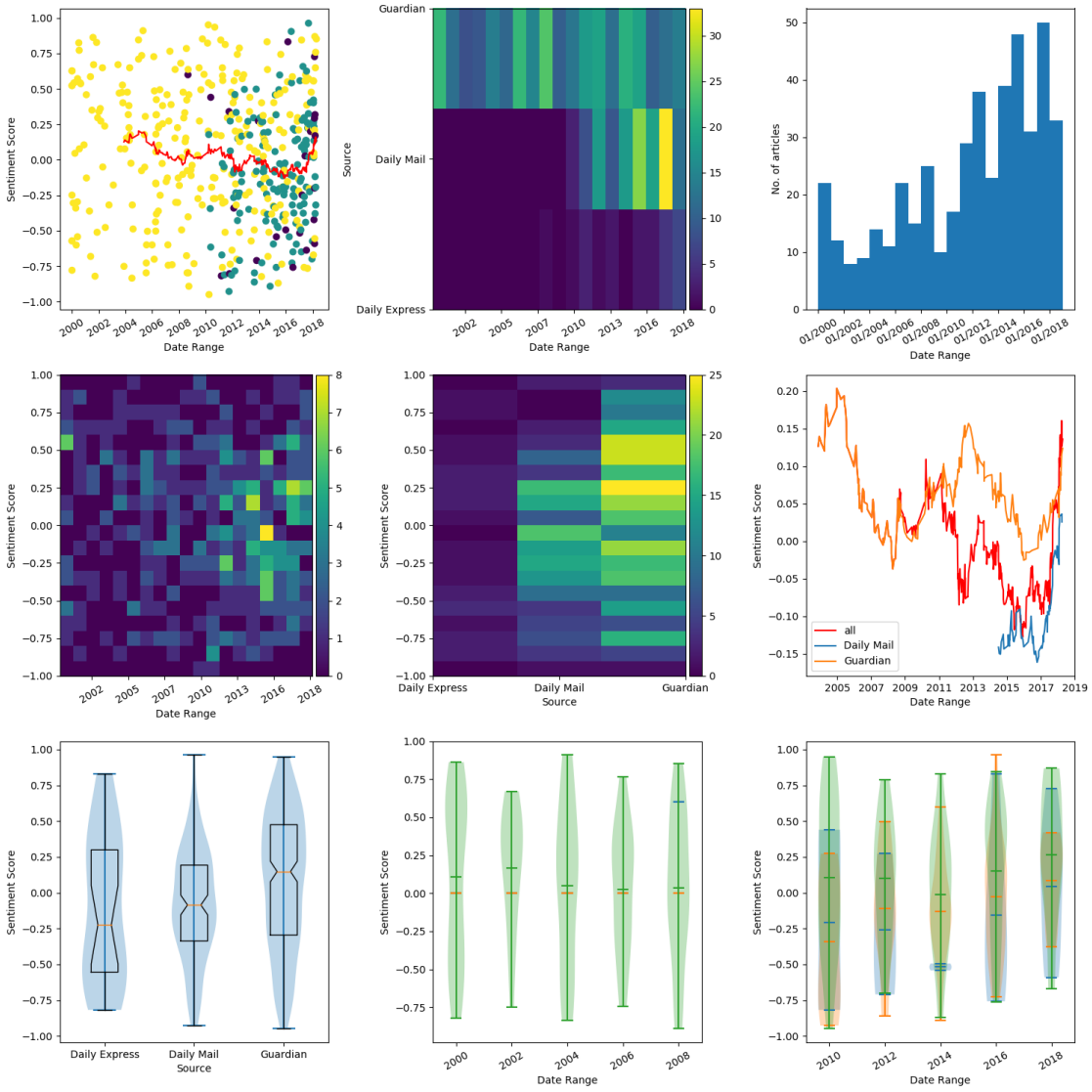
A.10 Topic: ‘mute’

Key Terms: ‘mute’, ‘muteness’, ‘mutism’, ‘cannot speak’, ‘difficulty speaking’, ‘synthetic speech’, ‘non-vocal’, ‘non-verbal’

Query Terms: ‘mute’, ‘muteness’, ‘mutism’, ‘non-verbal’

Sample size, n = 456

A.10.1 Sentiment Score Plots



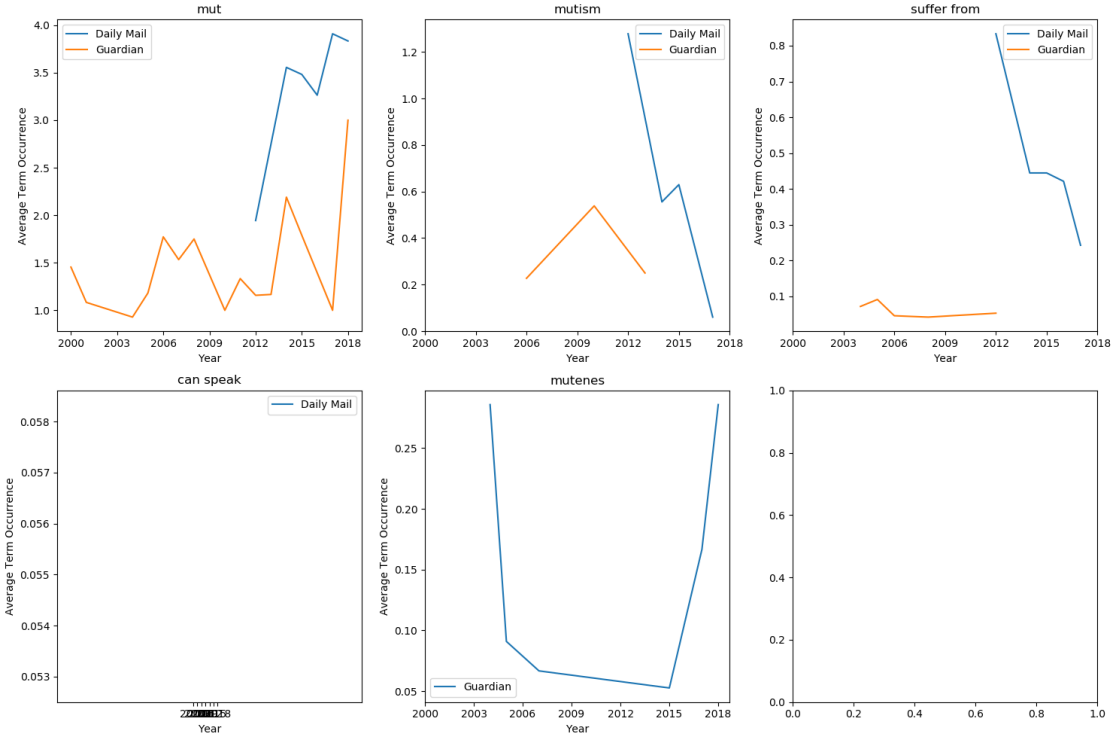
A.10.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$6.78 * 10^{-5}$	0.0311	0.346 **

Insufficient sample size for year-by-year comparisons. (n=456).

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

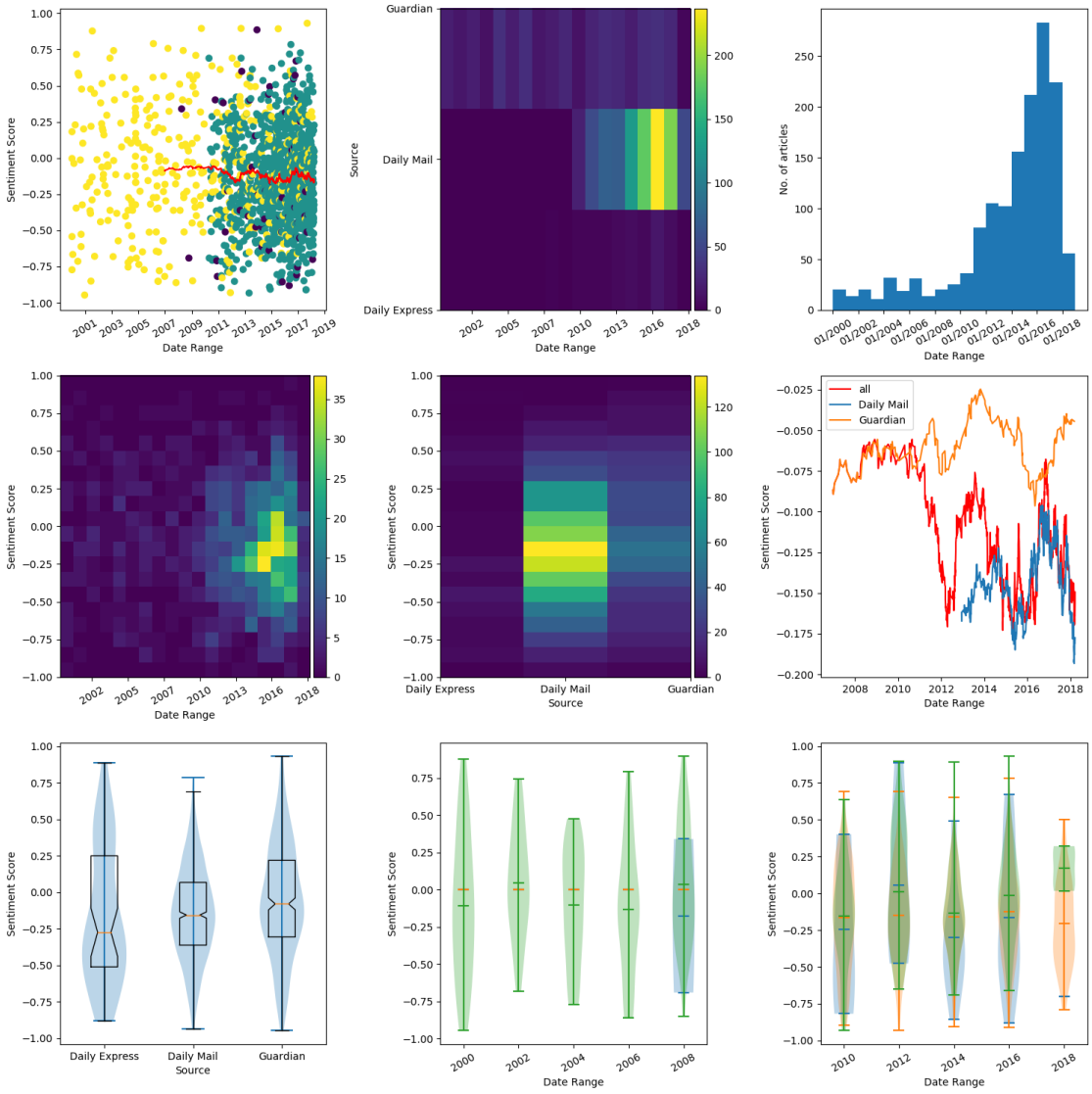
A.10.3 Keyword Trend Plots



A.11 Topic: ‘paralysis’

Key Terms: ‘paraplegic’, ‘quadriplegic’, ‘spinal cord’, ‘paraplegia’, ‘quadriplegia’, ‘paralysed’, ‘paralyzed’, ‘paralysis’, ‘crippled’, ‘leg braces’, ‘wheelchair’
Query Terms: ‘paraplegic’, ‘quadriplegic’, ‘paraplegia’, ‘quadriplegia’, ‘paralysis’
Sample size, n = 1,461

A.11.1 Sentiment Score Plots



A.11.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	$2.73 * 10^{-5}$	0.00909	0.0827 **
2011	0.352 **	N/A	N/A
2012	0.00184	N/A	N/A
2013	0.374	N/A	N/A
2014	N/A	N/A	N/A
2015	0.331	N/A	N/A
2016	0.149	N/A	N/A
2017	0.00431	N/A	N/A

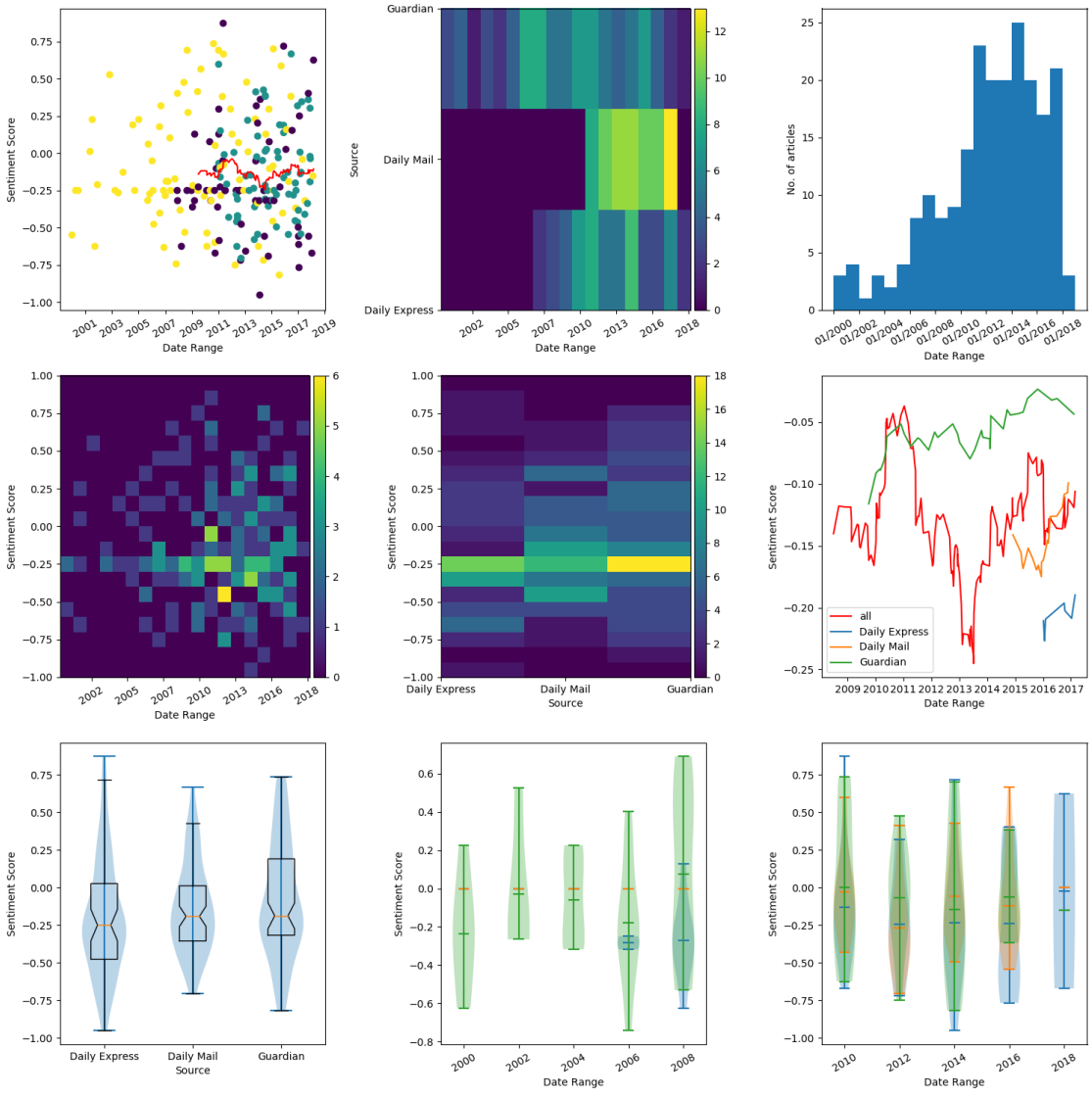
** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

A.11.3 Keyword Trend Plots

A.12 Topic: 'speech impairment'

Key Terms: 'speech impairment', 'stutter', 'speech disability', 'speech disorder', 'communication disability', 'difficulty speaking', 'language impairment', 'language disorder', 'language disability', 'speech impediment', 'stammer'
Query Terms: 'speech impairment', 'stutter', 'speech disorder', 'speech impediment'
Sample size, n = 215

A.12.1 Sentiment Score Plots



A.12.2 Mann-Whitney U Test Results (p-values)

Topic	Guardian > Daily Mail	Guardian > Daily Express	Daily Express > Daily Mail
All	0.235	0.0200	0.0546 **

Insufficient sample size for year-by-year comparisons. (n=215).

** Indicates where the reverse assumption is true (e.g. Daily Mail > Daily Express instead of Daily Express > Daily Mail)

A.12.3 Keyword Trend Plots