OLLSCOIL TEICNEOLAÍOCHTA
BHAILE ÁTHA CLIATH
DUBLIN
TECHNOLOGICAL
UNIVERSITY DUBLIN

Automatic Stance Detection using LDA Topic Modelling
Interim Report

DT228
BSc in Computer Science

Michael Lenghel

C16434974

Dr. John Gilligan

School of Computer Science

Technological University, Dublin

08/12/2019

Abstract

This thesis is an investigation into how an algorithm implementing latent Dirichlet allocation (LDA) topic modelling and stance detection can automatically identify bias within newspaper companies and determine and compare their stances on various topics. Stance detection determines the point of view or position that the media presents to the public in relation to a topic. It will also provide an overall view of how all media in newspapers perceives these topics and what are the most discussed topics within the media.

The scope of this dissertation is to use LDA modelling to create topics dynamically such as religion, health, and politics through linking words to various topics automatically and implement a stance detection algorithm that can automatically determine a newspaper companies' stance on such a topic through the use of sentiment scores per each topic. This sentiment score will be able to determine overall media perception, bias within certain newspaper companies and provide the means to compare the sentiment on topics across these companies.

For example, the topic of Brexit would be measured to determine if media perception is overall positive or negative. The second score measures the response and bias of individual companies. The procedure can be applied to various fields such as the area of mental health or religious beliefs.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

Michael Lenghel

Michael Lenghel

08/12/2019

Acknowledgements

I would like to thank my project supervisor, Dr. John Gilligan for his advice and guidance throughout this project and my family for their continued support.

Table of Contents

Table of Figures

1. Introduction

This section introduces the concepts of latent Dirichlet allocation topic modelling, the newspaper dataset, sentiment analysis and how this thesis plans to unite them in order to be able to determine how a newspaper company reflects sentiment towards certain topics, be it negative, neutral or positive.

1.1. Project Background

According to the Joint National Readership Survey (JNRS) 2012/2013 almost three million people in Ireland read the newspaper regularly [1]. The views presented by the media can undoubtedly influence and skew public opinion. The manner in which various issues such as mental health, political movements and technologies are presented is crucial in understanding the position that a news company is attempting to uphold in regard to that topic and this can become an issue where a newspaper company is biased towards a certain topic. As an example, by simply presenting a certain technology in a negative or positive light it can drastically affect the technologies public perception and naturally as a direct result increase or decrease the sales associated with that technology.

"Automatic Stance Detection" is an approach which uses sentiment analysis and topic modelling in order to understand the position of media in regards to various topics. LDA encapsulates the automatic creation of topics by building multiple topics which can be used to get the phrases used in a number of texts that relate to a topic and the sentiment analysis automatically discovers the stance within those sentences. Both approaches use machine learning in order to build models that can create the topics and understand the sentiments. An average is then accumulated over the respective topics in order to discover the stance towards this topic.

An example of stance detection is finding all of the occurrences within articles that discussed politics and determine the sentiment score on each occurrence. An average is the created and if the score is 0.8 this would mean that the news company has a positive view of this topic and inversely if the score is 0.2 it would mean they have a negative view towards this respective topic.

A need for automatic media stance and bias perception is also evident through the large amount of research work residing within this area. Multiple dissertations, research papers and conferences have been dedicated to unlocking more powerful methods to process media information and extract more meaningful data.

As discussed, the need for an automatic newspaper stance detection algorithm across topics seems to be crucial in terms of better understanding media perspective and attempting to determine which companies display bias towards certain topics.

Since the conception of sentiment analysis, the area has undergone a large shift of alterations. The cornerstone of this area where the level of research rose significantly was in 2004 where there was an outbreak of subjective text on the web. Consequently, 99% of the papers on sentiment analysis have been published after 2004 [2]. Since then the area has evolved significantly, as have the terms dictating this area. Originally sentiment analysis within the media was referred to as "Sentiment Analysis", which then shifted to "Points of View Analysis" and now finally in 2019 it is referred to as "stance analysis".

latent Dirichlet Allocation (LDA) Topic Modelling developed by Blei in 2003 has grown to be known as one of the most used topic modelling algorithms. It is an essential function to build topics from multiple documents using word frequency through statistical probability. It's use in natural language processing has increased and it is also a major area within computer science.

LDA predominately is used by organisations to analyse qualitative data such as interviews and surveys automatically by building topics in conjunction with thematic analysis. Using LDA to build topics in various media platforms such as twitter and newspapers is more recent and an immense point of discussion within computer science currently.

The component within this thesis that separates it from other projects is the combination of both stance detection and LDA Topic Modelling to automate topic creation within the news media in Ireland to uncover general media perception and individual news company bias.

1.2. Project Description

Automatic Stance Detection using latent Dirichlet allocation Topic Modelling is a research project that will investigate how popular newspaper companies in Ireland portray information to the public. The main application will be to identify what stances the overall media and different newspaper companies take on different issues and to assist in determining bias. It will also have the application of allowing the newspaper companies themselves to analyse their own bias towards certain subjects.

A growing area within natural language processing and machine learning is topic modelling. Topic modelling can be used to discover the abstract topics that occur in a collection or group of documents through examining the themes in a large collection of documents. A more formal definition of topics is that a topic is a multinomial distribution over a fixed vocabulary [3].

I will be using topic modelling to build topics within newspapers articles. The topic modelling algorithm of choice that I will be using is LDA which is a probabilistic method. LDA maps all the of the words within documents to topics in order to match them to imaginary topics. A more formal definition of LDA is "LDA is a generative statistical model that allows certain sets of observations to be explained by unobserved groups that may explain why some parts of the data are similar." [4].

The true power of LDA is that it can build a model that correlates certain words with a topic which can save a great deal of time as ordinary thematic analysis would require analysing texts by hand and manually mapping certain words which link to a topic. A basic example is that if in an interview a key topic that a company is interested in is streaming services, then the words correlated with streaming services could be "HBO, Netflix and Amazon prime" and all other streaming services deemed important. This process would then need to be applied to every topic which can be cumbersome where the topic count can be within the hundreds.

The approach that I am taking is to build a reference model to multiple topics such as religion, sports and politics, be able to expose when a newspaper is referencing a particular topic and then calculate their stance on this topic through a sentiment score where they discussed a particular topic across multiple papers. Afterwards, different newspapers such as "The Independent" and "The Irish Times" can be compared on how they discuss this topic to determine if one holds a stronger stance on certain issues, favours certain political parties over others or even upholds a bias towards certain topics. Naturally this stance will be also be calculated over an aggregation of multiple newspaper companies and therefore will be able to pinpoint how the media within Ireland perceive certain topics.

An example of a recent research project done within the scope of LDA is the journal publication [5] "Data Analysis and Visualization of Newspaper Articles on Thirdhand Smoke: A Topic Modelling

Approach”. Within this research a topic model is built on whenever third hand smoke is mentioned within the Chinese Media. This model was then used to understand the role the media plays in communicating this health concern.

The below diagram shows how LDA Topic Modelling generates topics. Firstly, it chooses a distribution of the topics which is displayed through the histogram on the right and then for each word choose a topic assignment (which is represented as the coloured coins) and then choose the words corresponding to each topic.

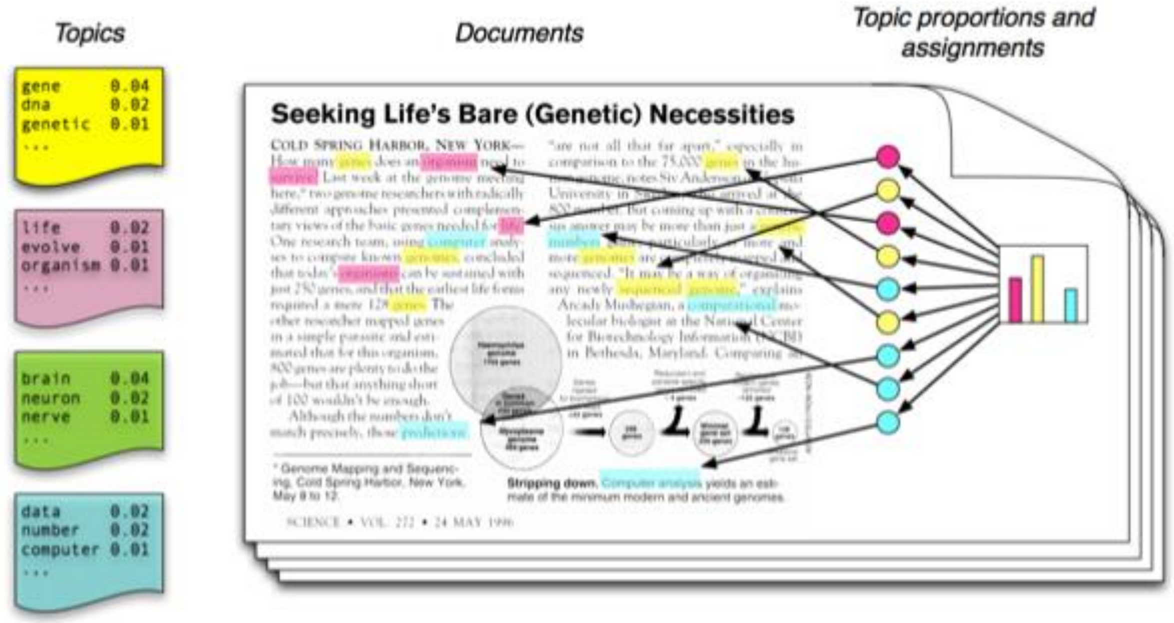


Figure 1 LDA Topic Modelling Through Mapping Words to Topics [6].

Stance detection is the key natural language processing technique that will provide insight into the media’s perspective on events. It requires a large amount of data cleaning. The data cleaning performed to be able to determine the sentiment of a sentence is as follows.

- Tokenization where each text is broken up into individual words.
- Stop word filtering where all stop words which are essentially words that carry no sentiment such as “to, is and I” are removed from the text.
- Useless information such as emails and numerical values such as dates are removed.
- Stemming and lemmatization is performed where suffixes and prefixes are removed to transform the word into its base state, for example the word “going” is transformed to the word “go”.
- A classification algorithm which determines the class a sentence belongs to, either positive, neutral or negative.
- Appling the sentiment class to the proposed sentence.

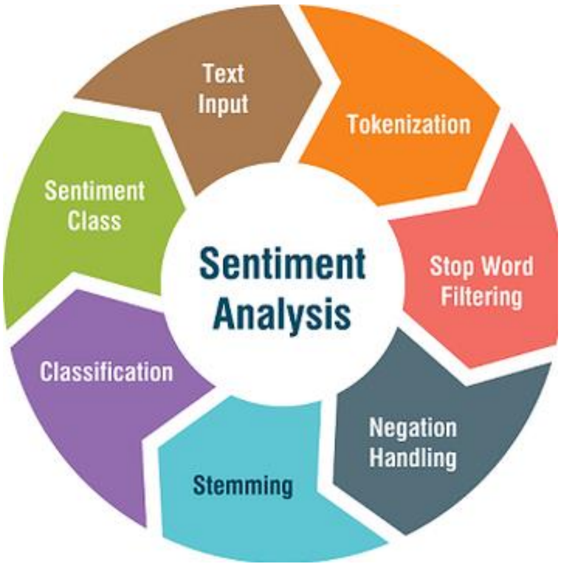


Figure 2 Stage of Developing Sentiment Analysis Algorithm [7]

1.3. Project Aims and Objectives

The overall aim of this dissertation will lead to the creation of a system that can successfully determine the overall sentiment of news companies in Ireland to various topics. To implement such a system several objections will need to be completed in phases.

Firstly, a suitable data source of news articles will be required to train models. For this phase there is a requirement of at least seven thousand articles in order to be build an accurate model. Once a semi-accurate model is built the next phase will be to connect to API's that provide newspaper that the models can be tested on to build even large LDA models and start testing the sentiment towards these scores.

The next phase requires the cleaning of data for both the LDA model and the Sentiment Analysis model. This will require tokenization, removing stop-words, getting rid of useless information such as numerical values, emails, headers, dates, new line characters and other information which is not useful for either building a model or sentiment analysis.

The next step is to build a suitable LDA model that can select themes from the articles and link the words of those themes to those models. There are a wide range of methods to implement this feature and many different libraries that can both optimise this model as well as improve the quality of models created.

Following the creation of an LDA model, a problem that occurs is that words are linked to an arbitrary number and an algorithm mapping the number to a topic will be required. This algorithm requires mapping word frequency within each topic number and using probabilistic similarity to match the number to the real topic. The data size will be important as there is a requirement of many articles to map this number correctly.

A method of visualization of the LDA topics will be another key objective in order to be able to accurately determine how accurately words map to their respective topics, as well as how linked certain topics are to each other.

A sentiment analysis will need to be performed or each reference to the topic. In an ideal case scenario, each topic will have a positive, neutral and negative ranking which will be averaged through each occurrence.

The sentiment score will need to be mapped to the created topic inside the LDA model which may prove to be a challenge as a great deal of processing separates the two methods and some ingenuity to map them together may be required.

Another key objective will be to evaluate the quality of the topics created through the LDA model, as well as how accurate the sentiment analysis scores were for each topic by comparing their accuracy to random participates. These participates will be required to fill out a questionnaire where they try to map the given words to a topic that the LDA algorithm created and see if they arrived at the same conclusion as the model.

Finally, the last objective is to further increase the accuracy of the model through better data cleaning, reducing the number of edge cases where the algorithm loses accuracy, increase the data size so that the model can be trained more accurately and many more techniques that are commonly used in natural language processing and machine learning in order to further improve the accuracy of both the LDA topic model and sentiment analysis scores.

1.4. Project Scope

This system will not be able to make direct assertions about newspaper companies. The implementation will instead create scores that dictate how positively or negatively the media and specific newspaper companies depict certain topics and it is then up to the user of the system to make a more profound judgement on the implication of the results. A negative score does not necessarily imply that the media is against this topic as it may indicate that they talked about it negatively in the past and that they do not hold the same position.

There are no plans to create this system on a remote server that will be hosted within a web application. Everything within this system will be ran locally on a machine that sets up the environment using the documentation that is provided. The documentation will also be specific to a certain configuration and will be limited to certain versions of python.

The implementation of the system will be targeted towards the English language. The system will not look towards extensibility for multiple languages or the implementation of the natural language processing.

1.5. Thesis Roadmap

This section will provide a summary of each of the chapters covered within this report.

Research

This chapter explores the background research related to LDA topic modelling, sentiment analysis, the visualization of the models, finding a data source to train the models and connecting to API's which source real newspaper articles. This chapter will also discuss the array of other relevant research that has been performed within this area of natural language processing and machine learning.

Design

The design chapter delves into and breaks down the different software methodologies that were considered and the thought process that was used to arrive to a conclusion. A view of the system architecture, use-cases and sequence operation will then be presented.

Development

The development chapter describes the entire development process of the system using the technical architecture that was described within the design section. The challenges and issues encountered will also be broken down within this chapter providing both the solutions and trade-offs.

Testing and Evaluation

This chapter will describe how the system was tested and evaluated. The key components that will be tested will be the accuracy of the LDA model to create topics, as well as the accuracy of the sentiment analysis. Both components will undergo multiple forms of testing and evaluation that will compromise both computer-only generated scores and how well the machine learning can resemble a real individual’s ability to manually understand topics and sentiment.

Conclusions and Future Work

The final chapter will reflect on the entirety of the project and will discuss the conclusions drawn and future work planned for the project.

2. Literature Review

2.1. Introduction

In this chapter some of the key areas of research that are related to this topic will be presented. Modern applications implementing LDA Topic Modelling will be explored, the current state of stance detection, natural language processing techniques as well as methods to obtain the most usable and accurate training and application data.

2.2. Alternative Existing Solutions to Your Problem

The area of sentiment analysis and particularly topic modelling is currently a topical area of growth within computer science and there are a large range of projects within both areas. Multiple projects have been built around the area of detecting media bias, detecting points of view, fake news detection, stance detection and topic authenticity. The purpose of this section is to explore and discuss the various research projects around these topics, that share similarity to the solution presented within this dissertation and draw conclusions around their application, significance and learning points.

What is Automatic Stance Detection?

"Automatic Stance Detection" is an approach which uses sentiment analysis and topic modelling in order to understand the position of media in regards to various topics. The automatic generation of topics is controlled by the topic modelling algorithm of choice which dynamically creates topics and binds the words most likely to be associated with that topic to their respective topics. The stance detection is performed through applying sentiment analysis on each occurrence of each topic. A sentiment score is averaged and whether this score is positive or negative dictates the stance of the news company towards those topics.

Both topic modelling and sentiment analysis require the use of machine learning in order to automatically process the data at a large scale. The topic modelling algorithm of choice that is used within this thesis will be latent Dirichlet allocation topic modelling which is a popular approach within machine learning to identify topics within documents. A popular definition of LDA is as follows, "Each document can be described by a distribution of topics and each topic can be described by a distribution of words" [8].

As a basic example, if a media company receives a score of 0.9 towards a topic such as a political party, this means that it is very likely that the news company desires to represent this political party in a positive light and biased towards this party. On the other hand, if a score of 0.1 is generated this means that the news company is biased and against this party.

A neutral base line will also be developed. Naturally, a score of 0.6 does not necessarily imply that a news company is biased as there are many factors within language that can slightly skew the score. This base line has the potential to extend further and appropriate research will need to be dedicated in understanding this base line.

Topic Modelling and Sentiment Analysis within news media

“Computer-Assisted Content Analysis: Topic Models for Exploring Multiple Subjective Interpretations” (Chuang J, Wilkerson J, Weiss R, Tingley D, Stewart B, Roberts M, et al.) [33] is a paper written and published by the Princeton university. This paper discusses computer aided topic modelling from the perspective of allowing experts to interact with the creation of topic models in order to increase accuracy, reliability and adaptation of automatic model creation.

The paper discusses the limitations of topic modelling such as how multiple runs of a topic model can result to having multiple different solutions due to the underlying optimization [9]. Another point of conflict of topic modelling is that the output of models must frequently be manually updated which can introduce significant changes in the output produced as well as it has the possibility to replace the initial coding instructions that have been set up by the expert. Experts have rejected up to two thirds of the machine-generated topics [10].

Solutions are then presented in order to increase the usability of topic modelling in order to maximise its advantages. The first solution is to create high quality iterative models through the exploration of model space suited for their specific type of analysis. Unless this specific model can be manually constructed, manual coding will prevail in terms of accuracy, although it will still lag behind in being able to analyse large amounts of texts as content analysis is notoriously time and labour intensive.

The article also hypotheses that the visualisation of the topic model will reduce selection bias by the individual that writes the coding scheme. Generally, when picking codes in thematic analysis bias can occur when picking certain coding scheme and through the use of developing automatically generated coding schemes a large exposure to more coding schemes will increase the reliability of the coding implemented by the analyser.

A conference paper with the title “Large-Scale Sentiment Analysis for News and Blogs” [11] is a project that shares a great deal of similarities this dissertation. This paper investigates the creation of binary sentiment scores, either positive or negative, in order to dictate how each mention of a topic in corpus is presented.

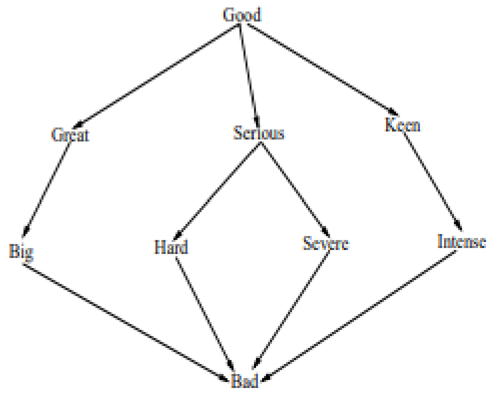


Figure 3 Sentiment Analysis Graph Hops [11]

An interesting conclusion drawn from the above conference paper is taking into account subjectivity scores. The amount of subjectivity associated with each entity or topic is calculated through reading all news texts over a period of time and counting the sentiment in each occurrence in order to get the average subjectivity for that topic. This subjectivity can be evaluated through the follow calculation and acts as the base line for calculating sentiment scores where the topic is mentioned.

$$world\_subjectivity = \frac{total\_sentiment\_references}{total\_references}$$

Figure 4 Calculating Subjectivity Score for Topics [11]

Another recent article published at the start of 2019, which was briefly mentioned within the introduced, presents an implementation of LDA Topic Modelling that attempts to retrieve any discussion of thirdhand smoke within the Chinese Media [5]. The purpose was to discover how often the Chinese Media discusses this issue, how well it is discussed and how this has changed over time. A total of 2000 articles were recovered through the Wiser and Factiva databases.

An LDA topic modelling approach was implemented in order to discover the main keywords and topics discussed within thirdhand smoke articles. The percentage distribution for topics discussed are shown in the graph that has been added below (Figure 3) and the words that link directly to the topic have been added in the second graph (Figure 4).

The topics generated through the LDA model were performed on articles that only discussed their hand smoke and through using the word frequency distribution the LDA model successfully managed to create the words associated with the imaginary topic and its correct topic correlation.

Findings were able to determine that from 2013 to 2017 the number of articles written about thirdhand smoke has decreased from 78 a year to 41 a year within Chinese as opposed to the US where the number of articles increased from 52 to 105 respectively [5].

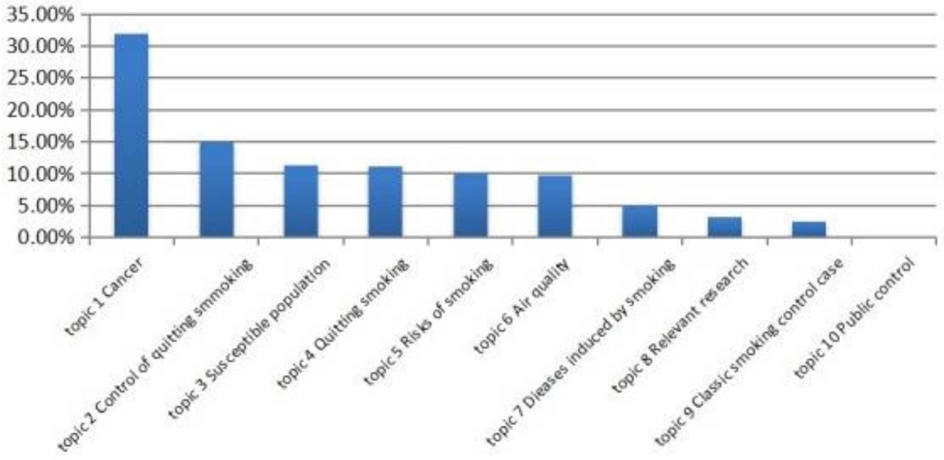
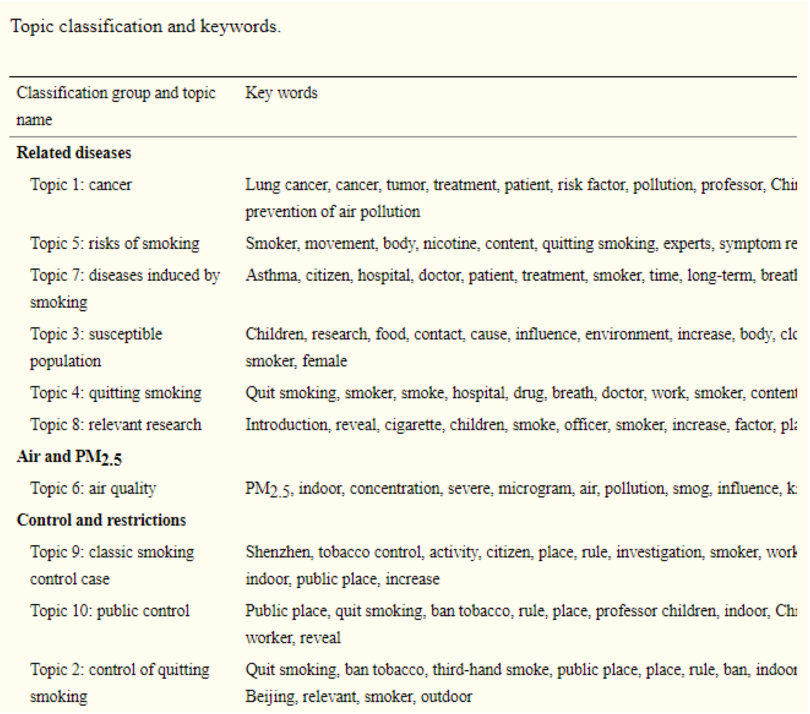


Figure 5 Thirdhand Smoking News Topic Model [5]



Topic classification and keywords.

Classification group and topic name	Key words
Related diseases	
Topic 1: cancer	Lung cancer, cancer, tumor, treatment, patient, risk factor, pollution, professor, Chi prevention of air pollution
Topic 5: risks of smoking	Smoker, movement, body, nicotine, content, quitting smoking, experts, symptom re
Topic 7: diseases induced by smoking	Asthma, citizen, hospital, doctor, patient, treatment, smoker, time, long-term, breat
Topic 3: susceptible population	Children, research, food, contact, cause, influence, environment, increase, body, cl smoker, female
Topic 4: quitting smoking	Quit smoking, smoker, smoke, hospital, drug, breath, doctor, work, smoker, content
Topic 8: relevant research	Introduction, reveal, cigarette, children, smoke, officer, smoker, increase, factor, pl
Air and $PM_{2.5}$	
Topic 6: air quality	$PM_{2.5}$, indoor, concentration, severe, microgram, air, pollution, smog, influence, k
Control and restrictions	
Topic 9: classic smoking control case	Shenzhen, tobacco control, activity, citizen, place, rule, investigation, smoker, work indoor, public place, increase
Topic 10: public control	Public place, quit smoking, ban tobacco, rule, place, professor children, indoor, Chi worker, reveal
Topic 2: control of quitting smoking	Quit smoking, ban tobacco, third-hand smoke, public place, place, rule, ban, indoor Beijing, relevant, smoker, outdoor

Figure 6 Topic to key word link [5]

Bias Detection

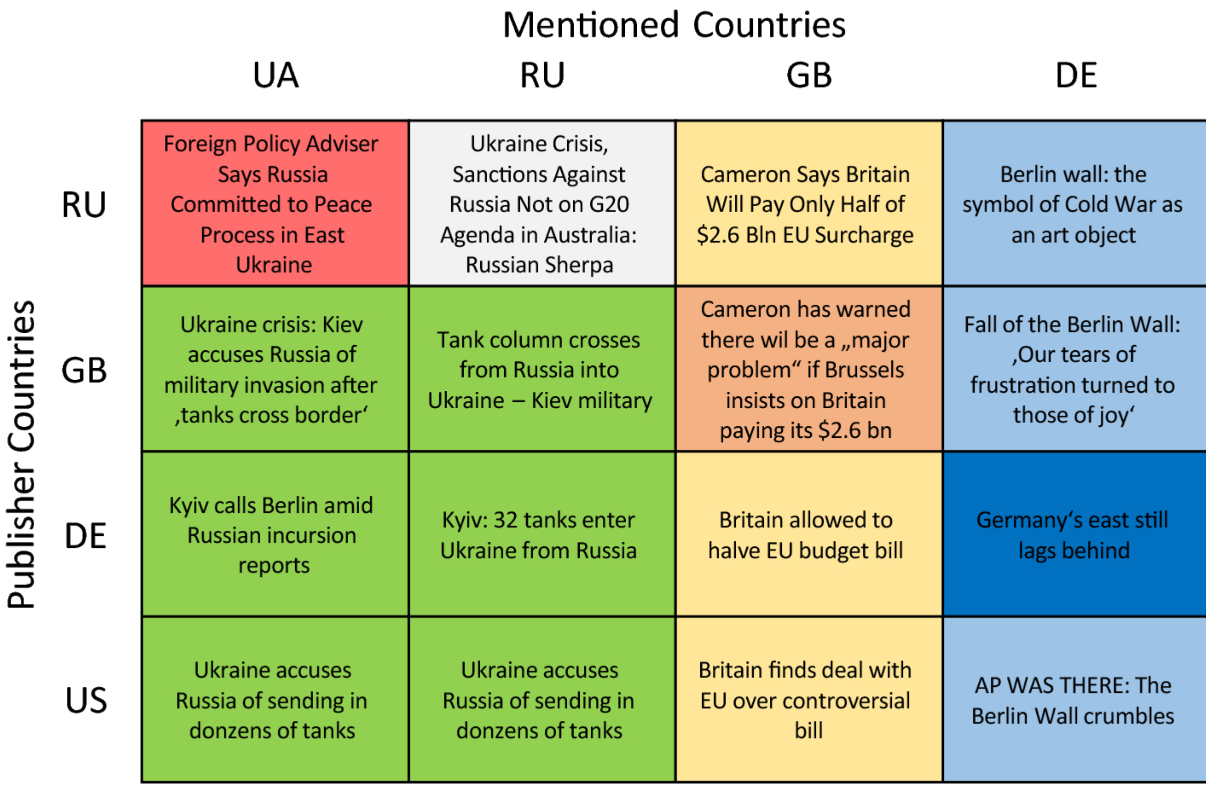
Media bias detection in news articles is one of the most prominent areas of growth within sentiment analysis. This area comprises of determining if certain news companies display bias towards certain topics. Bias analysis is generally performed through the use of content analysis. Marta Recasens, an expert in natural language processing, developed a method to detect bias using a dictionary of words that is also used by Wikipedia editors to ensure articles conform to Neutral Point of View (NPOV) rules [12].

One such journal article named "Automated identification of media bias in news articles: an interdisciplinary literature review" [13] examines bias detection through comparing by what means various news outlets report events differently. No physical implementation of such a system is developed within this article, but an in-depth explanation of how automating such an approach could work is provide. This article also delves into the advantages of automating such an approach in order to escape the time-consuming process of manually linking events to a baseline.

This article explains that it first derives a set baseline through articles which are generally perceived to be the most objective, such as police reports [14]. Events that are reported by multiple media outlets are then specifically chosen to be analysed. Two studies discovered that the volume of participants and the event type such as protests against legislation had a high impact on the number of news coverage by different media organisations [15].

To automate the above approach a sequence of steps was required. Firstly, relevant articles on events must be recovered, the articles must then be linked to the baseline data such as police reports and then statistics must be computed on the linked data.

The diagram below demonstrates the events that were examined based on different publishers across countries. The event in this scenario was where a publishing company in a country mentioned one of the four countries below (USA, Russian, Great Britain and the United Arab Emirates and Ukraine).



		Mentioned Countries			
		UA	RU	GB	DE
Publisher Countries	RU	Foreign Policy Adviser Says Russia Committed to Peace Process in East Ukraine	Ukraine Crisis, Sanctions Against Russia Not on G20 Agenda in Australia: Russian Sherpa	Cameron Says Britain Will Pay Only Half of $2.6 Bln EU Surcharge	Berlin wall: the symbol of Cold War as an art object
	GB	Ukraine crisis: Kiev accuses Russia of military invasion after ‚tanks cross border‘	Tank column crosses from Russia into Ukraine – Kiev military	Cameron has warned there wil be a „major problem“ if Brussels insists on Britain paying its $2.6 bn	Fall of the Berlin Wall: ‚Our tears of frustration turned to those of joy‘
	DE	Kyiv calls Berlin amid Russian incursion reports	Kyiv: 32 tanks enter Ukraine from Russia	Britain allowed to halve EU budget bill	Germany‘s east still lags behind
	US	Ukraine accuses Russia of sending in donzens of tanks	Ukraine accuses Russia of sending in donzens of tanks	Britain finds deal with EU over controversial bill	AP WAS THERE: The Berlin Wall crumbles

Figure 7 Publisher Bias Detection [13]

Another research projected developed within the Samsung R&D Institute focused on the implementation of a system that acts as a bias awareness news recommendation system. This system was built on the premise of scraping multiple news articles on a variety of topics from various news sources and then performing clustering on similar topics in order to calculate a bias score for each topic. The user can then generate a bias score for the article they are currently reading, as well as articles that are similar out of the previously web scraped articles.

The system architecture provided by the bias detection system is presented below. Newly created news articles are firstly scraped across multiple publishers, this is represented by the “crawler” and the content extraction is performed in order to obtain the articles and metadata such as which news company reported on the article. Topic modelling is then performed to construct the topics that were discussed followed by topic indexing which is effectively mapping the real topics to the imaginary topics created. Bias computation is then performed through a REST API. Firstly, the article is extracted based on topic, data cleaning and tokenization is then performed, passed to the REST API which returns an estimated bias score. The scores are than averaged across topics and sent to a database. Queries are then performed on this database each time a client accesses an article where a similar process occurs for that single article where it is compared to the other articles that discussed the same topic.

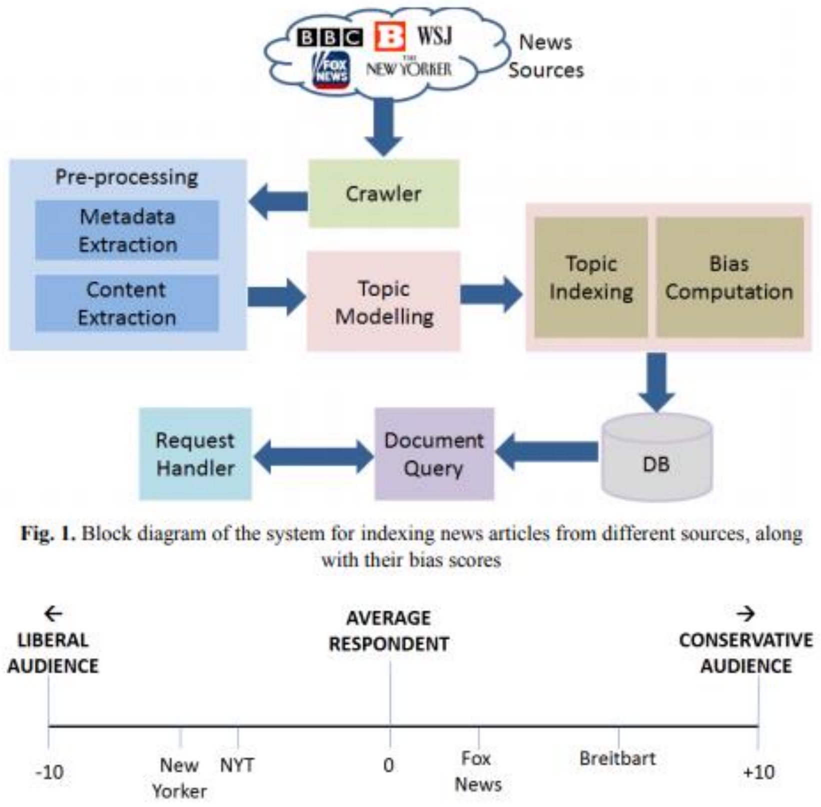


Fig. 1. Block diagram of the system for indexing news articles from different sources, along with their bias scores

Figure 8 System Architecture for Bias Detection [16]

Fake News Detection

“Is the News Deceptive? Fake News Detection Using Topic Authenticity” [17] is a paper that proposes an approach in order to detect fake news in online social media using a machine learning classifier. The machine learning classifier detects certain key pieces of information within the account in order to check if the account posts fake news.

The approach firstly checks if the profile and background image or description indicate that the news company supports one side on a controversial issue as many fake news sites have this characteristic. If this is found to be true the news company is automatically assigned as fake news. The next step checks if the profile description indicates it is a news feed. If the profile is a verified account it is legitimate, but if it is not tweets should seem to be objective and have retweets cited from reliable accounts. There is some discrepancy as to whether or not the account can then be labelled valid or not as the news account could be very small and missing retweets from verified accounts as few people are subscribed and regularly read the accounts posts.

On the one hand the approach applied to tweets within this project would not directly be applicable to the system described in this dissertation as it attempts to use specific information from twitter such as retweets as well as profile images which do not exist for popular newspaper companies. On the other hand, the machine learning and classification model has multiple points of interest that are still similar to newspaper stance detection. The project proposes a novel approach in finding similarities between fake news accounts such as average post length, average retweets and linking predications between max and total friends of accounts. The application within stance detection could be to build a similar model in order to find similarities with newspaper companies that are biased. These similarities could also be factors such as common headline terms, average article length and average number of topics discussed. This approach has a large amount of potential as no approach uses this metadata in order to build bias predication and while it may be inaccurate at times it has the potential to introduce novelty to the approaches that have already been developed.

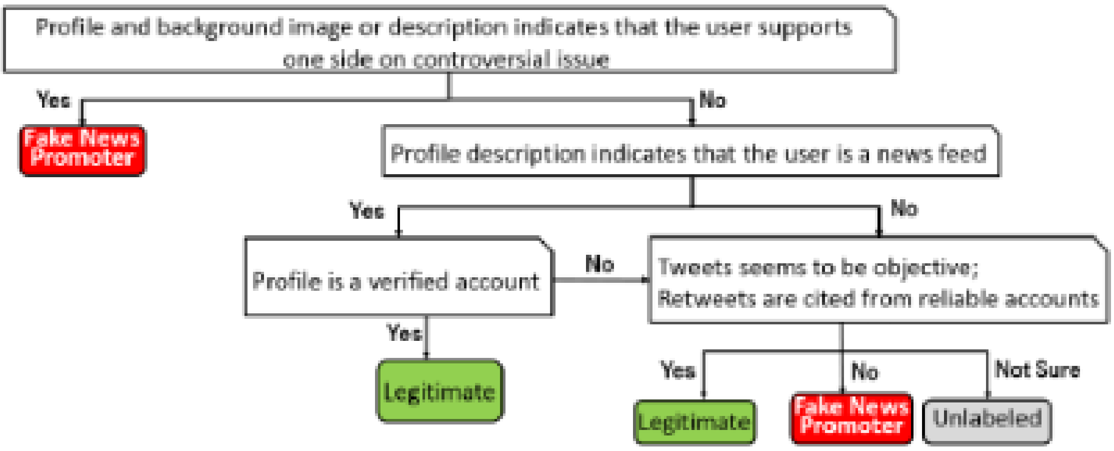

Profile and background image or description indicates that the user supports one side on controversial issue
Yes
No
Fake News Promoter
Profile description indicates that the user is a news feed
Yes
No
Profile is a verified account
No
Tweets seems to be objective; Retweets are cited from reliable accounts
Yes
Legitimate
Yes
No
Not Sure
Legitimate
Fake News Promoter
Unlabeled

Figure 9 Manual Labelling [17]

2.3. Technologies you've researched

As per the discussion within the introduction, the key objectives are to implement topic modelling, data cleaning and a sentiment analysis that can be performed at large scale among thousands of articles. Consequently, requirements such as natural language processing libraries for data cleaning, lexical databases for understanding how closely words are linked and API's for connecting to real Irish news source databases were investigated within this section.

The technologies researched consist of natural language processing libraries, existing lexical databases which apply components of text analysis, qualitative research approaches and API's that can scrape newspaper articles.

Natural Language Processing Libraries

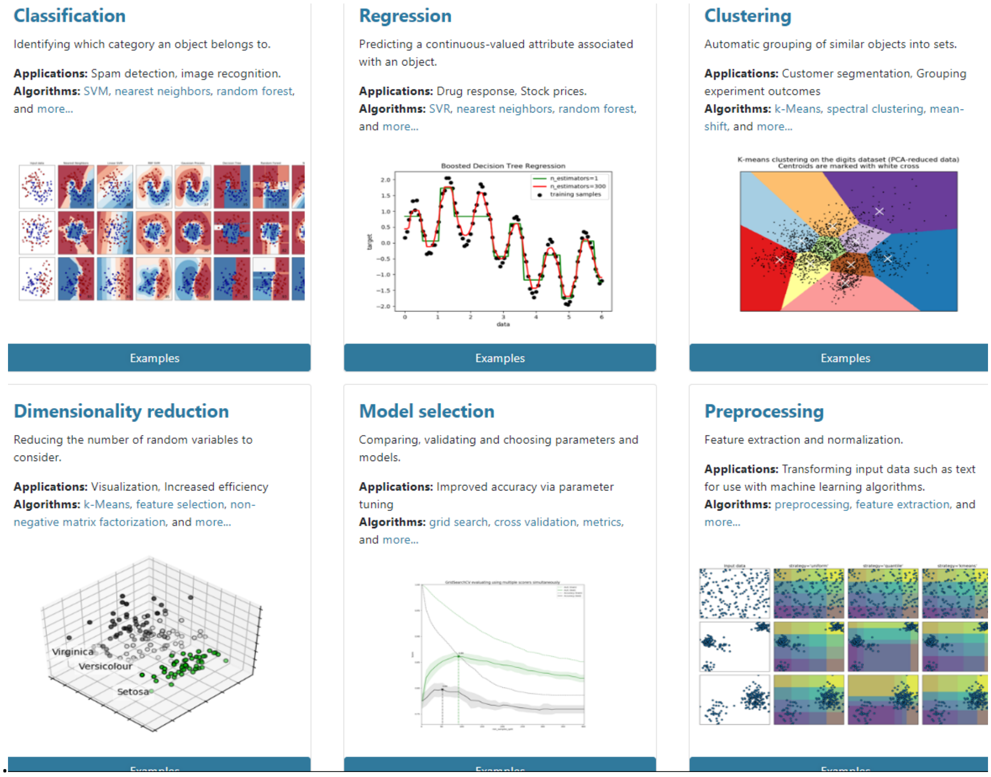
Python is at the forefront of natural language processing (NLP) and there is a plethora of different NLP libraries that are integrated with python. Different libraries have different advantages such as performance, accuracy and a more extensive feature lists, while others have specific trade-offs. The scope of this section is to discuss the most viable natural language processing libraries and single out the libraries that are most suited to this system.

The first natural language processing library that was uncovered within the research phase was NLTK or the natural language toolkit which was developed in python. This library provides a large suite of libraries for symbolic and statistical natural language processing within the English language. Advantages of this library include consistent use of data structures, extensibility to other third-party NLP languages, modularity between interactions of the components within the system and a through documentation. [18]

“SpaCy” is one of the fastest NLP languages that integrates the C programming language for some of the more process heavy tasks. A paper written in 2018 investigating “SpaCy” performance and made the follow claim. “We employ the POS Tagger of SpaCy, in preference to the CMU TweeboParser, due to the heavy processing time of the latter. The TweeboParser was 1000 times slower as opposed to SpaCy” [19]. Spacy also provides more advanced features than libraries such as “TextBlob” such as entity linking and working with vectors.

“TextBlob” is an NLP library that extends the NLTK library and provides additional functionality such as noun extraction, word tagging and text classification. “TextBlob” is generally the library of choice for beginners within the area of NLP as NLTK itself can become very tedious for completing even simple tasks, whereas “TextBlob” can achieve even more complex functionality through a simple interface, the draw back being that TextBlob is slower than NLP languages such as SpaCy and less advanced in the feature suite that it provides. [20]

Scikit-learn is another NLP library that performs a wide array of both natural language processing and machine learning. This library focuses more than any of the others on machine learning having natural integration with classification, regression, clustering and model selection. The advantages of this library are the wide array of various applications to machine learning.



Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...

Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: k-Means, feature selection, non-negative matrix factorization, and more...

Examples

Model selection

Comparing, validating and choosing parameters and models.

Applications: Improved accuracy via parameter tuning
Algorithms: grid search, cross validation, metrics, and more...

Examples

Preprocessing

Feature extraction and normalization.

Applications: Transforming input data such as text for use with machine learning algorithms.
Algorithms: preprocessing, feature extraction, and more...

Examples

Boosted Decision Tree Regression
n_estimators=1
n_estimators=300
training samples
target
data
K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross
Virginica
Versicolour
Setosa

Figure 10 Sci-learn Machine Learning Applications [21]

Gensim is an open-source library for unsupervised topic modelling and natural language processing implemented in python. As seen in the previous section, a great deal of research projects uses genism in order to establish various models within natural language processing. Genism also has integration in order to assist in the extraction of semantic topics from documents in an efficient process. Gensim can work both with topic modelling and sentiment analysis with it's well optimized Word2Vec and Doc2Vec support.

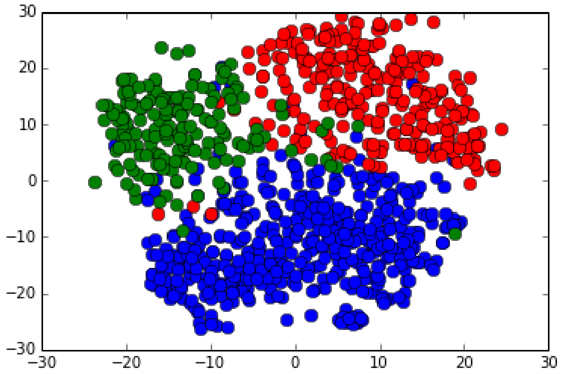
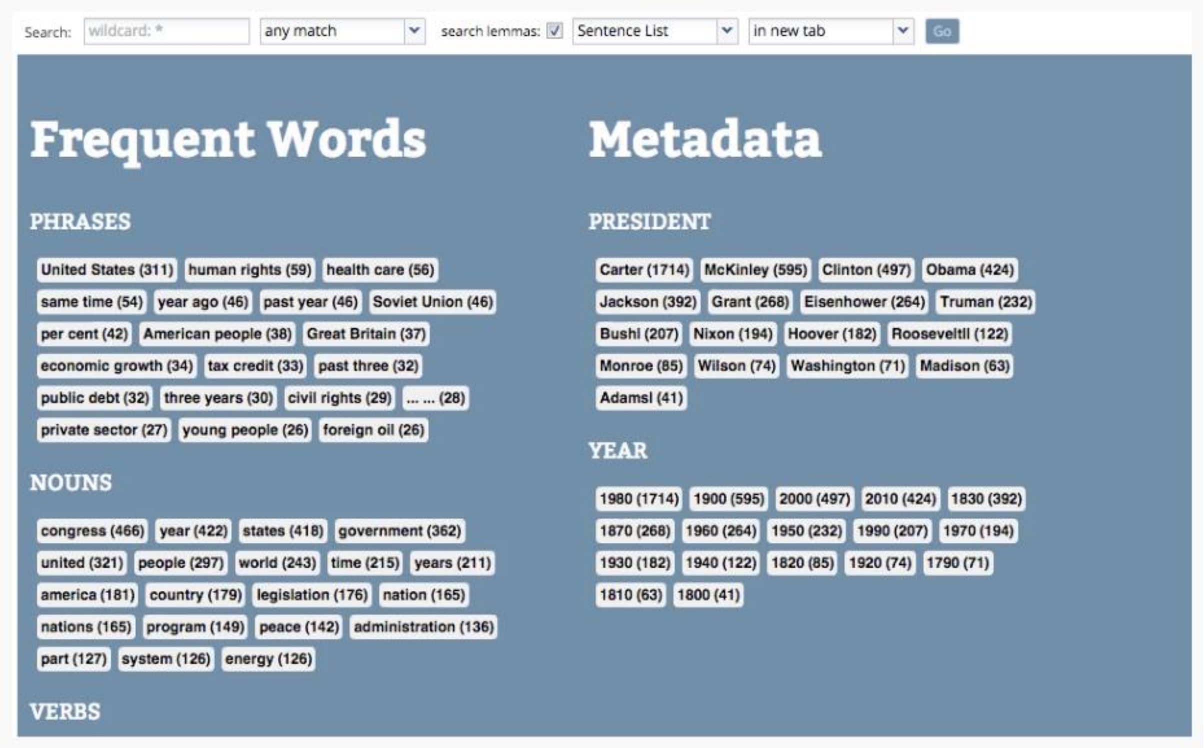


Figure 11 Gensim Classifier with food words (blue), sports words (red) and weather words (green) [22]

Natural Language Tools

Wordnet is a large English lexical database containing nouns, verbs, adjectives and adverbs which are grouped into sets of cognitive synonyms (synsets), each expressing a specific concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. [23] A lexical database such as wordnet is required in NLP as data cleaning processes such as lemmatization require breaking down each word into its normal form, removing prefixes and suffixes. This is a complex task that requires the assistance of such a lexical database.

WORDSEER is another online resource with an exposed API that performs textual analytics and visualisation to make text navigable and accessible. It splits phrases, nouns, years and even specific metadata such as the names of presidents from within texts.



Search: wildcard: * | any match | search lemmas: ☑ | Sentence List | in new tab | Go

Frequent Words

PHRASES

United States (311) human rights (59) health care (56)
same time (54) year ago (46) past year (46) Soviet Union (46)
per cent (42) American people (38) Great Britain (37)
economic growth (34) tax credit (33) past three (32)
public debt (32) three years (30) civil rights (29) ... ... (28)
private sector (27) young people (26) foreign oil (26)

NOUNS

congress (466) year (422) states (418) government (362)
united (321) people (297) world (243) time (215) years (211)
america (181) country (179) legislation (176) nation (165)
nations (165) program (149) peace (142) administration (136)
part (127) system (126) energy (126)

VERBS

Metadata

PRESIDENT

Carter (1714) McKinley (595) Clinton (497) Obama (424)
Jackson (392) Grant (268) Eisenhower (264) Truman (232)
Bushl (207) Nixon (194) Hoover (182) RooseveltII (122)
Monroe (85) Wilson (74) Washington (71) Madison (63)
Adamsl (41)

YEAR

1980 (1714) 1900 (595) 2000 (497) 2010 (424) 1830 (392)
1870 (268) 1960 (264) 1950 (232) 1990 (207) 1970 (194)
1930 (182) 1940 (122) 1820 (85) 1920 (74) 1790 (71)
1810 (63) 1800 (41)

Figure 12 WORDSEER Visualisation of Text

API's

For data training the "20 News Group Dataset" "BBC dataset" were considered. The "20 News Group Dataset" was chosen as a result of it having a simpler structure allowing it to be more easily parsed.

For acquiring real life datasets several API's responsible for retrieving newspaper articles were investigated. The most powerful was "newsapi.org" that links directly to Irelands live top and breaking news headlines and articles. The data recovered is in json but an issue that was discovered with this API is that there is a limit of 500 requests per day and only articles up to one month old can be retrieved for the free version. This may not be a crucial issue as within one month of making 500 daily requests, over fifteen-thousand news articles from Irish papers can be retrieved. [24]

Another news API is the "MyAllies Breaking News" API that provides the access to real time news from across the glove. This API is completely free to use on RapidAPI and overcomes the previous limitation, but it does not have access to articles older than a single day.

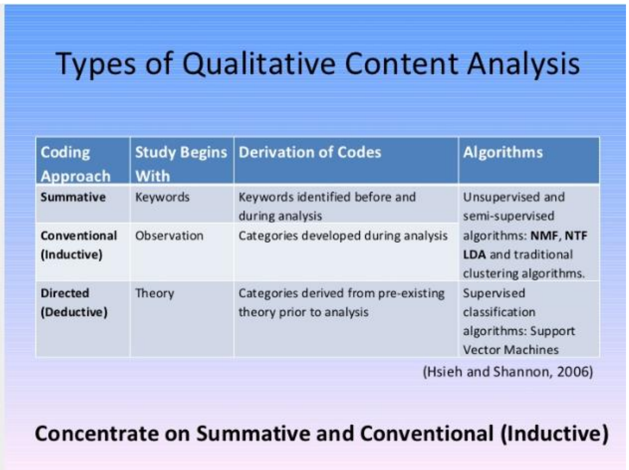
A possible solution to overcome the limitation of a maximum of 500 requests a day is to implement a web scraper that can that can retrieve the textual information from articles daily.

2.4. Other Research you've done

Qualitative Analysis Research

A great deal of work has been performed in the area of qualitative research in order to facilitate a better understanding of how extrapolating meaning from newspaper articles falls within this area to create an LDA topic model. Qualitative research is empirical research where the data are not in the form of numbers. [25]

The diagram below illustrates the different types of qualitative content analysis methods and the complementary algorithms that are directly linked. Summative uses keywords, conventional uses categories developed during analysis and directed is developed from categories created before the analysis.



Types of Qualitative Content Analysis

Coding Approach	Study Begins With	Derivation of Codes	Algorithms
Summative	Keywords	Keywords identified before and during analysis	Unsupervised and semi-supervised algorithms: **NMF, NTF LDA** and traditional clustering algorithms.
Conventional (Inductive)	Observation	Categories developed during analysis	
Directed (Deductive)	Theory	Categories derived from pre-existing theory prior to analysis	Supervised classification algorithms: Support Vector Machines

(Hsieh and Shannon, 2006)

Concentrate on Summative and Conventional (Inductive)

Figure 13 Types of Qualitative Content Analysis [26]

The method of qualitative data that seemed to apply the most to the area of creating topics through analysing word frequency patterns was thematic analysis. Thematic analysis is a method of content analysis which contrasts with other qualitative analytic approaches such as discourse analysis and grounded theory which are generally considered to be methodologies more so then methods. Thematic analysis emphasizes identifying, analysing and interpreting patterns of meaning (or "themes") within qualitative data. [27]

The basic key summary of the thematic method is as follows. Thematic analysis is implemented through the process of creating codes or words which will match to a theme. The codes are then later organised into themes. [28] An illustrative example is where the theme is “Phone Brands” and the codes are Nokia, Samsung and iPhone.

Once this process is implemented it can then be applied to interviews or conversations where each time the code word is mentioned the algorithm is able to find the mention of this keyword and sort it within the topics for further analysis. LDA Topic Modelling is essentially an altered automatic thematic analysis which can dynamically create these themes and match them to the words based on the patterns within word frequency. By being able to automatically create these topics the quality of the topics can be assured as a machine can consider many more articles than an individual could have read. The topics are also created based on frequency and as a result a machine algorithm is more likely to create the most relevant topics.

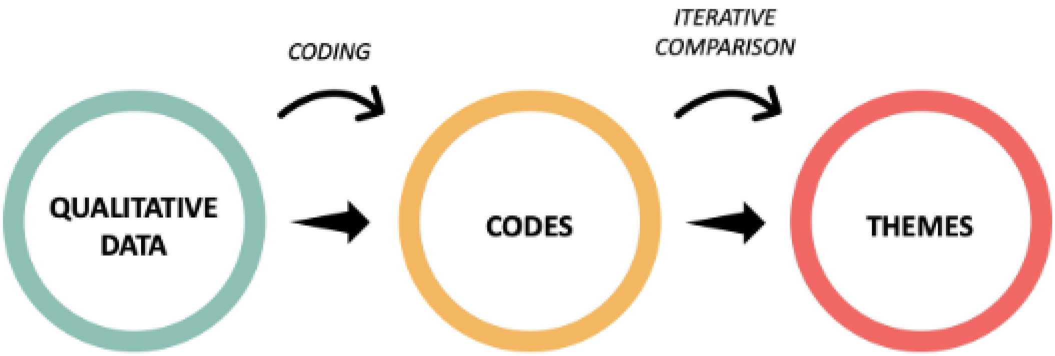


Figure 14 Thematic Analysis Process [29]

Bias in Media Research

Research in the area of the exact steps of how bias occurs within media is a quintessential step in understanding how to build bias detection for newspaper companies. This research aids in mapping the sentiment score to each topic as it allows the creation of an estimation of how much bias naturally occurs within the media.

The diagram below shows the process of how bias is formed within the media. It firstly breaks down the factors that influence the medias perception. The political and ideological view of the company naturally influence how various topics are represented and this consequently will affect and skew the representation of political parties and movements through bias. However, factors such as advertisements will skew sentiment greatly. Different articles can quickly switch between persuasive and informative quickly, where persuasive should generate outliers that have a very high sentiment score and informative could have an overly prevalent neutral score.

Owners will affect in how articles report on events as the news company might not report negatively on an event involving one of their partners, advertisers or sponsors. Another large factor that influences media representation is target audience. Target audience is a major factor as generally papers attempt to cater to the views and beliefs of the group that reads the paper in order to continue their readership of the paper and as a result this is another factor that effects media bias.

As can be seen in the centre of the diagram there are a lot of natural factors that further effect bias without considering opinionated bias. There can be discrepancies between different news companies and the gathering of data. Different news companies may be mis informed or miss key facts that will affect the writing style and generate more negative sentiment scores that other news companies even if they share the same perspective. Writing style can be more negative or positive for certain journalists as writing styles different. Local colloquialism would also make the sentiment analysis incapable of classifying the expression and generating a score as it may not have classified such phrases.

Finally, editing is a major contributor as it can both reduce and increase bias. Some papers may specifically remove certain words that are overly positive or negative with a more informative style of writing was required. Another factor within editing that effects bias detection is where miss spelled words are missed. Sentiment analysis cannot work on words that are miss spelt and as a result no sentiment score can be generated, and bias cannot be caught or dismissed.

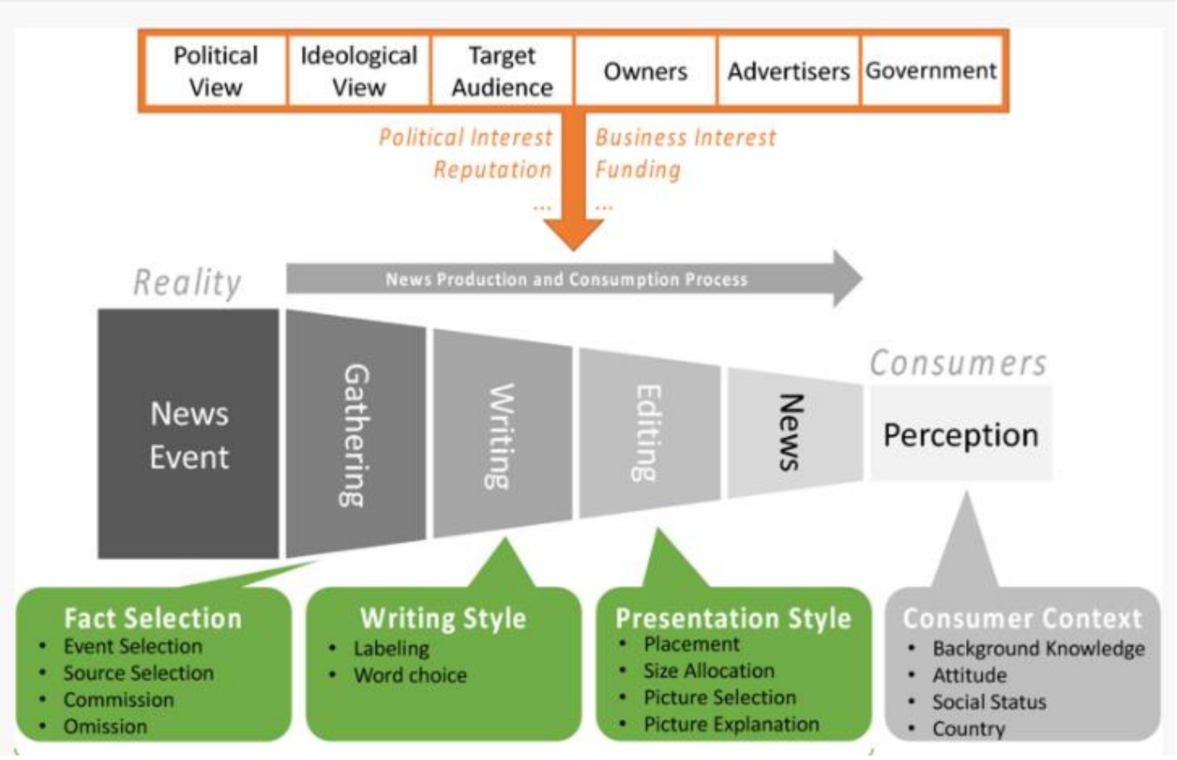


Figure 15 Bias within Media [13]

2.5. Existing Final Year Projects

Several final year projects from previous years were looked at in the research phase of the project. The elimination process for choosing final year projects to investigate was finding similar themes such as natural language processing, sentiment analysis and machine learning

Sentiment and Mood Interpreter for Logging Emotions

This final year project primarily focused on sentiment analysis for positive and negative moods. The sentiment analysis was performed on facial recognition (which was found to be 79% accurate) and diary entries (which was found to be 40% accurate). A large portion of the project also focused on the design, layout and usability of the application itself.

The application was designed using feature driven development with agile methodologies. Similarly, for the topic modelling within this dissertation, an iterative approach will also be essential as it will demonstrate a steady growth in the accuracy of the topics built. The feature driven approach will not work as well for this final year project as it will essentially focus on a few features and the complexity will follow from how they link together. For example, linking sentiment and topic analysis to determine sentiment towards certain topics.

Testing was implemented using percentage scores for facial / emotion detection and sentiment analysis. An iterative approach was then followed for monitoring how the percentage score changed as the analysis became more accurate. Edge cases that reduced the accuracy of the results were also flagged and different scores were built around them. An example of such an edge case is where the facial recognition is used in a dark room and in this scenario the accuracy of the facial recognition would have its own independent score rating.

NLPurchase – eCommerce Chatbot Final Year Project Report

This final year project makes use of natural language processing in order to make a chatbot that can successfully communicate with a customer on an eCommerence website. Natural language techniques such as lemmatization and removing stop words are used in order to break down the contents of the users input and allow a more standard, readable approach. In this newspaper topic and sentiment analysis thesis these data cleaning techniques are crucial, particularly for the topic modelling where thousands of newspaper article will each individually be scanned into memory and then split into a very large list of words where further analysis algorithms will be completed to extract meaningful topics and sentiment scores.

The design of the final year project followed an incremental cycle in order to allow the project to adapt to change, as well as provide continuous prototypes with simple complexity that gradually evolved over time. This design allows the project to be split into multiple stages, avoiding architectural risks very early in development as no crucial decisions are made very early in the analysis of the project.

Testing was performed continuously though user integration. A mixture of informal and formal testing was used. The informal testing was where multiple users between the ages of eighteen and twenty-four used the chatbot and multiple stages of its development and cycle and were proposed to fill in their criticisms, issues and proposed solutions when using the chat bot. The users would also give overall usability scores. The formal testing was implemented through a survey which contained questions based on the chatbot design guidelines. These guidelines compromised general usability and feature functionality.

2.6. Conclusions

With the knowledge and conclusions drawn from the background research the designing and development of the system can begin. The crucial learning from this research provided a stronger understanding of the current technologies within this area as well as their advantages and disadvantages. Furthermore, a better grasp of what can currently be achieved with LDA Topic modelling and stance detection within the media has also been ascertained.

This chapter has also further established the requirements necessary that will be discussed within the following chapter. Natural language Processing techniques will need to be designed in order to introduce an efficient and powerful data cleaning algorithm and a suitable LDA model will need to designed in order to produce accurate topics from the distribution of words.

The large scope of projects presented within this field also demonstrates how this area is a topical subject. Multiple dissertations, papers and conferences over the last couple of years have been dedicated to further exploring this topic.

3. Prototype Design

3.1 Introduction

Linking directly to the incremental development methodology, prototyping is a technique which breaks a project into smaller portions with the scope of enabling simpler requirements. The strength of such a process is exemplified by Jason and Smith [42] where they comment on the users inability to identify their own requirements and as a result the need to exhibit the requirements of an experimental system such as a prototype in order to be able to more accurately estimate the requirements and the feasibility of the system.

Connecting back to previous chapter where the key background research was presented the themes will be continued in this design chapter. The first section will look at the software methodologies employed in the creation of this project and then an overview and use cases will be created. The following section will analyse the system architecture from the perspective of the front-end, middle-tier and back-end.

3.2. Software Methodology

Multiple software methodologies were considered before arriving to a conclusion. Some of the methodologies considered includes the spiral model, waterfall, feature-driven development, extreme programming, and Kanban. Methodologies such as the waterfall method were instantly eliminated due to their well-known limitation of being unable to adapt to a changing in the requirements. As a result, the methodology of choice for this thesis was strictly required to be agile in nature and only the agile mythologies were seriously considered and examined.

Spiral model

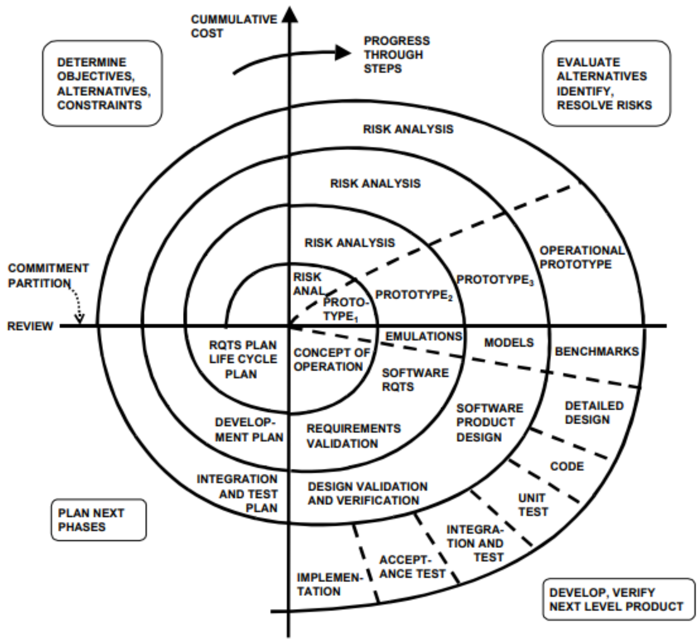


Figure 16 Spiral Model [32]

The spiral model is an agile methodology that is provides a several useful advantages that fit this project. It has one of the best risk analysis models as it is performed iteratively on each iteration. It also provides a realistic implementation as the project moves through loops in a spiral to be completed.

The disadvantages on the other hand bring in a costly model where a great deal of time is spent on each stage of the design and iteration process. The risk analysis also requires a highly specific expertise in order to design and deeply analyse each iteration. The spiral model is also particularly fine-tuned for large projects that span multiple teams and since this project will be independently completed it may not be suitable [35].

Feature-driven development model

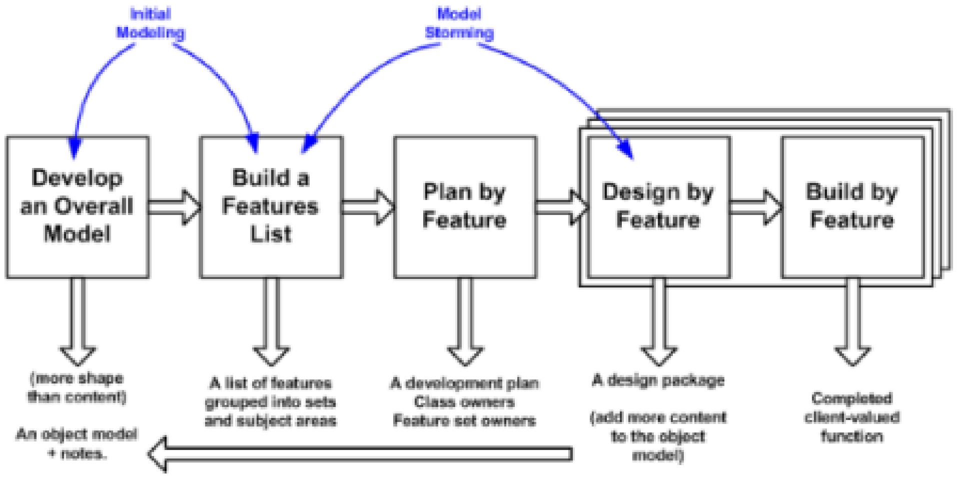

Initial Modeling
Model Storming
Develop an Overall Model
Build a Features List
Plan by Feature
Design by Feature
Build by Feature
(more shape than content)
An object model + notes.
A list of features grouped into sets and subject areas
A development plan Class owners Feature set owners
A design package
(add more content to the object model)
Completed client-valued function

Figure 17 Feature-driven development model [36]

Feature-driven development supports industry recognized best practices, breaking down large projects into smaller features and a simple five step process than does not require a specific expertise. Disadvantages include an iterative process that is not as well defined as other agile methods and generally this methodology is not suited to a single software developer, but multiple teams working in parallel.

A key aspect of feature-driven development is its emphasis on communication between teams and collaboration between users. This thesis is not an application and will not go deeply within the area of human computer interaction. As a direct result a lot of the design principles behind feature-driven development stray away from this projects core focus and would not be suitable for this thesis.

Extreme Programming

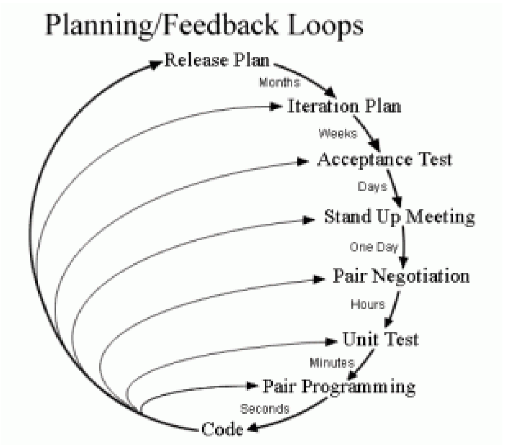
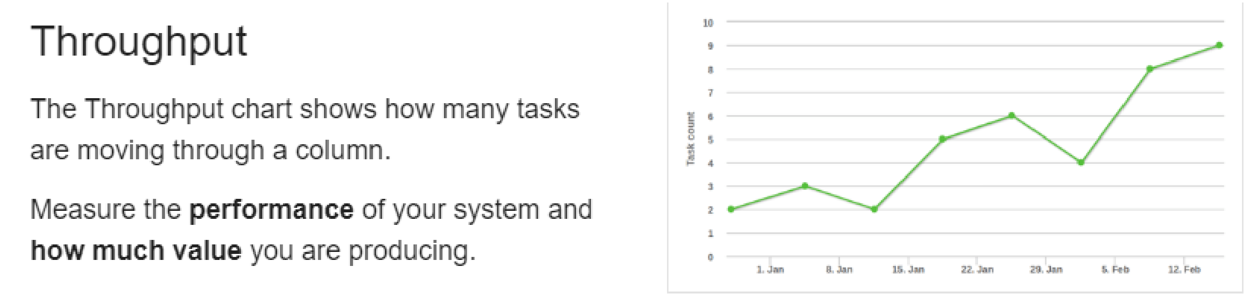


Figure 18 Extreme Programming Model [30]

Extreme programming is an agile methodology that follows specific programming guidelines such as test-driven-development, continuous integration and prototyping. This methodology focuses far more on writing code than in very short iteration cycles where short tests are continuously integrated within the development process and while the design is left more lacking than the previous methodologies the level of testing done acts as its own documentation and analysis. Disadvantages include a requirement for skilled programmers that can conceptualize a large portion of the design and work independently or in pairs. Another disadvantage is the reduced level of design may lead to a larger risk within the project's development for edge cases.

Kanban

Kanban is an agile methodology that splits tasks into achievable blocks. It encourages continuous integration where all the work and progress are reviewed daily, providing powerful goal orientation and progress reports. These reports provide meaningful data that allows the developer to stay on track with continuous planning and analysis of progress,

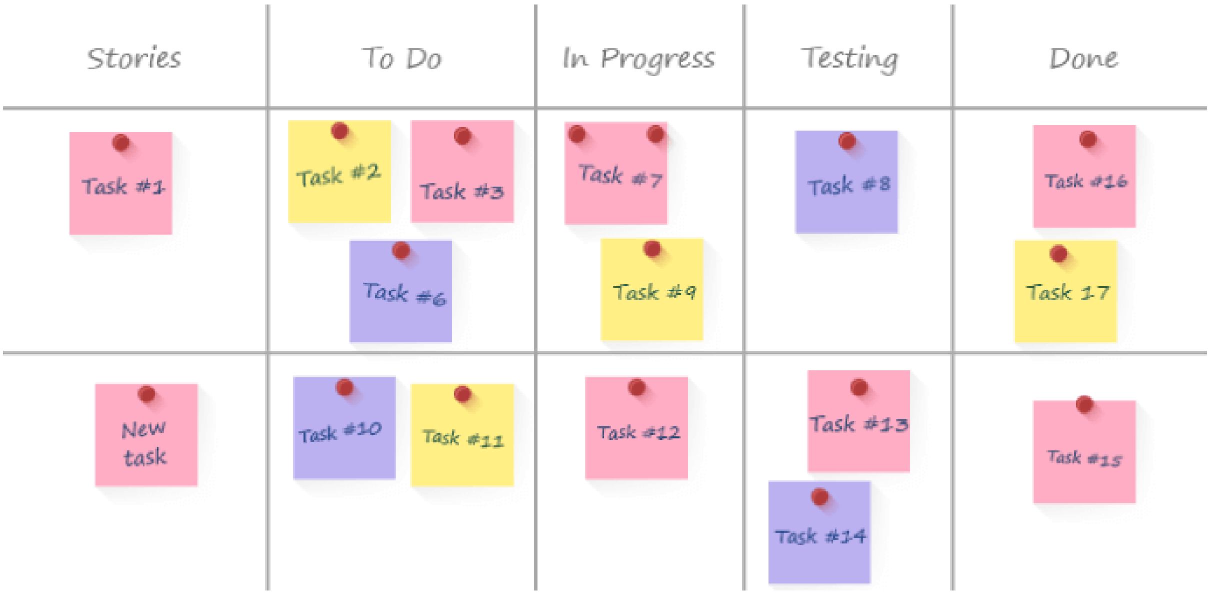


Throughput

The Throughput chart shows how many tasks are moving through a column.

Measure the **performance** of your system and **how much value** you are producing.

Figure 19 Kanban Progress Chart



Stories	To Do	In Progress	Testing	Done
Task #1	Task #2 Task #3 Task #6	Task #7 Task #9	Task #8	Task #16 Task 17
New task	Task #10 Task #11	Task #12	Task #13 Task #14	Task #15

Figure 20 Kanban Example [31]

CRISP- DM

The Cross Industry Standard Process for Data-Mining is a model that is commonly used to solve machine learning problems [44]. This is a fine-tuned model that focuses on understanding data, pre-processing data and finally training and testing the model.

This methodology fits this project very well since the majority of the complexity within this thesis resides around the main areas described by its iterations. In building the LDA and sentiment models each phase will compromise around understanding the requirements and data, cleaning and then training and testing the specified model.

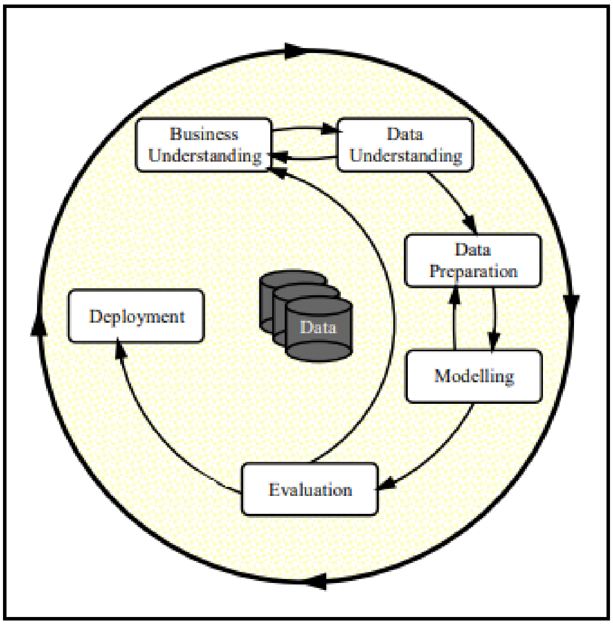


Figure 21 CRISP-DM [43]

3.3. Overview of System

A Kanban and simplified spiral model will be used where through each iteration a set of goals will be planned. The key aspects from the spiral model that will be extract are risk analysis and engineering testing methods. Kanban will be responsible for goal orientating each iteration, timeboxing using the Pomodoro technique as well as evaluating the success of each iteration continuously.

Both design and code will be delivered in stages to allow adapting to change in each iteration. The following features will need to be completed and then iterated over to be made more accurate and provide more meaningful data.

1. Sourcing of articles with various topics.
2. A data cleaning algorithm that will allow the use of natural language processing techniques on the articles sourced from newspaper companies.
3. A basic LDA model that can generate topics and then build on the accuracy, coherence and perplexity of the model through stages.
4. A mapping of the created imaginary topics to its real topic.
5. Multiple methods to visualise the topics created by the LDA algorithm.
6. A sentiment analysis model.
7. A mapping of the sentiment score to the topics.

The architecture does not rely on any client-server model. The complexity is sourced within the area of building topics from real newspapers and being able to map the topics to a sentiment score that reflects that newspaper companies' views towards these topics. There is also no database that is used as the articles for the newspapers are already stored in a textual format and it is more efficient to read them directly from the documents than build a new database.

The diagram below shows a high-level sequence of events that occur within the system when generating the sentiments of topics for each newspaper company. The processing is performed in stages where first both the training and real newspaper articled are sourced. The training data is then cleaned an LDA and sentiment model is built using this data. The models will then be used on the real newspaper articles in order to split the topics again and build a sentiment score around each topic.

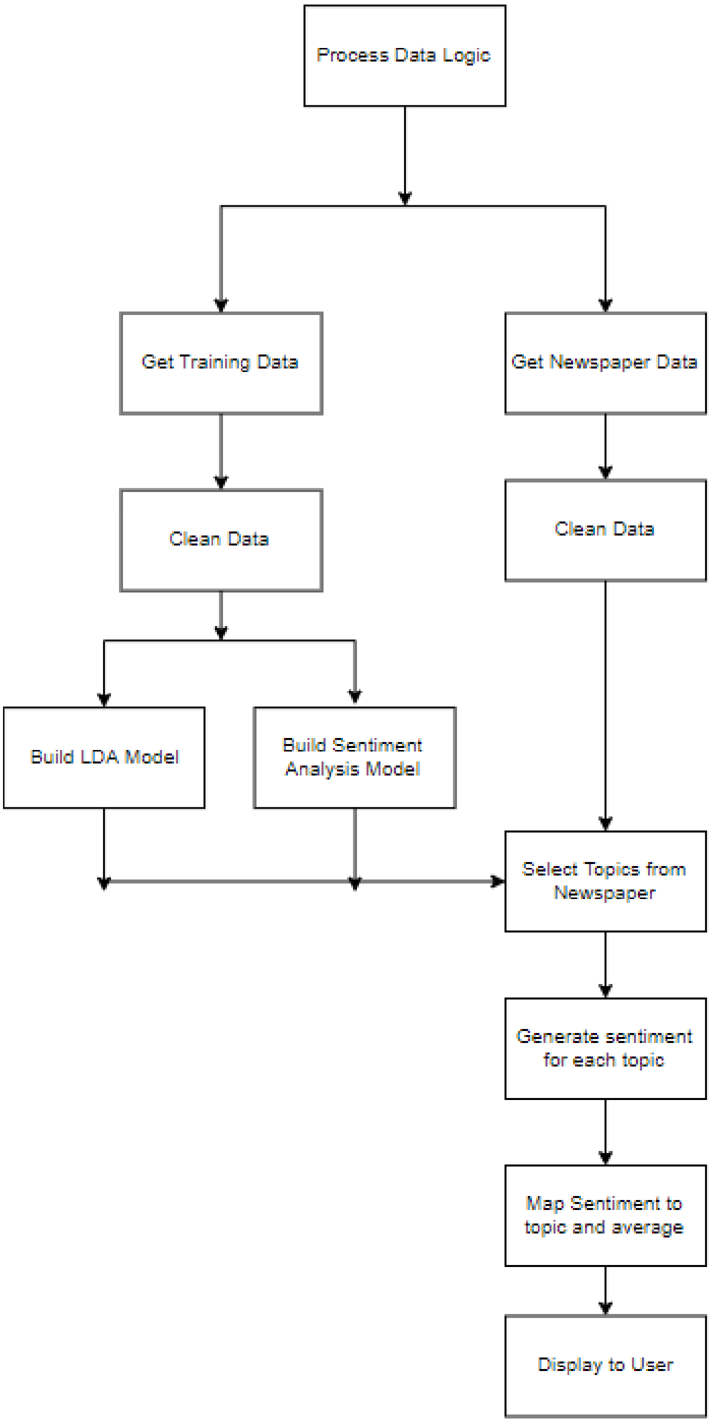

Process Data Logic
Get Training Data
Get Newspaper Data
Clean Data
Clean Data
Build LDA Model
Build Sentiment Analysis Model
Select Topics from Newspaper
Generate sentiment for each topic
Map Sentiment to topic and average
Display to User

Figure 22 System Sequence of Operation Diagram

3.4. Design of User-Display

The front-end layer will focus on how the generated topics are displayed to the user with a sentiment score for each news topics. There will be a basic CSV with all the data and there will also be a more visual representation of the topics with sentiment scores. There are also multiple third-party libraries that will assist in visualising these topics to create easy to read graphs that display all the topics with the words linking to those topics. The interface will also show the correlation between topics.

There will also be multiple graphs to show the word frequency in each topic. The choice of graph will be histogram as it is one of the most informative attempting to determine the total word frequency within a topic.

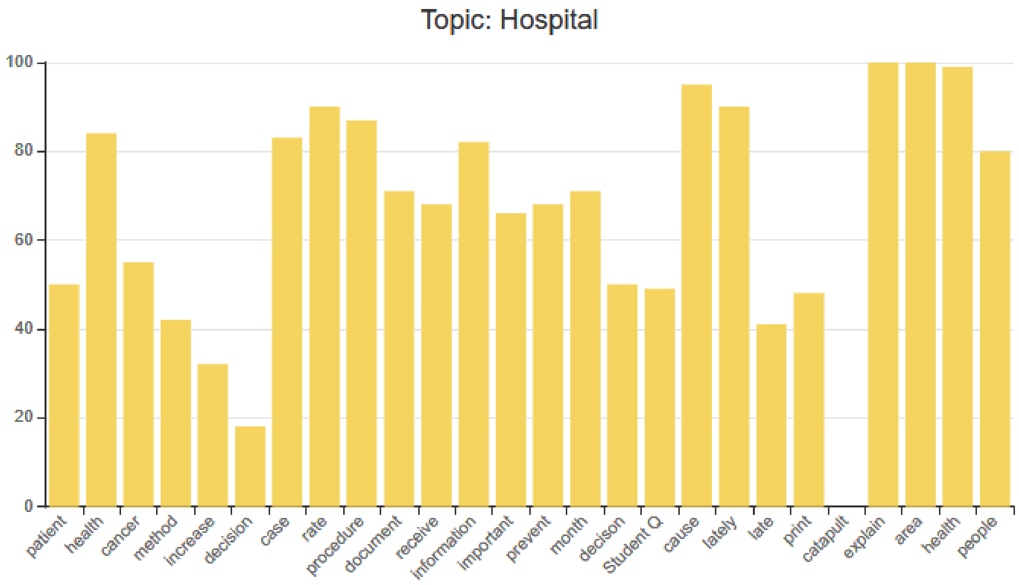


Figure 23 Word Frequency Topic Distribution Histogram

Below is a hand drawn design for how topics will be displayed. On the left a graph shows all the topics with numbers designating a topic. The larger a bubble the more commonly this topic is mentioned within the media. On the right is the display for a topic (9) which links to hospital as can be seen through the correlation of words such as patient, cancer, information etc.

Initially the LDA display does not process and correlate the imagery topic (which is the number) to the real topic as further processing and implementation is required to link it to the real topic. The distance between topics also represents how far away each topic is from each other. This is calculated using the similar words that are shared between topics. For example if two topics were football and basketball, then they would be tightly linked as multiple words would be shared between them such as “lose, win, team”.

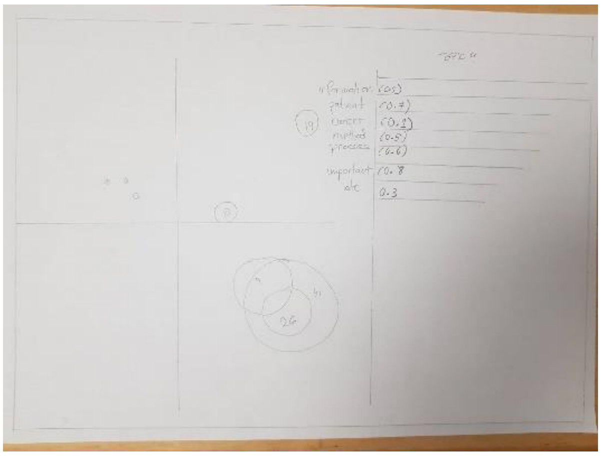


Figure 24 Hand drawn Topic Display

3.5. Design of Functionality

The use case diagram below shows the different pieces of functionality available to the user. The first option allows the user to view the overall sentiment towards a topic created by the LDA model which provides more in-depth information such as words used. The second option allows the user to see all the topics and sentiment scores generated with a basic interface. The third option retrieves the sentiment score towards a topic from a newspaper company. The fourth option allows the user to see all the sentiments scores towards all topics from that newspaper company. Finally, the last option shows all the topics without sentiment scores. This option also provides more information on the topics themselves such as what words make up that topic.

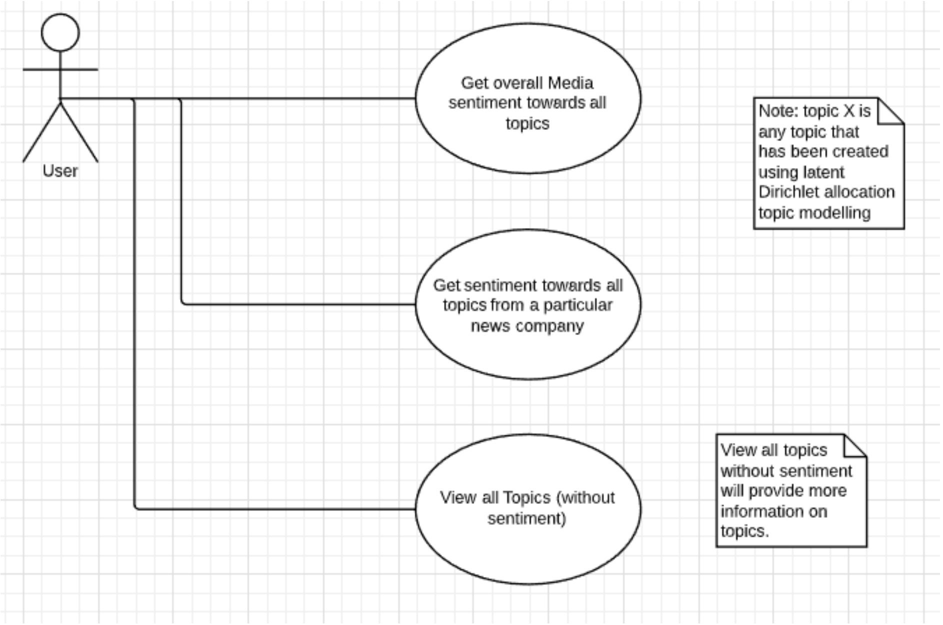

User
Get overall Media sentiment towards all topics
Note: topic X is any topic that has been created using latent Dirichlet allocation topic modelling
Get sentiment towards all topics from a particular news company
View all Topics (without sentiment)
View all topics without sentiment will provide more information on topics.

Figure 25 User use case diagram for viewing topics and sentiment in media

The middle-tier is the logic layer that drives the system and implements its core capabilities and it is this layer that the complexity from this system arises. The middle tier will implement the newspaper scraping through API's, the natural language processing, the data cleaning, the latent Dirichlet allocation topic modelling and the sentiment analysis algorithm.

A suitable method will be required to scrape newspaper articles from API's. Due to the large amount of data required it is likely a thread capable web scrapper will need to be implemented in order to decrease the time it takes to capture large amounts of data, especially in order to build the LDA Topic model.

The technical architecture for data cleaning below shows the data processing required in order to apply the LDA model as well as the sentiment analysis model. This step is pivotal in natural language processing as the data needs to be cleaned in order to efficiently and accurately create the models. Within natural language processing the more work that is performed in the data cleaning layer, the more accurate the models will be and developing a more accurate and efficient data cleaning algorithm will be a future concern in order to increase the accuracy of the model.

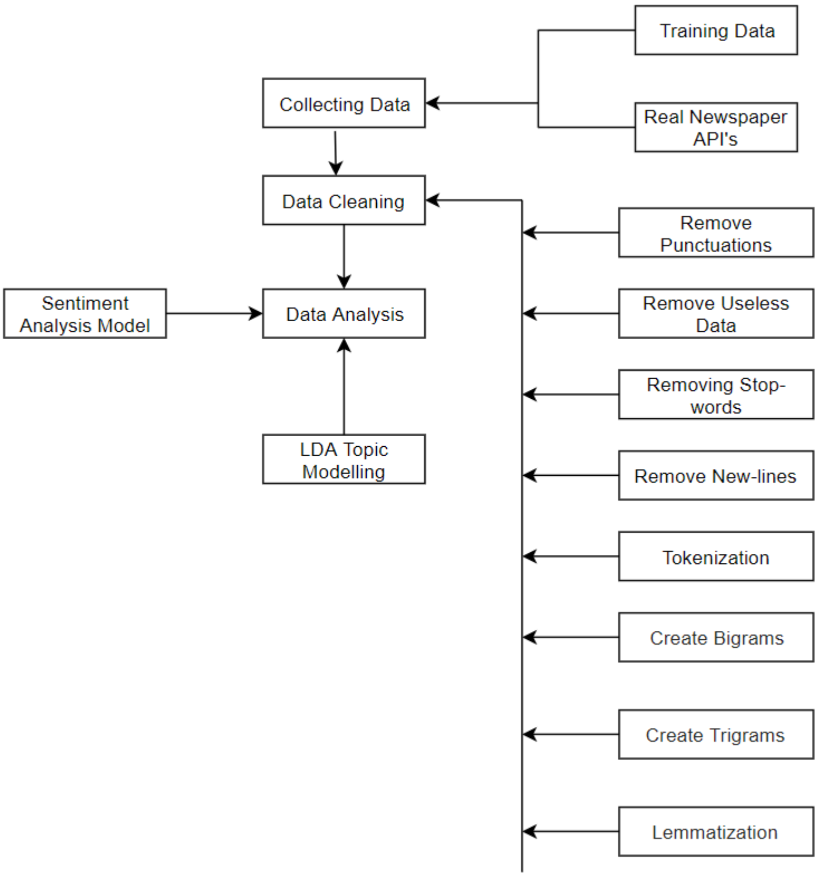


Figure 26 Process for Data Cleaning for Sentiment Analysis and LDA Topic Modelling

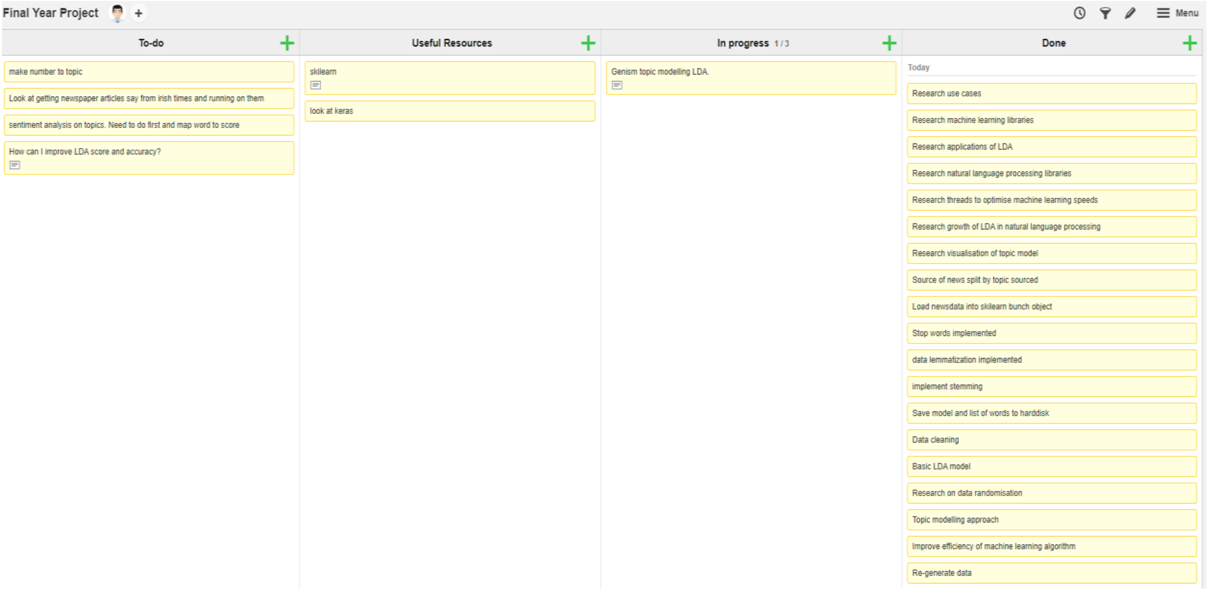
The implementation will require multiple natural language processing techniques as can be seen in the diagram above. Some of the techniques are straight forward such as removing punctuations and can be implemented with simple regex commands and other techiest require additional help from third party libraries. As an example, lemmatization will require removing all suffixes and prefixes from every word and transforming them into their neutral form. This is a complex area in natural language processing and contains an immense amount of edge cases which are cumbersome to solve manually. A more feasible solution is to investigate and find an optimal library that balances speed and efficiency in order to perform the lemmatization on every word within the newspaper articles.

Building the LDA Topic model will require the creation of a word dictionary that maps each word to an ID and corpus which maps each ID to the frequency of each word. The sentiment analysis model will need to classify each topic and generate an average score for each topic. Mapping the sentiment score to each topic will then need to be implemented. This will be difficult as the models will be created separately and some method to link them will need to be created.

This system will not require a central database. The only source of information this project requires will be newspapers to train and apply models to which are already in a textual format.

The data that is generated will be models, a list of topics and sentiment scores within CSVs and html files that will be used to display data to users more visually. Adding a database would have added a great deal of complexity to the system unnecessarily.

For the purposes of version control GitHub will be used. The iterative approach will be tightly linked to Kanban in order to keep track of progress through Pomodoro's, the tasks that need to be completed, the tasks that are completed and useful resources.



Final Year Project

To-do

make number to topic

Look at getting newspaper articles say from irish times and running on them

sentiment analysis on topics. Need to do first and map word to score

How can I improve LDA score and accuracy?

Useful Resources

skilearn

look at keras

In progress 1 / 3

Genism topic modelling LDA.

Done

Today

Research use cases

Research machine learning libraries

Research applications of LDA

Research natural language processing libraries

Research threads to optimise machine learning speeds

Research growth of LDA in natural language processing

Research visualisation of topic model

Source of news split by topic sourced

Load newsdata into skilearn bunch object

Stop words implemented

data lemmatization implemented

implement stemming

Save model and list of words to harddisk

Data cleaning

Basic LDA model

Research on data randomisation

Topic modelling approach

Improve efficiency of machine learning algorithm

Re-generate data

Figure 27 Author's Dissertation Kanban

3.7. Conclusions

Through the analysis of multiple agile methodologies, a mixed approach seems to be the most suitable for this thesis. The methodologies that will be employed include Kanban for task scheduling and progress, a more basic spiral model for risk analysis and CRISP-DM as it follows the exact approach this system will be taking for creating models.

Kanban and the spiral model will allow for early prototype development, the ability to adapt to change and monitor the completion of each task and goals as they are completed with multiple progress reports generated by a Kanban tool to monitor progress. CRISP-DM will focus on the design of each step with how data is processed and prepared.

Based on the key themes that were discussed within this chapter, the following chapter will delve into the development process and many of the issues covered in this chapter will be revisited. The development chapter will discuss the implementations of the design and any challenges or changes that developed as a result of implementation limitations.

4. Prototype Development

4.1. Introduction

This chapter will begin the outline and implementation of the final year project that was discussed and planned within the previous design chapter. The chapter will also discuss any challenges faced in the development process.

4.2. Prototype Development

For the initial prototype a decision was made to have implemented the following features. Source a suitable training dataset, implement a data cleaning algorithm in order to remove useless data that cannot be analysed from articles, implement natural language cleaning techniques in order to prepare data for creating LDA and sentiment models, create a basic corpus, word dictionary, LDA model, a method of persisting the model and dictionaries to the hard drive and finally a visualization of the topic model with word distributions and frequency of each word related to each topic. A representation of how closely linked different topics were would also be interesting in order to better understand commonly used words between certain models.

The prototype was completely implemented in the python language, version 3.6. Multiple libraries were used in order to assist in natural language processing, creating the LDA model and visualising the topic model such as SpaCy, NLTK, Gensim and pyLDAvis respectively. The LDA model's accuracy score was computed using both a coherence and perplexity scores.

As discussed in the previous design section the methodologies employed will be a mix of Kanban, spiral model. CRISP- DM is also a new methodology that was discovered after the design phase which is a widely used methodology within machine learning as it follows the pattern of understanding data, pre-processing data and training and testing the model.

4.3. User-Display

The user-display below shows all of the created imaginary topics, labelled as numbers. The display was created with the assistance of the pyLDAvis.gensim library which was specifically created in order to provide access to powerful methods for visualising topic models.

Each number within the graph represents an imaginary topic with a list of words that relate to that topic. At the moment, no successful algorithm has been implemented for the prototype which maps the number to its real-life topic, but based on the word distribution of a topic the real-life topic is easily identifiable by the individual using the graph. As an example, the topic which has been clicked is marked in red (42) and all of the words that are associated with this topic can be seen on the right of the graph. Words such as information, patient, cancer, cause and case instantly can be related to a hospital or health. Therefore, the number 42 represents this real-life topic and multiple algorithms are being tested in order to map the topic number to its real-life name using word distribution and machine learning.

The distance between topics represents how closely linked different topics are to each other. As an explicatory example the diagram below which represents the topic 26 is completely covered within topic 41s circle. This represents that there is a strong link between the two topics and when examining the contents of the two topics they share many words, however the words that are used are generic in 41 (would, go, make, think) and in 26 the words that are used are presumably from political articles where this more common direct speech from anecdotes is present (people, group, believe, speak, die) where many generic words are often tightly coupled. The interface shown below matches the hand drawn graph which was the proposed method for displaying topics in the design chapter.

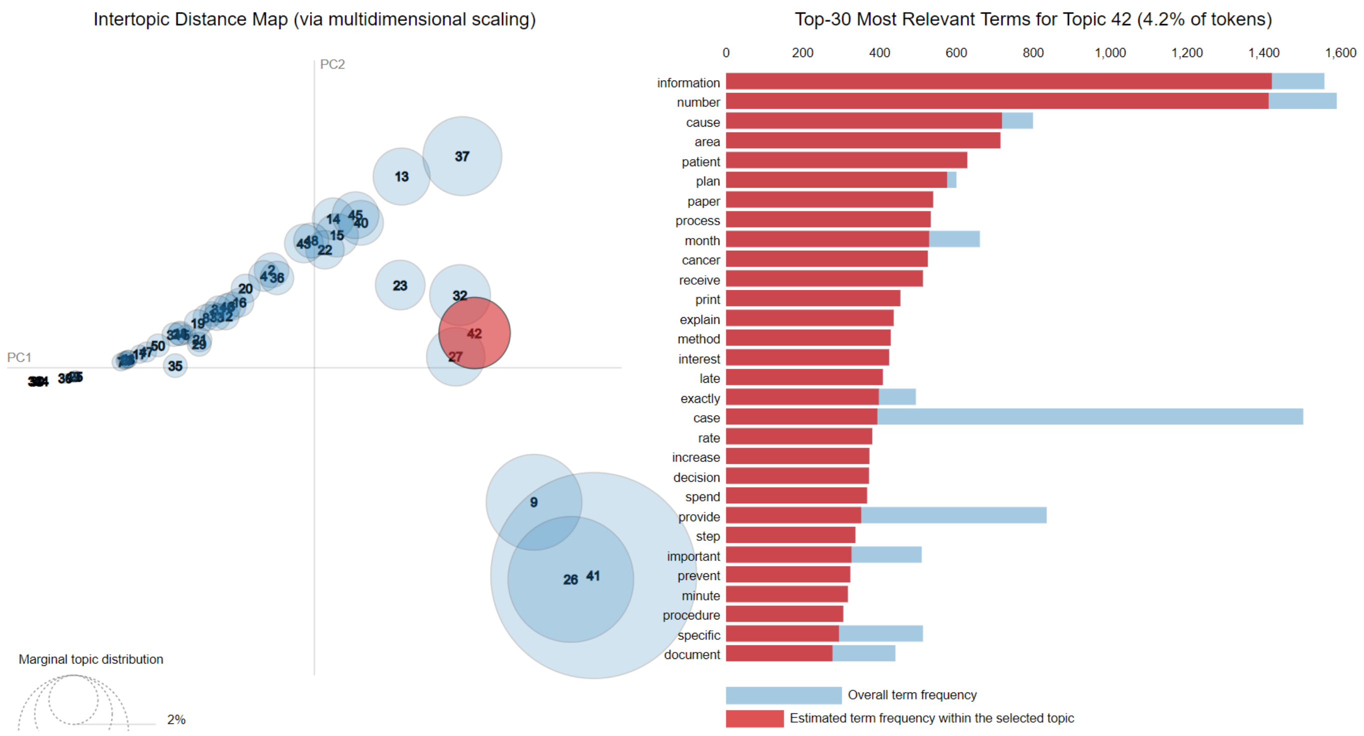


Figure 28 Topic Modelling Visualisation for Topic 42 (Hospitals)

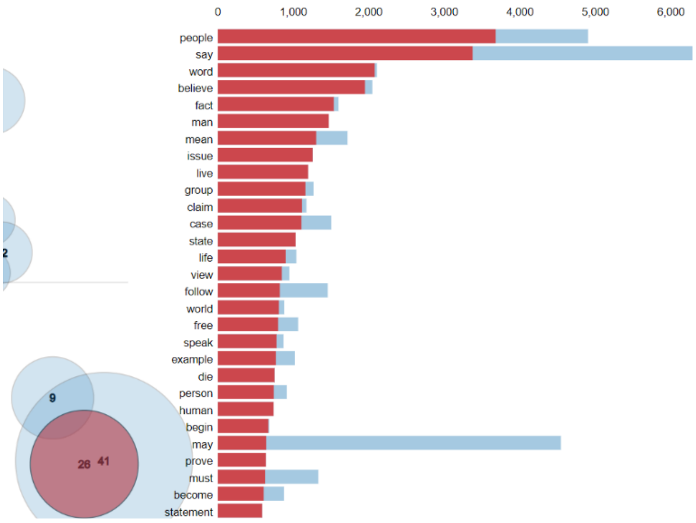


Figure 29 Topic Modelling Visualisation for Topic 26 (Politics)

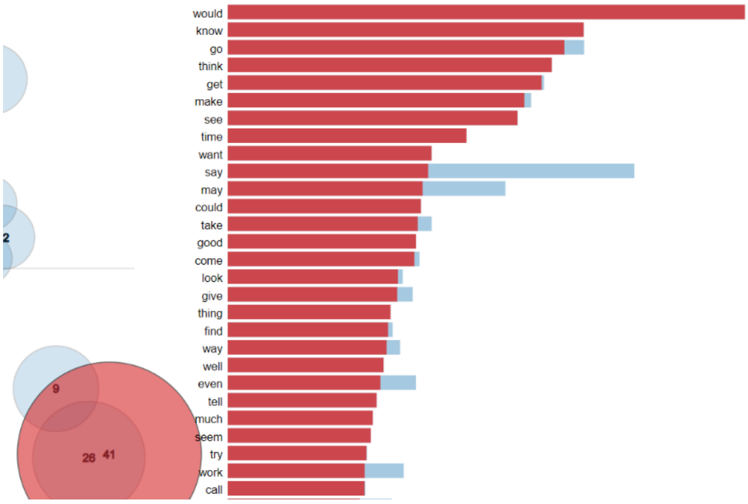


Figure 30 Topic Modelling Visualisation for Topic 41 (Generic Words)

4.4. Functionality

The GitHub repository that contains the source code, models, html visualisation file and corpus can be found at the following link: https://github.com/MichaelLenghel/FinalYearArticleLDAPrototype

Various challenges were encountered within the development process. Initially the model computed a very low coherence and perplexity score meaning that the quality of topics was very low. In order to fix this issue a more powerful, exhaustive data cleaning algorithm was required. Methods introduced include removing special symbols, creating bigrams which combine two words that are commonly used together into one word. The data size for building the model was also increased to over seven thousand in order to further increase the accuracy of the model. These issues were not encountered during the design phase as model accuracy was not exhaustively discussed and consequently these solutions were developed within the development process.

Another issue that was encountered was the time it takes to create a model. For a relatively small data set of 7000 articles the LDA model, as well as data cleaning took over 20 minutes for competition. In order to increase the efficiency different libraries were looked into that were more optimised for natural language processing. The library that was previously mentioned within the background research phase was SpaCy which has a large portion of its functionality implemented in the C programming language. The library that was used prior was exclusively sklearn which had a slower implementation of the data cleaning algorithms since it is completely implemented within the python language. Certain regex expressions were also further simplified and less exhaustive approaches to remove all useless characters were created.

The first step for the development of the prototype was sourcing the training data. As discussed in the background research, different sources for this data were investigated such as the “20 News Group Dataset”, the BBC dataset and real time article API’s. After further investigation of both sources the “20 News Group Dataset” was chosen as it had a larger number of articles and was also split into two folders which each had an equal distribution of topics. This may come in handy if in the future a supervised machine learning implementation is implemented as the BBC dataset did not provide this functionality.

For the implementation of retrieving the dataset the file names were retrieved using the relative directory of the script in relation to the corpus folder. This means that the script can be ran from anywhere with the command line and the all of the relevant files will still be successfully found. The generate_lda script retrieves the data set and generates a bunch object to store the data using sci-learn library. The respective categories are created and a random state shuffles the data recovered. The visualise.py scripts recovers the saved model that the previous script generated.

generate_lda.py:

```
def gen_bunch(news_path):
    # Cateogies in data set
    news_categories = ['alt.atheism', 'comp.graphics', 'comp.os.ms-
windows.misc'
                    , 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.w
indows.x'
                    , 'misc.forsale', 'rec.autos', 'rec.motorcycles', 'rec.sport.b
aseball'
```

```
                , 'rec.sport.hockey', 'sci.crypt', 'sci.electronics', 'sci.med
'
                , 'sci.space', 'soc.religion.christian', 'talk.politics.guns'
                , 'talk.politics.mideast', 'talk.politics.misc', 'talk.religio
n.misc']

    # Setup path to test corpus
    NEWS_GROUPS_TEST_PATH = os.path.abspath(os.path.join(os.path.dirname(__fil
e__), news_path))

    # Print the path
    # print(NEWS_GROUPS_TEST_PATH)


    ##### Need to implement a method for including custom categories! ####

    # Load all test data.

    # news_test = load_files(NEWS_GROUPS_TEST_PATH, description='News Paper Te
st Topics from 20 news groups'
    #                            , categories=news_categories, load_content=T
rue , shuffle=False, encoding='latin1'
    #                            , decode_error='strict')

    # Shuffling the data in order to increase distribution of topics and not o
verly simplify NLP patterns
    # news_test.data is everything in one big string
    news_test = load_files(NEWS_GROUPS_TEST_PATH, description='News Paper Test
 Topics from 20 news groups'
                                , categories=news_categories, load_content=Tru
e , shuffle=True, encoding='latin1'
                                , decode_error='strict', random_state=30)

    # Note:
    # Shows the topic and document ID + the article.
    # print(news_test.filenames[0])
    # print(news_test.data[0])

    # Get all of the file names
    # for integer_category in news_test.target[:10]:
    #     print(news_test.target_names[integer_category])

    return news_test
```

As discussed within the design phase, a method for cleaning and processing the data from the news articles was then implemented through regex operations, natural language processing techniques such as creating bigrams, removing stop words and lemmatizing words.

```
def multiple_replacements(article):
    empty_str = ""

    # Replacing all dashes, equals, cursors
    replacements = {
        "-" : empty_str,
        "=": empty_str,
        "^": empty_str,
    }

    # Replace newlines
    article_list = re.sub('\s+', ' ', article)

    # Replace emails
    article_list = re.sub('\S*@\S*\s?', '', article)

    # Replace quotes
    article_list = re.sub("\'", "", article)

    # Create a regular expression using replacements and join them togetehr.
    # re.compile creates the pattern object
    # re.escape avoids using special characters in regex
    reg = re.compile("(%s)" % "|".join(map(re.escape, replacements.keys())))

    # For each match, look-up corresponding value in dictionary
    return reg.sub(lambda value: replacements[value.string[value.start():value
.end()]], article)


def split_to_word(articles):
    # Iterate over every article
    for article in articles:
        # Yield to not overload the memory with the big data set.
        # Deacc parameter removes all punctuations as well as spliting each wo
rd.
        yield(gensim.utils.simple_preprocess(str(article), deacc=True))

def create_bigrams(articles, bigram_model):
    return [bigram_model[article] for article in articles]

def remove_stopwords(articles):
    return [[w for w in simple_preprocess(str(article)) if w not in stop_words
] for article in articles]

def lemmatize_words(bigram_model):
    # Only considers nouns, verbs, adjectives and adverbs
    return [[w for w in lemmatize(str(article))] for article in bigram_model]

# This method is about a minute faster for a data set of 7000 than the one abo
ve
def lemmatization(articles, allowed_postags=['NOUN', 'ADJ', 'VERB', 'ADV']):

    articles_lem = []

    # Load the spacy lammatixation model for english
    spacy_lem = spacy.load('en_core_web_sm', disable=['parser', 'ner'])

    for article in articles:
        w = spacy_lem(" ".join(article))
        articles_lem.append([token.lemma_ for token in w if token.pos_ in allo
wed_postags])

    return articles_lem
```

Words are tokenized and a bigram model is created in order to understand which words are commonly used together and link them appropriately. This is implemented before the create bigrams method is called:

```
   # brigrams model
    # Need bigrams as it cuts word that go together down into one
    bigram = gensim.models.Phrases(article_word_list, min_count=8, threshold=1
00)

    bigram_model = gensim.models.phrases.Phraser(bigram)
```

Lemmatization is performed on nouns, verbs, adjectives and adverbs as can be seen within the code below:

```
    # Lemmatize - By default only nouns, verbs, adjectives and adverbs
    # lemmatized_article = lemmatize_words(bigram_words)

    lemmatized_article = lemmatization(bigram_words, allowed_postags=['NOUN',
'VERB', 'ADJ',  'ADV'])
```

The corpus and word dictionary to map topics to an ID and ID's to their frequency is then created with the following code:

```
    # Create dictionary. This maps id to the word
    word_dict = corpora.Dictionary(test_data_cleaned)

    # Create corpus. This directly contains ids of the word and the frequency.
    corpus = [word_dict.doc2bow(data) for data in test_data_cleaned]
```

The LDA model is then constructed using the corpus and word dictionary. Each document is iterated through ten times, the data is shuffled in chunks of 100 documents and fifty topics are created as follows:

```
def build_lda_model(articles, word_dict, corpus):
    # Build LDA model
    # Retry with random_state = 0
    lda_model = gensim.models.ldamodel.LdaModel(corpus=corpus,
                                                id2word=word_dict,
                                                num_topics=50,
                                                random_state=100,
                                                update_every=1,
                                                chunksize=100,
                                                passes=10,
                                                alpha='auto',
                                                per_word_topics=True)
```

The model is then saved through:

```
def save_model(lda_model):
    # Below doesn't work due to access denied issues, datapath is the alternative

    # MODEL_PATH = "../models/"
    # model_file = os.path.abspath(os.path.join(os.path.dirname(__file__), MODEL_PATH))

    model_file = datapath("model")
    lda_model.save(model_file)
```

The model is read from hard disk using:

```
# Get the model from genism path
def retrieve_modal():
    # Path to model
    model_file = datapath("model")
    # Load
    lda = gensim.models.ldamodel.LdaModel.load(model_file)

    print('Finished retrieving model...')
    return lda

def retrieve_word_article_list():
    MODEL_PATH = "../newspaper_data/newspaper_list.txt"
    newspaper_list_file = os.path.abspath(os.path.join(os.path.dirname(__file_
_), MODEL_PATH))

    MODEL_PATH_WRITE = "../newspaper_data/newspaper_list_writing.txt"
    newspaper_list_file_write = os.path.abspath(os.path.join(os.path.dirname(_
_file__), MODEL_PATH_WRITE))

    newspaper_article_list = []
    newspaper_word_list = []

    with open(newspaper_list_file, 'r') as filehandle:
        for line in filehandle:
            # String splitting removes first bracket and new line + closing br
acket
            current_line = line[1:-2]

            # Split each word into the list
            current_list = current_line.split(', ')

            for word in current_list:
                # Append the word and remove closing and opening quotation
                newspaper_word_list.append(word[1:-1])

            newspaper_article_list.append(newspaper_word_list)
            newspaper_word_list = []

    print('Finished retrieving article word list...')
    return newspaper_article_list
```

The coherence and perplexity scores are generated through:

```
def compute_complexity(article_word_list, word_dict, corpus, lda):


    # Coherence Score: 0.4132613386862506
    # Coherence score is the probability that a word has occured in a certain
topic, so the quality of the topic matching.
    coherence_model_lda = CoherenceModel(model=lda, texts=article_word_list, d
ictionary=word_dict, coherence='c_v')
    coherence_lda = coherence_model_lda.get_coherence()
    print('\nCoherence Score:', coherence_lda)

    # Perplexity: -21.294153422496972
    # Calculate and return per-
word likelihood bound, using a chunk of documents as evaluation corpus.
    # Also output the calculated statistics, including the perplexity=2^(-
bound), to log at INFO level.
```

```
    print('Perplexity:', lda.log_perplexity(corpus))  # a measure of how good the model is. lower the better.
```

4.6. Conclusions

In conclusion the development of the prototype specifically focused on creating and visualising the LDA model as well as sourcing training data and preforming natural language cleaning on the data in order to prepare the model. Within the next section the methods used to test and evaluate this model will be presented and discussed.

5. Testing and Evaluation

5.1. Introduction

This chapter will discuss the methods for testing and evaluating the created topic models that were discussed in the previous development chapter.

5.2. Plan for Testing

Within the current testing has only been achieved through monitoring the increase within the accuracy of the model using coherence and perplexity scores. As the system increases in both size and complexity more testing will be introduced as is described below.

The project will be tested iteratively throughout the development process. Each new feature introduced will first be prototyped in order to quickly introduce new features that will be continuously improved through each iteration. Using prototypes reduces the time it takes to implement all of the required pieces of functionality and allows for an adaption to change as the development process progresses. Prototyping allows the simulation of important aspects, unveiling the previously not-fully appreciated design issues [37].

The spiral model requires user feedback and as a result user testing will also be integrated throughout the iterative stages. This will be performed through random participant surveys. The survey question will be created by the model’s predications and the user’s ability will be compared to that of the models. This will ensure that accuracy is not lost as new features and the existing features are integrated and updated.

Unit-tests will be performed in order to ensure that individual units of the software are tested. Unit testing is the first level of software integration and ensures that as new code is introduced each prior unit does not break in terms of functionality. Unit tests also act as documentation that explains what the code is performing at various stages.

Integration testing will also be performed where multiple units are combined in order to expose faults within the interaction of units. Big bang or top down testing will be used in order to combine all of the units and test them appropriately.

System testing will be implemented within Grey-Box testing which is discussed later within this section. This stage tests the overall functionality of the entire system where all units are interacting with each other.

The last stage within integration testing is acceptance testing. This stage is also implemented the Grey-Box testing and ensures that the required functionality is fully implemented and working as expected.

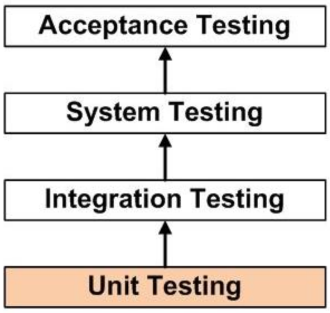


Figure 31 Stages of Integration Testing [41]

Grey-Box testing will be integrated in order to ensure that both the internal logic and paths of execution and requirements and functionality are successfully tested. Grey-box testing combines both White-Box and Black-Box testing. It is a more advanced form of testing that aggregates both methods in order to provide a different angle of testing to ensure that different paths of execution work as expected (from White-Box testing) and find errors in performance, incorrect functions, initializations and more (from Black-Box testing). The aim of Grey-Box testing is to verify system implementations against its specification. [38]

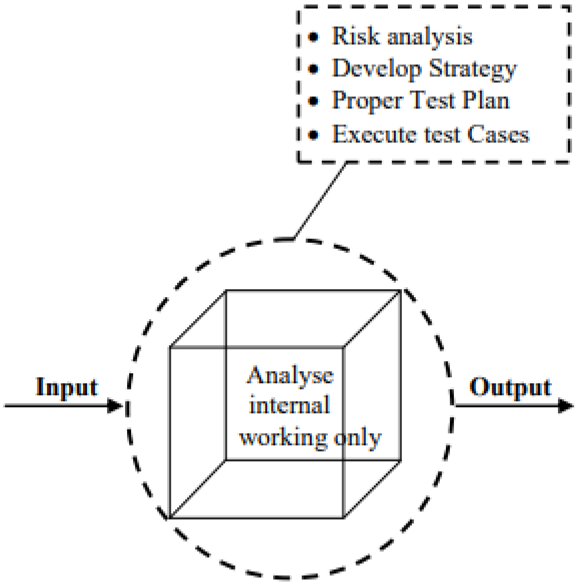


Figure 32 White-Box Testing [39]

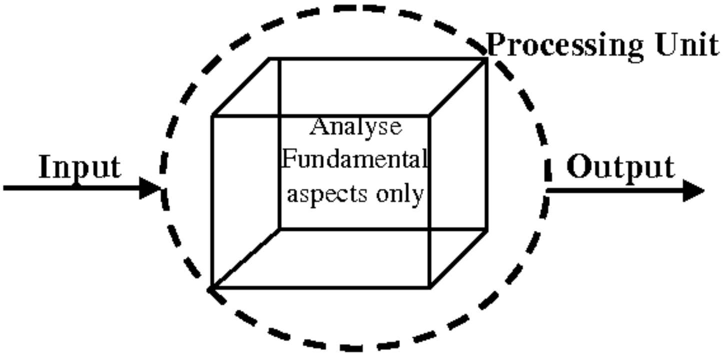


Figure 33 Black-Box Testing [39]

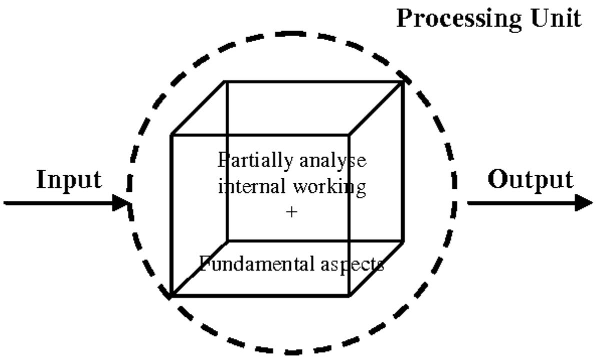


Figure 34 Grey-Box Testing [39]

As can be seen from the above diagrams Grey-Box testing encapsulated both the internal working analysis from White-Box analysis and functional aspects of Black-Box testing.

GitHub will be used as version control to ensure that any changes that reduce accuracy can be easily rolled back. GitHub also acts as a failsafe in the scenario where sections of the project may be deleted.

5.3. Plan for Evaluation

The evaluation of the system is equally as important as the testing phase. Multiple methods of evaluating the system have been investigated and the solutions have been explicitly varied in their approach in order to ensure that the data created is meaningful and has a strong degree of accuracy.

The first method of evaluation requires random participates to fill out a questionnaire where they try to map the given words to a topic that the LDA algorithm created and see if they arrived at the same conclusion as the model. The same would be done for the sentiment scores where human sentiment detection would be compared to the models.

An example of such a survey would present multiple diagrams to the participant similar to the one below and ask them to guess the topic using the words presented. The participant can then infer that the topic is hospitals using words such as (patient, cancer, process, late) and so on. If a high accuracy between human topic matching and automatic topic matching is created then it stands to reason that the topic modelling approach is accurate. This approach would also be copied and altered to suit a survey for users guessing the sentiment of sentences and comparing their results to the ones automatically generated.

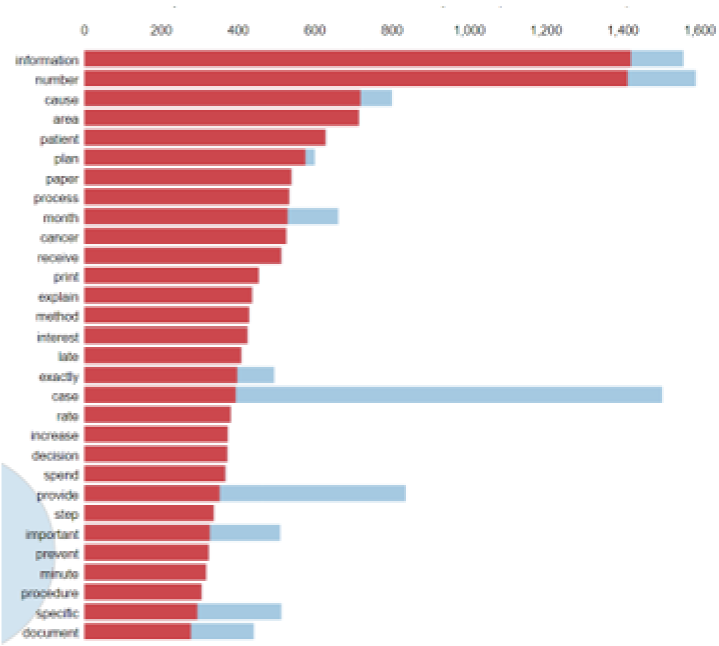
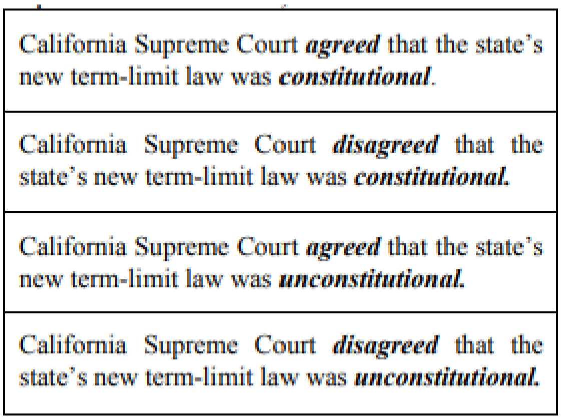


Figure 35 Random Participant Survey through guessing topics



California Supreme Court ***agreed*** that the state's new term-limit law was ***constitutional***.
California Supreme Court ***disagreed*** that the state's new term-limit law was ***constitutional.***
California Supreme Court ***agreed*** that the state's new term-limit law was ***unconstitutional.***
California Supreme Court ***disagreed*** that the state's new term-limit law was ***unconstitutional.***

Figure 36 Random Participant Survey Through Guessing Sentiment [40]

A second method of evaluating the LDA and sentiment topic models would be to use probabilistic algorithms that are used to define the accuracy of machine learning models. For LDA two such sores are Perplexity and Coherence scores. Perplexity score is a statistical measure of how well the model predicts a topics relation to the words through comparing word distribution and topic mixtures. The coherence score is used for assessing the quality of the learned topics through checking the number of words that appear in multiple topics. The score is higher the less the words are couple to multiple topics.

Another method of evaluation that applies to both LDA and sentiment topic modelling testing is to check the accuracy against a document that has manually computed either sentiment scores or topics. This requires new data for evaluation as training data that the model was created upon cannot be used. By comparing the data that was manually completed an accurate score can be generated around how closely the model resembles a human's capability of deciphering either sentiment or topic creation.

5.4. Conclusions

This chapter examined both the methods of testing and evaluations of the presented system. The testing section discussed the testing methodologies that will be integrated, unit testing and Grey-Box testing. The evaluation section delved into how to critically test the LDA model's capability to produce topics and the sentiment model's ability to understand human sentiment through three alternative methods. The first evaluation is completed through comparing the model's capability to that of random individuals through random participant surveys. The second computes scores to critically determine that the algorithms model is accurate using probabilistic algorithms. The final method of evaluation uses new data that already has already had sentiment scores and topics generated by an individual and this data is compared to their results.

6. Issues and Future Work

6.1. Introduction

This chapter introduces the issues and risks within the development of this project and the future work that is required to be completed. Currently what has been achieved so far within the prototype has been a data cleaning algorithm using natural language processing techniques, a basic LDA topic model and a detailed method for visualising the topic model. The literature review has aided in the creation of all of the requirements for this system and better understanding the suite of technologies and approaches available. The design has also encompassed the requirements gathering and the approach to be taken in order to complete this system using a mix of Kanban, simplified spiral modelling and CRISP-DM.

6.2. Issues and Risks

The challenges that still need to be resolve within this project are as follows:

- A method of linking the imaginary numbers of topics to real-life topics.
- Linking to a real newspaper API that has access to thousands of newspaper articles.
- A method of parsing the json data provided by the API above.
- Methods to increase the accuracy of the LDA model.
- An increased level of knowledge in order to implement Grey-Box testing.
- A sentiment analysis algorithm to determine whether or not a sentence is positive, neutral or negative.
- A method of mapping each sentiment analysis result to where the topic is flagged.
- A user interface that can display topics as well as their sentiment score and link them to specific newspaper companies.
- Adding random participants to evaluation and testing phases.

The proposal to approach these challenges are as follows (respectively):

- Using automatic probabilistic methods in order to map word-frequency to topics.
- Acquiring an API key from a news source API source such as: https://newsapi.org/s/ireland-news-api and writing code to retrieve articles daily. (Limit of 500 a day using above source).
- JSON parser and regex expressions for finding interesting metadata.
- Using different LDA model libraries, increasing data size and distribution of topics, more fine-tuned data processing and more methods to specifically increase model accuracy.
- Research Grey-Boxing projects and Udemy courses on this topic.
- Research how sentiment analysis works implement a suitable model.
- Both models will be computed separately and some pre-processing will be required in order to link them, possibly using the hash maps data structure.
- A user interface can be displayed with a suite of topic modelling libraries such as “matplotlib” and “pyLDAvis” exist for visualisation. The key interface will be accessed through command line which will create various html files for visualising different graphs and components. A textual result in a CSV will also be computed of all newspaper included and their bias score as well as the overall bias score by media.

- Random participants will be included in the development and evaluation process through requesting individuals to perform in a study where they will answer questions within surveys that will dictate how accurate the sentiment and topic models are in assessing the results.

The risks that may occur include are as follows:

- A limit of 500 requests per day using the main news API for Irish newspapers.
- Failing to complete the proposed project and all of the core functionality with high accuracy models implemented.
- Topic Model being too inaccurate to be useful.

The planned approach to these various risks is as follows:

- Request 500 articles daily. Within 10 days this can amass to 5000 articles which is enough to build an LDA Topic model from the real data.
- Using the Kanban method work will be time boxed each week. Progress will be followed by counting the number of Pomodoro points that were completed each week and a consist pace will be maintained in order to keep track of all work. The simplified spiral model will also provide a method to analyse risks before each iteration and aid in their prevention.
- An iterative approach will be implemented in order to gradually improve the accuracy of the LDA model and the progress will be monitored seeing the updated coherence, perplexity score and accuracy in terms of user evaluation surveys.

6.3. Plans and Future Work

The future plans for the project can be seen within the GANTT chart below. Another GANTT chart will be created when the project is completed and these two charts will be compared at the end in order to provide the difference between the planned and implemented approach.

The plan focuses on finishing the creation of all prototypes, implementing various features, testing throughout and writing the dissertation documentation.

6.3.1. GANTT Chart

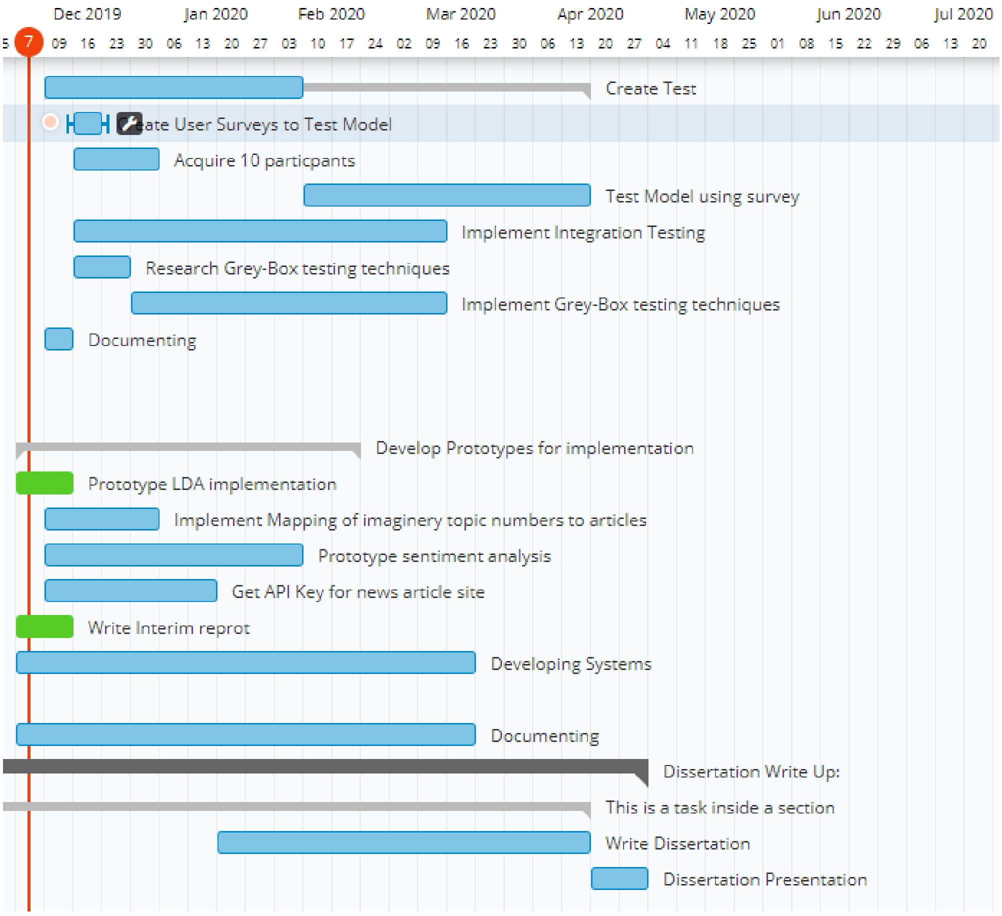


Figure 37 GANTT Chart

Bibliography

1. 2012/2013 JNRS - Readership [Internet]. News Brands Ireland. 2013 [cited 2019 Dec 1]. Available from: https://newsbrandsireland.ie/jnrs-20122013/
2. Mäntylä MV, Graziotin D, Kuutila M. The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. Computer Science Review. 2018 Feb;27:16–32.
3. Blei D. Probabilistic topic models. Communications of the ACM [Internet]. 2019 [cited 4 December 2019];(55):77–84. Available from: https://dl.acm.org/citation.cfm?id=2133826
4. Chen J, Yamada Y, Ryoke M, Tang X. Knowledge and systems sciences. 1st ed. Tokyo: Springer Singapore; 2018.
5. Liu Q, Chen Q, Shen J, Wu H, Sun Y, Ming W-K. Data Analysis and Visualization of Newspaper Articles on Thirdhand Smoke: A Topic Modeling Approach. JMIR Med Inform [Internet]. 2019 Jan 29 [cited 2019 Dec 6];7(1). Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6371067/
6. Blei D. Probabilistic topic models. Communications of the ACM (55):77–84. Available from: https://dl.acm.org/citation.cfm?id=2133826
7. T.P. A Sentiment Analysis Approach to Predicting Stock Returns [Internet]. Medium. 2018 [cited 2019 Dec 4]. Available from: https://medium.com/@tomyuz/a-sentiment-analysis-approach-to-predicting-stock-returns-d5ca8b75a42
8. Ganegedara T. Intuitive Guide to Latent Dirichlet Allocation [Internet]. Medium. 2019 [cited 2019 Dec 8]. Available from: https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-latent-dirichlet-allocation-437c81220158
9. Margaret E. Roberts, Brandon M. Stewart, and Dustin Tingley. Navigating the local modes of big data: The case of topic models. In R. Michael Alvarez, editor, Data Science for Politics, Policy and Government. In Press
10. David Hall, Daniel Jurafsky, and Christopher D Manning. Studying the history of ideas using topic models. In EMNLP, pages 363–371, 2008.
11. Godbole N, Srinivasaiah M, Skiena S. Large-Scale Sentiment Analysis for News and Blogs. In 2007.
12. Recasens M, Danescu-Niculescu-Mizil C, Jurafsky D. Linguistic Models for Analyzing and Detecting Biased Language. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) [Internet]. Sofia, Bulgaria: Association for Computational Linguistics; 2013 [cited 2019 Dec 6]. p. 1650–1659. Available from: https://www.aclweb.org/anthology/P13-1162
13. Hamborg F, Donnay K, Gipp B. Automated identification of media bias in news articles: an interdisciplinary literature review. Int J Digit Libr. 2019 Dec 1;20(4):391–415.
14. Gruenewald J, Pizarro J, Chermak SM. Race, gender, and the newsworthiness of homicide incidents. Journal of Criminal Justice. 2009 May 1;37(3):262–72.
15. Oliver PE, Maney GM. Political Processes and Local Newspaper Coverage of Protest Events: From Selection Bias to Triadic Interactions. American Journal of Sociology. 2000 Sep 1;106(2):463–505.
16. Patankar A, Bose J, Khanna H. A Bias Aware News Recommendation System. In: 2019 IEEE 13th International Conference on Semantic Computing (ICSC) [Internet]. Newport Beach, CA, USA: IEEE; 2019 [cited 2019 Dec 7]. p. 232–8. Available from: https://ieeexplore.ieee.org/document/8665610/
17. Elyashar A, Bendahan J, Puzis R. Is the News Deceptive? Fake News Detection Using Topic Authenticity. In 2017.
18. Loper E, Bird S. NLTK: The Natural Language Toolkit. CoRR. 2002 Jul 7;cs.CL/0205028.

19. Dutt R, Hiware K, Ghosh A, Bhaskaran R. SAVITR: A System for Real-time Location Extraction from Microblogs during Emergencies. arXiv:180107757 [cs] [Internet]. 2018 Nov 19 [cited 2019 Dec 5]; Available from: http://arxiv.org/abs/1801.07757
20. Natural Language Processing for Beginners: Using TextBlob [Internet]. Analytics Vidhya. 2018 [cited 2019 Dec 5]. Available from: https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/
21. scikit-learn: machine learning in Python — scikit-learn 0.22 documentation [Internet]. [cited 2019 Dec 5]. Available from: https://scikit-learn.org/stable/
22. Labs DD. Modern Methods for Sentiment Analysis [Internet]. Medium. 2017 [cited 2019 Dec 5]. Available from: https://medium.com/district-data-labs/modern-methods-for-sentiment-analysis-694eaf725244
23. WordNet | A Lexical Database for English [Internet]. [cited 2019 Dec 5]. Available from: https://wordnet.princeton.edu/
24. Documentation [Internet]. News API. [cited 2019 Dec 5]. Available from: https://newsapi.org
25. Punch KF. Introduction to Social Research: Quantitative and Qualitative Approaches. SAGE; 2005. 342 p.4
26. aneeshabakharia. Algorithms for the thematic analysis of twitter datasets [Internet]. Education presented at; 06:19:41 UTC [cited 2019 Dec 5]. Available from: https://www.slideshare.net/aneeshabakharia/algorithms-for-the-thematic-analysis-of-twitter-datasets2
27. Antezana L, Lagos C, Cabalin C. La opacidad de la política en la prensa chilena: un análisis de suplementos semanales. Estudios sobre el Mensaje Periodístico. 2017 Nov 20;23(2):727–46.
28. Mortensen D. How to Do a Thematic Analysis of User Interviews [Internet]. The Interaction Design Foundation. [cited 2019 Dec 5]. Available from: https://www.interaction-design.org/literature/article/how-to-do-a-thematic-analysis-of-user-interviews
29. Experience WL in R-BU. How to Analyze Qualitative Data from UX Research: Thematic Analysis [Internet]. Nielsen Norman Group. [cited 2019 Dec 5]. Available from: https://www.nngroup.com/articles/thematic-analysis/
30. Andrew Powell-Morse. Extreme Programming: What Is It And How Do You Use It? [Internet]. Airbrake Blog. 2017 [cited 2019 Dec 2]. Available from: https://airbrake.io/blog/sdlc/extreme-programming
31. What is a Kanban Board? Why and When to Use Kanban? [Internet]. Productivity Land. 2019 [cited 2019 Dec 5]. Available from: https://productivityland.com/what-is-kanban-board/.
32. Boehm B. Spiral Development: Experience, Principles, and Refinements [Internet]. 1st ed. 2000 [cited 2 December 2019]. Available from: https://resources.sei.cmu.edu/asset_files/SpecialReport/2000_003_001_13655.pdf
33. Chuang J, Wilkerson J, Weiss R, Tingley D, Stewart B, Roberts M, et al. Computer-Assisted Content Analysis: Topic Models for Exploring Multiple Subjective Interpretations. 2014.
34. Liu Q, Chen Q, Shen J, Wu H, Sun Y, Ming W-K. Data Analysis and Visualization of Newspaper Articles on Thirdhand Smoke: A Topic Modeling Approach. JMIR Med Inform [Internet]. 2019 Jan 29 [cited 2019 Dec 1];7(1). Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6371067/
35. admin. What is spiral model?7 Advantages and Disadvantages of model [Internet]. Am7s. 2019 [cited 2019 Dec 2]. Available from: https://www.am7s.com/spiral-model/

36. Ambler S. Feature Driven Development (FDD) and Agile Modeling [Internet]. Agilemodeling.com. 2019 [cited 2 December 2019]. Available from: http://agilemodeling.com/essays/fdd.htm
37. Buchenau M, Suri JF. Experience prototyping. InProceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques 2000 Aug 1 (pp. 424-433). ACM.
38. De Nicola G, di Tommaso P, Rosaria E, Francesco F, Pietro M, Antonio O. A Grey-Box Approach to the Functional Testing of Complex Automatic Train Protection Systems. In: Dal Cin M, Kaâniche M, Pataricza A, editors. Dependable Computing - EDCC 5. Berlin, Heidelberg: Springer; 2005. p. 305–17. (Lecture Notes in Computer Science).
39. Khan ME, Khan F. A Comparative Study of White Box, Black Box and Grey Box Testing Techniques. In 2012.
40. Kim S-M, Hovy E. Determining the sentiment of opinions. In: Proceedings of the 20th international conference on Computational Linguistics - COLING '04 [Internet]. Geneva, Switzerland: Association for Computational Linguistics; 2004 [cited 2019 Dec 7]. p. 1367-es. Available from: http://portal.acm.org/citation.cfm?doid=1220355.1220555
41. Unit Testing [Internet]. Software Testing Fundamentals. 2011 [cited 2019 Dec 7]. Available from: http://softwaretestingfundamentals.com/unit-testing/
42. Janson M, Smith L. Prototyping for Systems Development: A Critical Appraisal. MIS Quarterly. 1985;9(4):305.
43. Wirth R. CRISP-DM: Towards a standard process model for data mining. In: Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining. 2000. p. 29–39.
44. What is the CRISP-DM methodology? [Internet]. Smart Vision - Europe. [cited 2019 Dec 9]. Available from: https://www.sv-europe.com/crisp-dm-methodology/