University of London

CM3070 Final Project
Final Report

Literature Summarizer and Clustering using NLP and Deep Learning

Link: https://github.com/Muddykarp/Literature-Summarizer-and-Clustering-with-NLP-and-Deep-Learning

Date: 25/03/2024

Written By: Lee Kuan Wei

Chapter 1 - Introduction

Abstract

Machine learning (ML), deep learning (DL), and natural language processing (NLP) are branches of artificial intelligence (AI) technology and are becoming increasingly popular. ML and DL processes require huge amounts of data to train an application to help automate tasks and/or conduct analysis without significant human intervention. NLP deals with the processing of natural language data such as text and speech corpora using ML and DL techniques. This process allows computers to comprehend, generate and manipulate human language, which automates the extraction of important information and details in documents.

Introduction

This project is titled, "Literature Summarizer and Clustering using NLP and Deep Learning", with the aim of creating a software application that can process textual data in literature to provide a summary and group related literature together. This application requires the use of NLP to process the textual data in literature and ML techniques to help group the literature together. This project would focus on the use of ML techniques to cluster the data and follows template 1 of the CM3015 "Machine Learning and Neural Networks" module, "conducting deep learning on a public dataset". The dataset used for this project is obtained from Kaggle, "COVID-19 Open Research Dataset Challenge (CORD-19)", which is an open dataset that has over 1,000,000 scholarly articles about COVID-19, and related coronaviruses. This huge dataset of literature provides a good opportunity to develop the application.

Background and Motivation

When researchers embark on a new project, preliminary research and literature review are key components that must be conducted. However, searching for existing research documents that are related to their project can be extremely time-consuming and tedious. Each piece of literature contains large amounts of textual data, where academic papers can have a total of 4000 to 7000 words each. This slows down the research process as often the literature used as research is not directly relevant to the research they are conducting. The usual literature review process can be broken down into 2 stages: Stage 1 (Searching) and Stage 2 (Filtering).

Stage 1 is the process of searching for literature that provides relevant information to the project. This process usually starts by searching libraries using keywords relevant to the research title, and the results returned are usually literature that have those keywords in their title. Using keywords as the primary searching tool limits the scope of the search, providing literature that only explicitly mentions the searched keywords. It is very likely that relevant literature exists that does not use the same keywords and will be omitted in the search process. Additionally, new related techniques and/or research may not be found this way as the researchers do not have pre-existing knowledge of the keywords required for the new research.

Stage 2 is the process of filtering through all the information and data presented in the literature, and figuring out what is relevant to their research. Reading each individual literature is very time-consuming, and they may not provide the relevant information required by the researcher. If the researcher does not gather sufficient information, the entire process must be repeated from stage 1 which further increases the time spent on preliminary research.

This process can be further exacerbated if the research topic is niche or is simply difficult to search for relevant published research. The number of literature present in the library, the ease of access to literature, and even the number of researchers conducting the research can affect the amount of time spent on this process.

With the problems stated above, the motivation for this project is to develop a software application that can help quicken the process for preliminary research, especially during the literature review process. Using AI technology to automate both the searching and filtering processes of literature review can help save time on conducting literature review. By having the application conduct a thorough review process on the library of literature beforehand and clustering related literature together, the application would not be limited in scope by keyword searching mentioned above. Researchers can use the application to search for papers using relevant keywords to find the group of related literature.

Goals

There are 4 goals of this project:

1. Aid researchers in sieving through all the textual data in a piece of literature.
2. Provide a summary of the literature presented.
3. Group related literature together, helping researchers narrow their scope.
4. Create a data visualization tool to view the related literature.

Approach

To achieve the goals of this project, the software application would have to undergo a 4-step process. Step 1 is the reading, storage, and exploration of the data from the dataset, while ensuring the data is clean and appropriate for use. Step 2 utilizes NLP techniques to parse the textual data from the literature and convert each literature document into a feature vector. This feature vector can then be used in step 3, utilizing ML techniques to group related documents together. Step 4 is for this grouping to be represented in a data visualization tool, where it can meet the goals of this project.

Chapter 2 - Literature Review

Overview

This project can be divided into 3 main components, the current literature landscape, NLP techniques related to the goals of this project, and ML techniques related to clustering textual data. Each component would determine the effectiveness of the application in achieving its goals, hence sufficient research and literature review must be conducted.

The current literature landscape is an important area as it determines the usefulness of the software application, where the more literature being published results in the application being increasingly cost-efficient. For this project, the current literature landscape would be specific to academic papers and publishing.

The second area is to review NLP techniques relevant to the requirements of this project. Literature, like academic papers, contains a lot of textual data, with academic papers having between 4000 to 7000 words each. The NLP techniques used must be able to understand the context of the whole paper to be able to appropriately summarize and be able to be used as input for the ML portion of the algorithm.

Finally, reviewing ML techniques appropriate for the clustering of textual data summarized by the NLP is key to the project's goals. There are many clustering techniques, but researching for the best clustering technique helps improve cost-effectiveness of this project.

Landscape of Academic Papers

To review the current landscape of literature publishing, the paper titled: "Over-optimization of academic publishing metrics: Observing Goodhart's Law in action" is an excellent paper that provides context. The paper mentions that there has been an exponential growth of the number of publications year on year. The measure of academic success has not changed, and this measure of the number of publications and number of citations has led to the drastic increase in publications of academic papers. While the paper continues to research on the effectiveness of continuing the use of current academic success measures, it is not relevant to the understanding of the landscape of academic papers and would not be useful for my project. [1]

It is clear from the paper and its many data visualizations that there is an increasing trend of the number of academic papers being published year on year. This is apparent that from the 2000s until the present, publishing has increased many times. With more papers present in the global library, it makes sieving through all the data more time-consuming and tedious. This result could also relate to this project becoming increasingly useful in helping researchers reduce the amount of time spent on searching for other papers during literature review.

The next paper, "How Many Is Too Many? On the Relationship between Research Productivity and Impact" can also help provide context. It is known from the previous paper that there have been bigger incentives to publish as many papers as possible within the past few decades, and this has led many to debate about the quality vs quantity of academic papers. This paper seeks to find out if placing a larger emphasis on publishing more papers (i.e. preferring quantity), would result in lowered productivity and impact of academic papers. To summarize the main points that the paper presents, it is shown that the more papers a researcher publishes, the more citations would be made. This relationship

is seen to be more strongly correlated with older researchers, with decreasing returns with younger researchers. The paper also states that prior research has shown that this focus on quantity as research evaluations has had adverse effects, and this loss of quality in pursuit of quantity can provide less meaningful literature for research. [2]

Regardless, this provides a backdrop for this project that we can still expect researchers regardless of experience to continue to put emphasis on publishing more papers. While the effectiveness of getting more citations by publishing more papers is weaker in younger researchers, there is still a positive correlation between the 2 variables. More papers being published would lead to a larger volume of textual data being available. This increase in output of having more literature is helpful to both tune and optimize the algorithm, but also likely to increase the usefulness for this project to researchers.

Natural Language Processing Techniques

To better understand NLP techniques and what could be considered effective and efficient to use for this project, the papers: "An Exploratory Study of Helping Undergraduate Students Solve Literature Review Problems Using Litstudy and NLP" and "Natural language processing (NLP) in management research: A literature review" are useful in exploring NLP methods for textual data.

The first paper provides great insights into the many challenges student researchers face when undergoing the process of literature review, many of which relate to the lack of time and experience. These challenges limit the effectiveness of the produced literature review and hopes to produce a recommendation system that utilizes NLP and other tools. While the scope of the paper is similar to this project, the actual problem the paper addresses is different, with the aim of providing aid to students when they are conducting literature review. This aid is provided through three levels of analysis: demographic statistical analysis of literature, articles' coupling network and citation network, and using NLP and text-mining for topic modeling, word clouds, etc. [3]

This project can draw many parallels from the information provided in the paper, and further proves that the time-constraint that researchers face during literature review is a major issue. This project also aims to be able to group papers together with visualization tools, which is mentioned in the paper via the use of the coupling network and citation network. The networks draw links between different literature and could be a strong indicator of how well these articles relate to one another and could be a technique this project can utilize. Finally, the use of NLP mentioned in the paper is mostly used for topic modeling and as cues (using word frequency) for brainstorming opportunities for student researchers. While this may not be directly applicable to this project, the technique of topic modeling could be helpful during word processing in stage 1 of the algorithm and can also help with the classification problem.

The second paper is a literature review of the applications of NLP in different industries and the relevant technologies, and it provides a comprehensive review of NLP technologies. It is clear from the paper that NLP is useful even in many different use cases. As of today, most textual data is generated by humans through social media, reviews, etc., and they mostly have underlying sentiments behind the data. Therefore, even "technical" industries that deal with factual data and numbers (e.g. accounting & finance) also utilize sentiment analysis NLP technologies to generate insights. While there are 6 major NLP methods–text preprocessing, text representation, classification, topic modeling, sentiment analysis,

and deep learning–most applications of NLP do not require the use of all 6 methods. The frequently used techniques across all industries are classification, topic modeling and sentiment analysis. [4]

This project primarily deals with academic papers that tend to be factual without significant sentiments embedded in the research. The database of academic papers is huge and would result in literature that conducts research on all types of fields, and each paper has thousands of textual data which needs to be processed to gain insights. Therefore, it is unlikely that sentiment analysis would be required for this project, but all other 5 methods should be used to generate insights from every literature document.

Machine Learning Techniques

For ML techniques, this project specifically requires clustering and would be most beneficial if the clustering algorithm suits textual data. The paper "FOCT: Fast Overlapping Clustering for Textual Data" provides meaningful information on the possible application of overlapping clustering algorithms for the classification process of this project. Real-world datasets, including textual data and those from literature, often belong to different groups and the use of overlapping clustering algorithm can help link literature to different groups (e.g. this project can be linked to literature review, NLP, deep learning, instead of just 1 of those groups). This paper's newly proposed algorithm, fast overlapping clustering for textual data (FOCT) is based on the overlapping version of the algorithm, self-organizing map (SOM), called overlapping self-organizing map (OSOM). SOM is a commonly used algorithm for clustering textual data as it helps convert high-dimensional data into a low-dimensional space and can be understood by humans. The FOCT algorithm is also tested against other algorithms such as k-means, overlapping k-means (OKM), SOM, and OSOM, all of which are viable algorithms for text clustering and could be implemented for this project. Finally, the results of the experiments between all 5 algorithms show that while FOCT does better against OKM and OSOM, k-means and SOM still performs better than FOCT but it does not have overlapping topics. [5]

This paper has a lot of technical details on the implementation and design of the new algorithm, FOCT, which aims to overcome the limitations of OSOM which is its high computational requirements and long processing times. As stated above where the results of execution time and topological measures are better in FOCT against the 2 other overlapping algorithms (OKM & OSOM), the overall results are still not great in comparison with current non-overlapping algorithms (k-means, SOM). While the benefits of overlapping can be greatly beneficial for this project, the limitations of the overlapping algorithms may impede the classification of the literature this project aims to conduct. Hence, non-overlapping clustering algorithms will be used to conduct classification in this project and the classification algorithms of k-means and SOM should be considered. Overlapping clustering algorithms could be used as further research in the future to improve and benefit from the advantages of overlapping clustering with regards to textual data and literature clustering.

The next paper, "Textual data mining for industrial knowledge management and text classification: A business oriented approach" helped provide useful information on textual data mining techniques to help classify textual data, and proposed a new algorithm to improve the classification accuracy. The new algorithm proposed in the paper known as multiple key term phrasal knowledge sequences (MKTPKS) utilizes 3 stages of processing to produce 2 classes of output–good information documents, and bad information documents. The general flow of data would go through a text mining

unit → 1st level knowledge processing unit → clustering → 2nd level knowledge refinement unit → 3rd level knowledge utilization & text classification unit. [6]

The text mining unit is used to pre-process the textual data to help reduce the dimensions of the data and make it clean such that it would not negatively affect the classification process. The 1st level knowledge processing unit is then used to process the textual data into a suitable representation that can retain the entire information space and reduce the loss of key information, and methods such as term frequency (TF), inverse document frequency (IDF), or term frequency-inverse document frequency (TF-IDF) can be used. The clustering process is the grouping of data into similar types via certain quantitative measures. Clustering techniques such as K-means, K-nearest neighbours (K-NN), and others can be used. The 2nd level knowledge refinement unit utilizes the Apriori algorithm, which is a technique used to uncover hidden relations between variables in large datasets. Finally, the 3rd level knowledge unit utilizes a domain expert to manually inspect the data into 2 classes and is primarily used to test the accuracies of the classifiers used.

This paper has a lot to unpack as it provides more technical details as compared to other literature reviewed papers. To begin, the paper mentions many crucial details about textual data that affects the usability of a general-purpose algorithm like the one this project is aiming for. Textual information from industrial or corporate environments often exists in the form of descriptive data formats which are concise and contain many industry specific terms, many of which are also present in academic papers and literature. While the final output of the MKTPKS algorithm is in the form of 2 classes (good or bad information) is not useful for this project, the process of breaking down the raw textual data can be used. It is likely that with the huge textual data that this project's algorithm will have to process, following the multi-stage process of MKTPKS can help improve the accuracy of the classification algorithm. Hence, the application of having the raw textual data undergo a text mining unit → 1st level knowledge processing unit → clustering → 2nd level knowledge refinement unit could be effective in generating an accurate classification of the data. Having accurate classification of the data can help my project achieve its goal of providing researchers with accurate, related literature.

Summary and Takeaways

The literature reviewed regarding the current landscape of academic papers helped to provide context and is evidence to the potential cost-effectiveness of this project. The papers showed substantial evidence that the total number of newly published literature will continue to increase over time. This growing amount of literature makes it more tedious for researchers to review as there would be more content to process. But helps the application grow its library of processed literature, creating an increasingly robust clustering of literature that researchers can use to find literature relevant to their research. This further improves the effectiveness of the application and is a motivator for the development of the application.

The literature review for NLP techniques helped provide a deeper understanding of the commonly used techniques for textual data processing. Common techniques relating to NLP such as text preprocessing, text representation (like word frequency), topic modelling, and classification are all required for this project. The use of citation network can help link different literature together, helping with the clustering problem this application aims to solve. A key takeaway is to utilize NLP packages and libraries to help with text processing, this is due to the high effectiveness of the models in these packages and that this project's focus is on ML clustering techniques and not NLP techniques.

The literature reviewed for ML techniques are technical but provide useful information in developing an effective clustering model specifically for textual data processing. Both papers provided details on 2 new algorithms that solve different problems for text clustering. The key issues presented are overlapping topics and classification accuracy. While both may seem unrelated, they are interconnected. Most literatures have multiple topics, but commonly used and robust clustering models can only classify by single topics, which negatively affects the accuracy of the classification algorithm. Further development to this project could be to develop a robust classification algorithm that can process textual data and have overlapping topics with good accuracy.

References

[1] Fire, Michael & Guestrin, Carlos. (2019). Over-optimization of academic publishing metrics: Observing Goodhart's Law in action. GigaScience. 8. 10.1093/gigascience/giz053.

[2] Larivière V, Costas R (2016) How Many Is Too Many? On the Relationship between Research Productivity and Impact. PLOS ONE 11(9): e0162709. https://doi.org/10.1371/journal.pone.0162709

[3] Wong, G.K.W.; Li, S.Y.K. An Exploratory Study of Helping Undergraduate Students Solve Literature Review Problems Using Litstudy and NLP. Educ. Sci. 2023, 13, 987. https://doi.org/10.3390/educsci13100987

[4] Yue Kang, Zhao Cai, Chee-Wee Tan, Qian Huang & Hefu Liu (2020) Natural language processing (NLP) in management research: A literature review, Journal of Management Analytics, 7:2, 139-172, DOI: 10.1080/23270012.2020.1756939

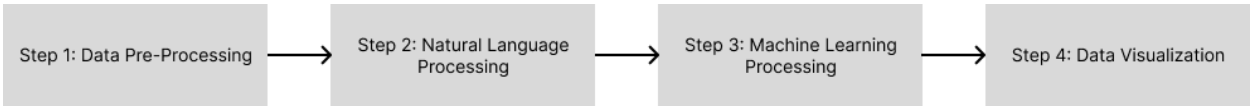
[5] N. Ur-Rahman, J.A. Harding, Textual data mining for industrial knowledge management and text classification: A business oriented approach, Expert Systems with Applications, Volume 39, Issue 5, 2012, Pages 4729-4739, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2011.09.124.

[6] A. Khazaei, H. Khaleghzadeh and M. Ghasemzadeh, "FOCT: Fast Overlapping Clustering for Textual Data," in IEEE Access, vol. 9, pp. 157670-157680, 2021, doi: 10.1109/ACCESS.2021.3130094.

Chapter 3 - Design

Overview

To design the software application for this project, I have broken down the individual components as well as the expected timeline for each component. To reiterate the goals of this project, the application needs to be able to process the textual data in a piece of literature, provide a summary of the literature, classify related literature together, and create a data visualization tool for the user. Additionally, as briefly described in chapter 1, the approach to the development of this application is to formulate a 4-step process. Step 1 is to preprocess the data, which includes reading and extracting the data from the dataset into data frames, and the cleaning of the data. Step 2 would be the use of NLP libraries and packages to help parse the textual data from every literature article and convert them into feature vectors. Step 3 would use those feature vectors to classify and cluster the literature documents together. Step 4 would be to represent the clusters in a data visualization tool for users to interact with. Deep learning is conducted in both steps 2 and 3.



Step 1: Data Pre-Processing

Before being able to utilize deep learning techniques, the raw data needs to be pre-processed and transformed into a form usable by the model. The input data is from a dataset of academic papers, each with its own metadata. This metadata contains a lot of crucial data such as the title, abstract, authors, etc., and will be used to identify each document. JSON file format is the primary file storage and for the application to access all the textual data in the dataset. The use of helper functions to help process the data and only extract key features is necessary as there is a huge quantity of data and minimizing data extracted is important. Additionally, the dataset contains many academic papers, and some are not written in English, which should be removed as they would not work optimally with the NLP techniques that follow English semantics.

Step 2: Natural Language Processing

With the data pre-processed and cleaned, it can now be used and processed using NLP techniques. As researched during the literature review, text preprocessing and text representation needs to be conducted to further clean the data and transform it. Each document has thousands of words, and they need to be trimmed appropriately to help improve accuracy of the application and to minimize excessive time spent rendering and/or processing.

To begin conducting text preprocessing, the removal of stop words will help eliminate a huge portion of the text. Stop words are common words used in English that help create a cohesive sentence but provide little useful information for the NLP process. Additionally, since the dataset consists of academic papers, stop words commonly used in these papers can also be removed to help further narrow the scope to process. Tokenization and text parsing must also be used to break the text and transform it into discrete data structures that can then be further processed.

Once the textual data has been preprocessed, they can transform into a format that can be processed by the NLP algorithms. The data will be vectorized using term frequency-inverse document frequency (TF-IDF) method which essentially measures how frequently a word appears and that

determines its importance in the document. TF-IDF is only 1 method in this textual representation process, other methods such as bag-of-words, and singular value decomposition (SVD) are all viable methods. Using only 1 method for text representation would be required, the other techniques may be employed in the project to cross-check and determine their respective effectiveness.

Step 3: Machine Learning Processing

Step 3 is where the classification and topic modeling will be conducted. To conduct classification, it could be conducted using supervised learning or unsupervised learning techniques. However, supervised learning is not applicable for this project as every literature document is unlabeled. For unsupervised learning techniques, there are multiple suitable classification methods as stated in the literature review, such as K-means clustering, self-organizing map (SOM), and Bayesian classification methods. While only 1 method is required, the other methods can be used to test and evaluate the effectiveness of each algorithm to determine which provides the best results.

K-means clustering will be used to conduct clustering on the vectorized textual data, and the number of clusters will be determined using the elbow method. Additionally, principal component analysis (PCA) must be conducted before K-means clustering to reduce the dimensionality of the data since each vectorized data would still be large. Using PCA can also help reduce noise while maintaining high variance.

When clustering is completed, topic modeling can be conducted to find the most significant feature/word of each cluster. Topic modeling is another unsupervised classification method, where it can be used to learn and predict key values or words from the clusters. The clusters of literature likely do not have a group name or theme, and topic modeling will help to identify the similar themes of every cluster. This process creates meaning to each cluster and aids in creating a useful user interface for the application which users can interact with. Latent Dirichlet Allocation (LDA) will be the method used, where it attempts to find the topic(s) for the cluster, because each document can be described by a distribution of topics and each topic can be described by a distribution of words.

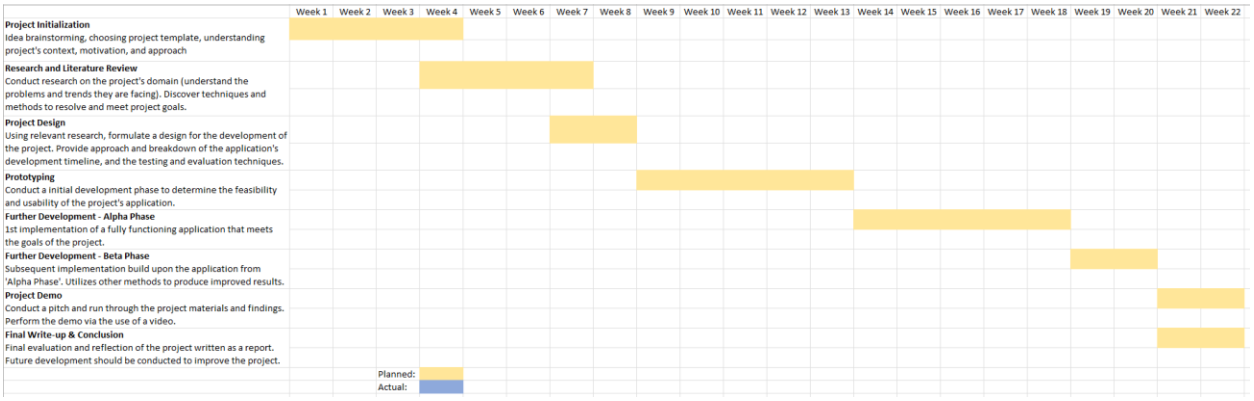
Step 4: Data Visualization

Once steps 2 & 3 are completed, the data visualization can be conducted by plotting the clusters together with each papers' metadata. Data visualization is an important component as it helps bring the NLP and deep learning models together to produce an improved searching experience for researchers. Generating a simple visualization of the clusters can help meet the goals of the project by providing a visual representation of the related literature. However, this would not be effective nor an improvement of the current search methods. To resolve this, visualization needs to be clear and interactive for users to navigate around. Each paper must belong to a cluster, should be able to show its connection with the other papers of the same cluster (e.g. colour coding), and should be able to show the metadata/details of itself.

Additional features such as a search tool for researchers to search specific keywords can be a great improvement by combining current keyword searching method with this model. Isolating specific clusters for further investigation can also aid the searching process. A 2-dimensional model may not be sufficient to showcase the complex data and connections, implementing a 3-dimensional model together with interactive elements (such as zooming, pull & drag functions) might help.

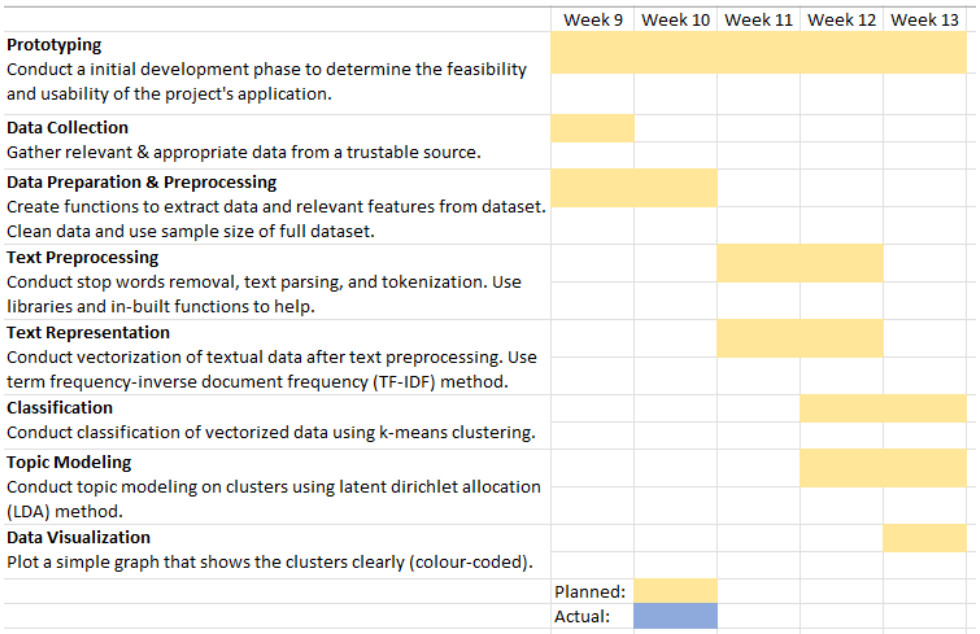
Implementation Workplan

The workplan for this project can be broken down into multiple development phases for the implementation of the application—prototyping, and further development (Alpha/Beta Phases). Prototyping is the development process using simplified methods to test the viability of the application. Further development is the iterative process of continuously improving the application with testing and evaluation, and can be further split into 2 phases, alpha and beta. Documentation and evaluation will be conducted throughout the development process but will be consolidated and finalized in the final weeks of the project timeline.



Prototyping

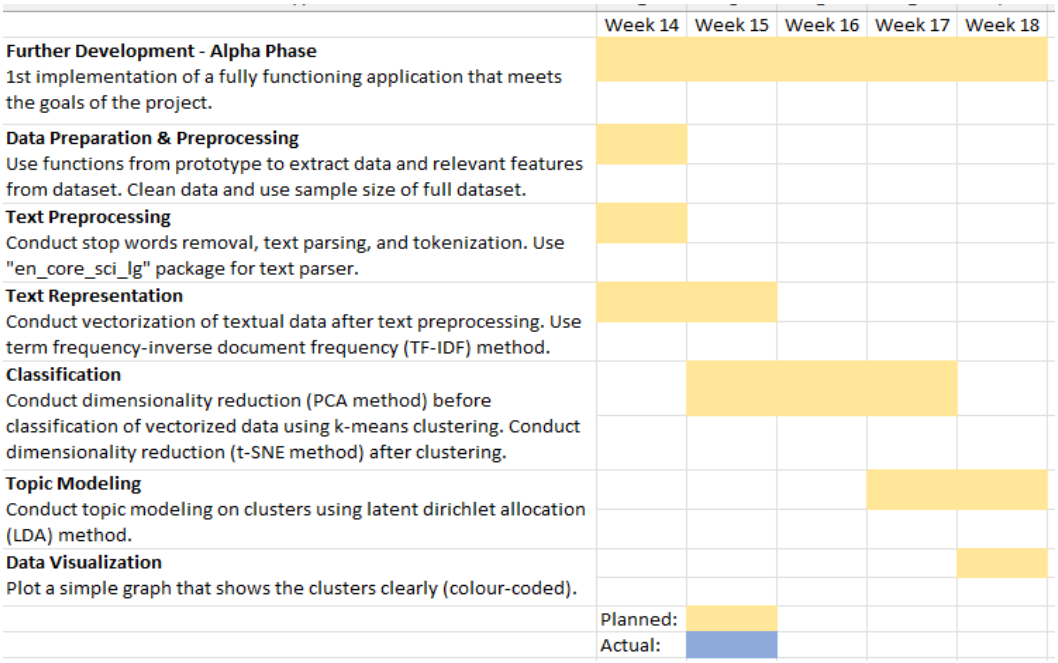
This prototype is used to test the viability of the application, primarily using in-built functions from libraries and packages. There are 2 goals to meet, develop a “bare-bones” model that satisfies all of the project’s goals, and use the prototype as a baseline.



The prototyping workplan timeline is split into 5 weeks, with each stage overlapping with each other as the methods used are easy to implement. Each of the 4 steps are given about 1.5 weeks to be implemented, except for the simple plotting of a graph that shows the clusters.

Further Development (Alpha Phase)

Once the prototype is completed and the results prove that the application is viable, alpha phase can begin to develop a fully functional application. More advanced and/or relevant techniques will be used during different stages of processing, instead of relying on in-built packages, to develop a better performing and more robust model. This model, also known as the 'Alpha Model', can be considered as the first implementation of a fully functional application.

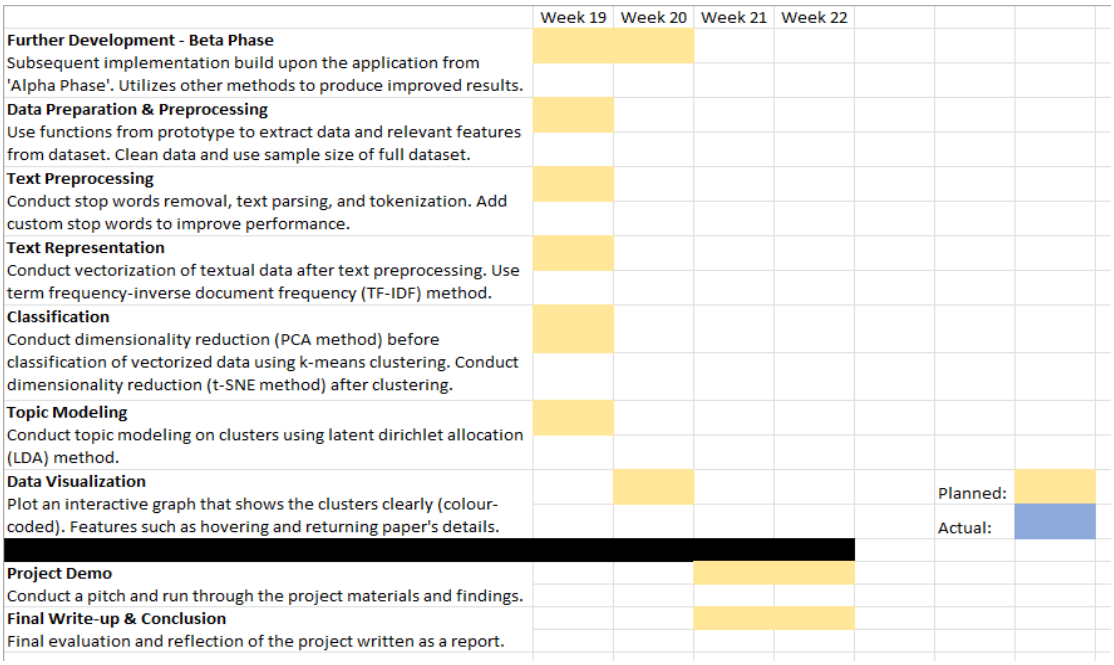


The focus for this development phase would be to improve the classification process and its performance. This is done by conducting PCA and t-SNE dimensionality reduction techniques. The text preprocessing process is also improved by using a different text parser.

Further Development (Beta Phase) and Final Write-up

Once the first implementation of a fully functional model is created, beta phase development on further improving the results can start. This phase is an iterative developmental process on improving the model, which can result in improved accuracy of the model, more efficient processing of data, clearer data visualization, etc. During this phase, other methods can be used and evaluated against the new baseline results from the 'Alpha Model'. This is an important phase to develop the application and bring it closer to commercially usable software.

Due to the short timeline for this project, I have placed less focus on this phase of the project, hence the short 2-week assigned to beta phase. The main focus is to improve the data visualization such that an interactive plot can be created, this way users can interact with it and have a better searching experience. The final 2 weeks of the project are assigned to the project demo video and the final write-up of the report. Since documentation is conducted throughout the development process, less time is required for the final write-up.



Testing and Evaluation

The testing of the application is conducted periodically whenever a new method or an entire development phase is completed. This is to ensure that the method used is functioning as expected, and the results are appropriate. Examples of testing would be the exploration of the dataset after cleaning is conducted or plotting of the data to check for deficiencies.

For the evaluation of the application's performance, the nature of the dataset makes it difficult to conduct a direct evaluation. A combination of methods needs to be used to estimate the rough performance of the application. Additionally, evaluation is most effective when conducted on the machine learning algorithms used in the project. This is because the performance can be measured, and the plotting of the clusters allows for some manual examination.

Chapter 4 - Implementation

Step 1: Data Pre-Processing

The goal of step 1 is to be able to access the dataset, read and extract relevant data features, load it into a data frame, and to clean it before it is ready to be used by the models. The dataset used for this project has a few characteristics that cause data preprocessing to be difficult. Firstly, the dataset stores all its textual data using JSON file format. Secondly, each document of the dataset could be structured differently, and each contains a lot of textual data. Finally, the dataset is huge with over 400,000 files in the directory. To overcome the 3 characteristics of the dataset, helper functions are created to deal with each of them. Certain key techniques used was the cleaning of the data by removing files that are not written in English. This was important as the NLP techniques used are designed for English and are not effective in processing other languages. Semantics and other language structures are not transferable and could negatively affect the performance of the NLP methods.

[16]:
```
# Explore the languages in the dataset
languages_dict = {}
for lang in set(languages):
    languages_dict[lang] = languages.count(lang)

print("Total number of documents: {}\n".format(len(languages)))
pprint(languages_dict)
```

```
Total number of documents: 62697

{'ar': 2,
 'ca': 7,
 'cy': 5,
 'da': 1,
 'de': 621,
 'en': 61280,
 'es': 437,
 'fr': 241,
 'hr': 1,
 'id': 2,
 'it': 15,
 'nl': 38,
 'pl': 2,
 'pt': 38,
 'so': 1,
 'sq': 1,
 'sv': 1,
 'sw': 1,
 'zh-cn': 3}
```

[17]:
```
# Remove data that are not in English (would not work optimally w English NLP methods)
df_covid['language'] = languages
df_covid = df_covid[df_covid['language'] == 'en']
```

Step 2: Natural Language Processing

For this step of the process, the main goal is to process the cleaned raw textual data using NLP techniques to transform it into a form that can be used by the classification algorithm. There are 2 main NLP techniques that would be employed here: text preprocessing and text representation.

Text Preprocessing

This is the process of removing textual data while retaining semantic and contextual information of the words in the literature. The goal is to reduce the amount of textual data, keeping the important data and converting them into discrete pieces of textual data. 3 NLP techniques were applied to help reduce the dimensionality of each file by removing irrelevant data such as stop words, punctuation, upper case characters, and by breaking groups of words into smaller chunks.

Stop Words Removal

Stop words are common words used in English that help construct a cohesive sentence structure but provide little to no useful information for the NLP model. Removing these words will help reduce a lot of textual data yet preserving a lot of useful information. Custom stop words are also added to the model, these words are commonly used in academic papers (not commonly used in conversations or text messaging) and do not provide value.

[19]:
```
# Use NLP package built-in English stop words
punctuations = string.punctuation
stopwords = list(STOP_WORDS)
```

[20]:
```
# Add custom stop words relevant to academic papers
custom_stopwords = [
    'doi', 'preprint', 'copyright', 'peer', 'reviewed', 'org', 'http', 'https', 'et', 'al', 'author', 'figure', 'fig', 'rights',
    'reserved', 'permission', 'used', 'using', 'biorxiv', 'medrxiv', 'license', 'Elsevier', 'PMC', 'CZI', 'citation', 'cite', 'reference']

for i in custom_stopwords:
    if i not in stopwords:
        stopwords.append(i)
```

Text Parsing & Tokenization

Text parsing is the process of breaking down textual data into individual components which the model can understand and use. Punctuations, upper & lower case letters are all removed or converted to lowercase during this process. The particular text parser used is "en_core_sci_lg" from the "scispaCy" library. This is due to the parser being specifically tuned for biomedical, clinical and scientific text processing, which are very relevant to the data in the dataset.

Tokenization is the process of converting group of texts into smaller portions, known as tokens. This process helps the model interpret and analyze the textual data in the body of text. The tokenizer from spaCy is used to help conduct this process.

[21]:
```
# Initiate text parser
parser = en_core_sci_lg.load(disable=["tagger", "ner"])
parser.max_length = 7000000

# Create tokenizer function
def spacy_tokenizer(sentence):
    mytokens = parser(sentence)
    mytokens = [word.lemma_.lower().strip() if word.lemma_ != "-PRON-" else word.lower_ for word in mytokens]
    mytokens = [word for word in mytokens if word not in stopwords and word not in punctuations]
    mytokens = " ".join([i for i in mytokens])
    return mytokens
```

[22]:
```
# Apply text parsing and tokenization on the processed text data
tqdm.pandas()
df_covid["processed_text"] = df_covid["body_text"].progress_apply(spacy_tokenizer)
```

```
100%|██████████████████████████████████████████████████████████| 61280/61280 [7:26:50<00:00,  2.29it/s]
```

Text Representation

This is the process of using the preprocessed textual data and transforming it into discrete data structures. The transformation process is known as text vectorization, and it converts the textual data into numerical representations which ML models can process and classify them. The vectorization method used for this project is Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF vectorizes the data by measuring the frequency of words that appear in a body of text, and determines its importance based on the frequency. Limiting the maximum number of unique words from each document is conducted to help the classification model.

Text Vectorization

As mentioned above, text vectorization is needed to transform the data. There are multiple methods that can be used to transform the data, but this project would utilize Term Frequency-Inverse Document Frequency (TF-IDF) method. This method works by measuring the frequency certain words appear and determine its importance within the document. Quantifying the maximum number of features (unique words) possible from each document helps with the classification process later on.

[23]:
```
# Create vectorization function using tf-idf
def vectorize(text, maxx_features):
    vectorizer = TfidfVectorizer(max_features=maxx_features)
    X = vectorizer.fit_transform(text)
    return X
```

[24]:
```
# Apply vectorization on the processed text data
text = df_covid['processed_text'].values
max_features = 2**12
X = vectorize(text, max_features)
```

Step 3: Machine Learning Processing

The goal of this processing step is to use unsupervised machine learning techniques to find clusters of literature. By using ML techniques, the prediction process can be automated without human intervention and is done by using vectorized textual data to perform classification to create a visual plot of the clustered data. This is followed by topic modeling, which assigns a feature for every cluster.

Classification

There are 3 primary ML techniques applied during this stage, Principal Component Analysis (PCA) for dimensionality reduction, K-means Clustering or Self Organizing Mapping (SOM) for classification, and t-distributed stochastic neighbour embedded (t-SNE) for dimensionality reduction.

PCA is conducted to reduce the high dimensions of a large and complex dataset into manageable chunks and is done by summarizing the information content in each document. A variance value of 95% is maintained to ensure that not too much information is lost during the process. This process will also help remove noise from the data, creating better clusters.

Dimensionality Reduction (PCA)

The huge number of data points, and the large number of variables in each data point makes it difficult to visualize the data directly. Hence, the dimensionality reduction technique of Principal Component Analysis (PCA) is used to help summarize the information content in each data point. Summarizing the data while maintaining a high variance (95%) of the data is crucial so as to not lose too much information while dimensionally reducing the data. Another benefit of this process is that it will filter out noise from each data point, further cleaning the data.

[25]:
```
# Conduct PCA on the data, while keeping variance of 95%
pca = PCA(n_components=0.95, random_state=42)
X_reduced= pca.fit_transform(X.toarray())
X_reduced.shape
```

[25]: (61280, 2774)

The classification of the data can be conducted by many methods, and 2 unsupervised clustering algorithms will be used. This is to test the performance of the algorithms and use the one that performs better. K-means clustering is the first method, and it works by categorizing each data point by taking its mean distance to 'K' number of randomly initialized centroids, and the closest data point would be clustered around that centroid. 'K' is the number of centroids created and determines the number of clusters. The positions of the centroids are also updated as the algorithm progresses.

[29]:
```
# Conduct k-means clustering with the optimal number of K
k = 40
kmeans = KMeans(n_clusters=k)
y_pred = kmeans.fit_predict(X_reduced)
df_covid['y'] = y_pred
```

T-SNE is conducted to help reduce the high dimensional features from K-means clustering, such that the data points can be plotted on a 2-dimensional graph. T-SNE is a technique that reduces dimensionality by keeping similar data points together and dissimilar data points apart in both low and high dimensions. However, this process also clusters the data points and would negatively affect the accuracy of the K-means clustering shown in the graph.

Dimensionality Reduction (t-SNE)

The high dimensional features after performing K-means clustering is difficult to plot. Another dimensionality reduction method is employed to reduce them into 2 dimensions, which can be used as x and y coordinates for a plot. t-distributed Stochastic Neighbour Embedding (t-SNE) is a technique that reduces dimensionality by keeping similar data points together and dissimilar points apart in both low and high dimensions. This is similar to PCA as they both reduce dimensionality while maximizing variance, but by using different methods.

It is important to note that the dimensionality reduction process performed by t-SNE is likely to also perform another clustering process. This can result in the same data point being clustered differently by the 2 clustering algorithms. This may be a drawback of using this method.

[30]:
```
# Perform t-SNE and transform the data
tsne = TSNE(verbose=1, perplexity=50)
X_embedded = tsne.fit_transform(X.toarray())
```

```
[t-SNE] Computing 151 nearest neighbors...
[t-SNE] Indexed 61280 samples in 0.329s...
[t-SNE] Computed neighbors for 61280 samples in 214.262s...
[t-SNE] Computed conditional probabilities for sample 1000 / 61280
[t-SNE] Computed conditional probabilities for sample 2000 / 61280
[t-SNE] Computed conditional probabilities for sample 3000 / 61280
[t-SNE] Computed conditional probabilities for sample 4000 / 61280
```

SOM is the next clustering algorithm, and it works by using an artificial neural network that employs the competitive learning method. The algorithm transforms the high dimensional data into smaller dimensions while maintaining the topological properties of the data. The algorithm undergoes 5 stages: initialization, learning, neighbourhood updating, convergence and clustering.

[32]:
```
# Conduct SOM clustering with the optimal number of clusters
a = 4
b = 2
# Initialization and training of a MiniSom object
som_shape = (a, b)
som = MiniSom(som_shape[0], som_shape[1], X_reduced.shape[1], sigma=.5, learning_rate=.5, neighborhood_function='gaussian')
som.train_batch(X_reduced, 1000, verbose=True)  # Train for 1000 iterations

# Each neuron represents a cluster
winner_coordinates = np.array([som.winner(x) for x in X_reduced]).T

# Using "np.ravel_multi_index" to convert the 2-dimensional coordinates to a 1-dimensional index
cluster_index = np.ravel_multi_index(winner_coordinates, som_shape)
```

```
 [ 1000 / 1000 ] 100% - 0:00:00 left
 quantization error: 0.940922696356802
```

Topic Modeling

To conduct topic modeling on the clusters, the Latent Dirichlet Allocation (LDA) method is used, where it attempts to find topics for each cluster by the distribution of topics of documents in each cluster, and each topic can be described by a distribution of words. This process helps with the searching process when using keywords.

Topic Modeling

This process helps find the most significant feature/word of each cluster. The clusters of literature are not labelled by topics, and topic modeling can help identify the most significant terms for each cluster. This provides additional context and meaning to each cluster to be identified by keyword searching. For this project, topic modeling is performed via Latent Dirichlet Allocation (LDA), which attempts to find the topic by the distribution of topics and each topic can be described by a distribution of words.

[35]:
```
# Create vectorizers=number of clusters, to topic label each cluster
vectorizers = []

for i in range(0, (a*b)):
    # Creating a vectorizer
    vectorizers.append(CountVectorizer(min_df=5, max_df=0.9, stop_words='english', lowercase=True, token_pattern='[a-zA-Z\-][a-zA-Z\-]{2,}'))
```

[36]:
```
# Vectorize the data from each cluster
vectorized_data = []

for current_cluster, cvec in enumerate(vectorizers):
    try:
        vectorized_data.append(cvec.fit_transform(df_covid.loc[df_covid['y'] == current_cluster, 'processed_text']))
    except Exception as e:
        print("Not enough instances in cluster: " + str(current_cluster))
        vectorized_data.append(None)
```

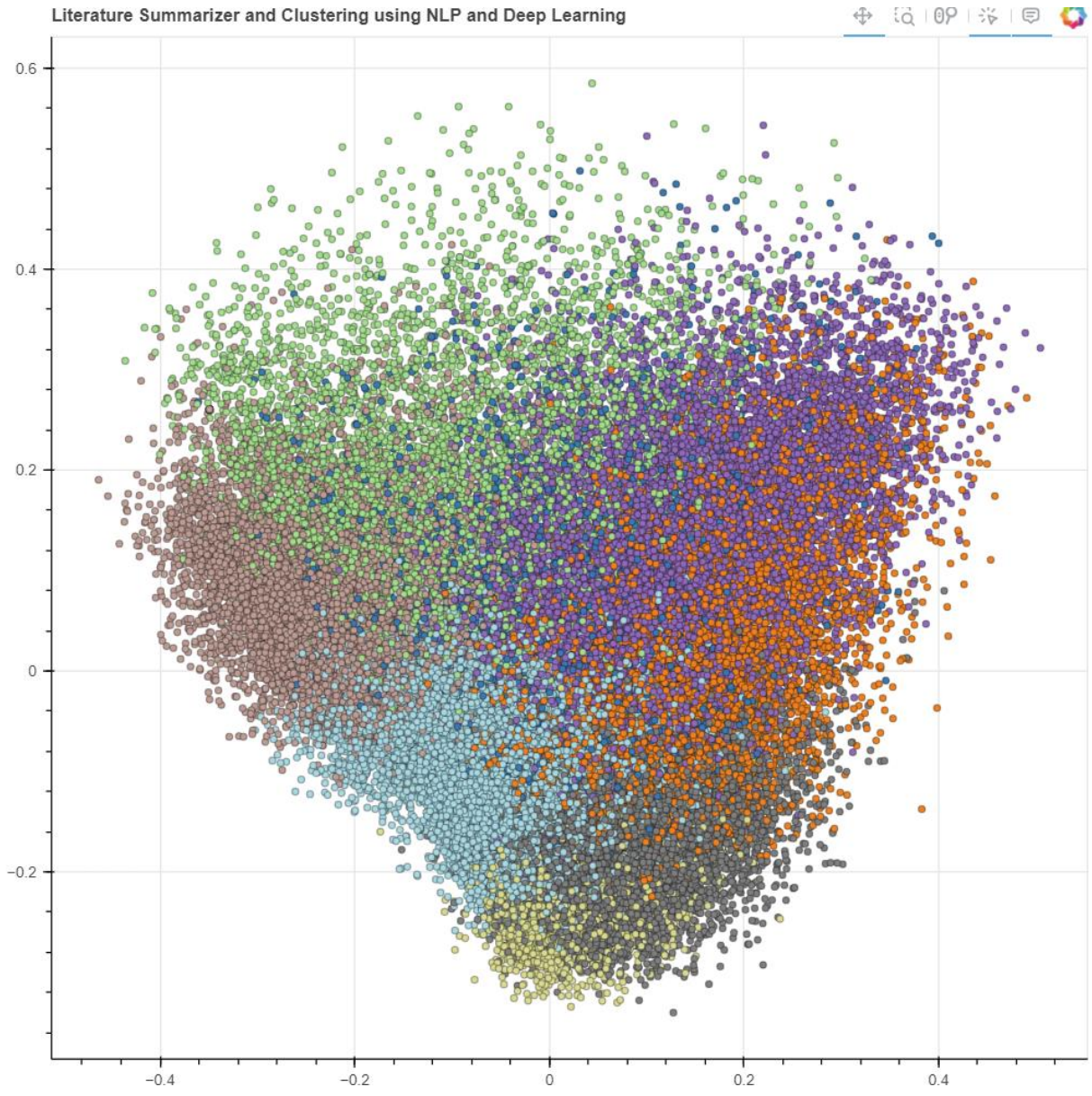
[37]:
```
# Perform topic modeling with LDA

# Number of topics per cluster
NUM_TOPICS_PER_CLUSTER = len(vectorized_data)

lda_models = []

for i in range(0, NUM_TOPICS_PER_CLUSTER):
    lda = LatentDirichletAllocation(n_components=NUM_TOPICS_PER_CLUSTER, max_iter=10, learning_method='online',verbose=False, random_state=42)
    lda_models.append(lda)
```

[38]:
```
# For each cluster, fit the generated LDA model
lda_clusters = []

for current_cluster, lda in enumerate(lda_models):
    print("Current Cluster: " + str(current_cluster))

    if vectorized_data[current_cluster] is not None:
        lda_clusters.append((lda.fit_transform(vectorized_data[current_cluster])))
```

```
Current Cluster: 0
Current Cluster: 1
Current Cluster: 2
Current Cluster: 3
Current Cluster: 4
```

Step 4: Data Visualization

To generate an interactive plot for users to have an interface and more effectively search for their research, the 'Bokeh' package is used instead of 'Seaborn'. Functions like hover can be implemented to help users interact with the plot and the clusters of data in real-time.


Literature Summarizer and Clustering using NLP and Deep Learning
0.6
0.4
0.2
0
−0.2
−0.4
−0.2
0
0.2
0.4

Chapter 5 – Evaluation

Overview

Due to the nature of the dataset and the methods used to process them, it is difficult to evaluate the project directly. Steps 1 and 4 of the application are mainly the processing of the dataset or the visualization output of the entire project. It provides little value to rigorously evaluate the results during these 2 stages. For step 2, the NLP is conducted with in-built functions from open-source packages. While they can be evaluated to determine the performance and results of these techniques on the data, the NLP processing is not the primary focus of this project. Hence, the only areas for evaluation would be during ML processing in step 3.

Evaluating Machine Learning Techniques

To evaluate machine learning and deep learning models, it is common to generate a naïve baseline to test the developed models against. However, due to the dataset and the processes taken in this project, it is difficult to generate such a baseline. Generating a baseline by manually checking and clustering would not be practical as it would require looking into every paper. Randomly clustering literature would likely produce clusters of data that do not actually relate to each other. Therefore, this project does not have a naïve baseline to compare against. Evaluations would have to be conducted within each ML technique performed.

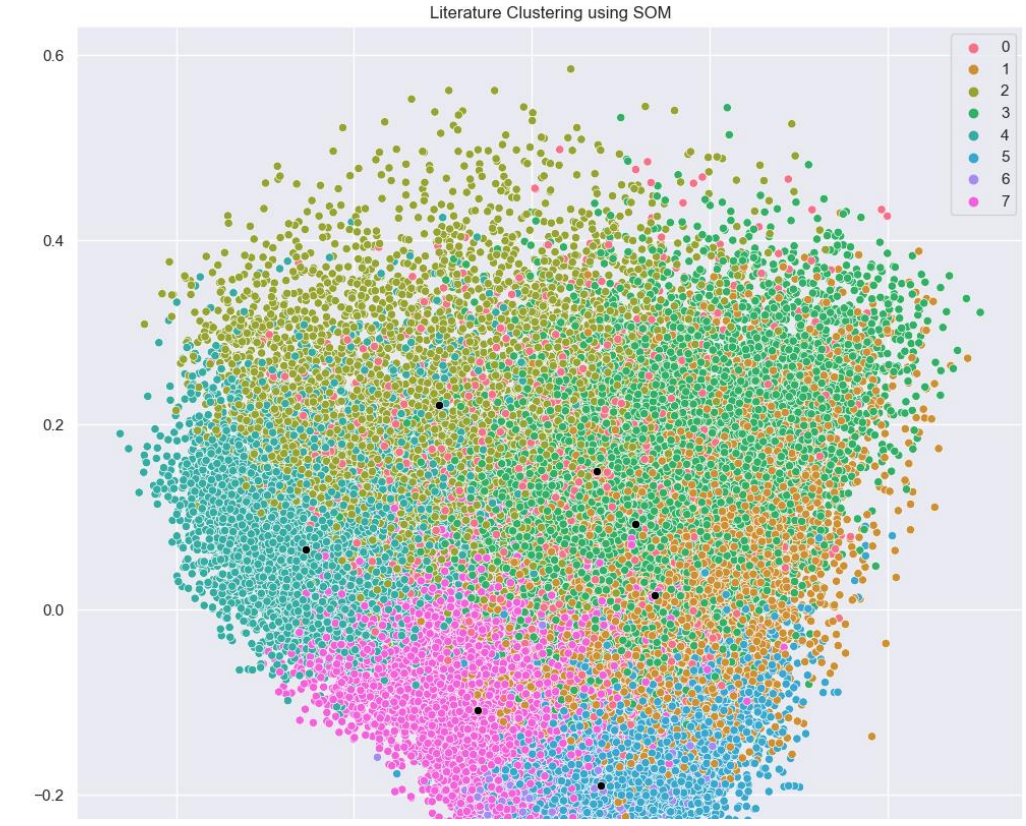
To evaluate the results and performance of the clustering algorithms, K-means clustering and Self Organizing Map (SOM) clustering, performing tests on it will be key. Both methods will be evaluated by these 2 metrics: Silhouette score and Davies-Bouldin index, as well as manually looking at the plot. The silhouette score measures the similarity a data point is within-cluster compared to other clusters, while the Davies-Bouldin index measures the average similarity of each cluster to its most similar cluster.

To determine the optimal number of clusters for either clustering algorithms, the use of the 2 metrics along with within-cluster sum of squares (WCSS) value for K-means clustering and the quantization error value for SOM clustering. WCSS value represents the sum squared distance between the data point and the centroid of a cluster. While the quantization error value represents the average distance between each data point and its corresponding best-matching unit. It is also a measure of how well the SOM preserves the details of the data. With its respective 3 methods, we can determine the optimal number of clusters to be used. However, the tests show poor performance from both algorithms and is difficult to determine an “optimal” number of clusters. Hence, the local best number of clusters is used but this is unlikely to be the model’s global best value. The tests are limited to a small range due to resource constraints.

The plots and their evaluations are shown below with the relevant test results. Both clustering algorithms have poor evaluation scores and represent poor clustering attempts at the dataset. This could be due to the limitation of the dataset, where academic papers often address multiple topics and are difficult to cluster into a singular topic.

Despite SOM clustering having a poorer silhouette score and Davies-Bouldin index, the visual representation of the clustering is significantly clearer and better defined. The plot using K-means clustering shows a messy visualization of the data points with few well defined clusters. Therefore, the SOM clustering is used for the rest of the project.


Literature Clustering using t-SNE and K-means
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
100
50
0
-50


Literature Clustering using SOM
0
1
2
3
4
5
6
7
0.6
0.4
0.2
0.0
−0.2

Evaluation of the Plots

1. K-Means Clustering

- **Evaluation Scores**: Silhouette Score: 0.0234; Davies-Bouldin Index: 4.7196
- **Overview**: By manually evaluating the results of the clustering, the application is able to loosely cluster groups of literature together with many outliers. As seen from the centroids of the clusters (black data points), they are all centred in the middle, away from any of the 'clearly' defined clusters and is likely to be incorrect.
- **Strengths**: There are multiple data points that are in 'clear' clusters. These 'clear' clusters are clusters that only 1-2 type of colour-coded data points congregate.
- **Limitations**: A 2-dimensional graph is unable to show the complexity and relations that the every literature has with each other. The clustering algorithms are only limited to classifying under 1 topic/cluster, but academic papers often do not only address 1 topic and hence is difficult to cleanly separate them.
- **Remarks**: The graph above shows the clusters that both k-means clustering and t-SNE has provided, both algorithms have clustered the literature independently and has resulted in some data points being far from their intended cluster (colour-coded points away from its cluster), lowering the accuracy of both clustering algorithm.

2. Self Organizing Map (SOM) Clustering

- **Evaluation Scores**: Silhouette Score: 0.0160; Davies-Bouldin Index: 5.8488
- **Overview**: The application is able to cluster groups of literature together with some outliers. The centroids of the clusters (black data points) are properly centered for each cluster.
- **Strengths**: Many of the papers are able to be clustered cleanly with little overlap. Manually evaluating the data points and comparing with the results of K-means clustering would show that SOM performs better.
- **Limitations**: There are certain clusters that are difficult to clearly define. A 2 dimensional graph may be limiting the algorithm's ability to effectively cluster and visualize that to the user.
- **Remarks**: Despite visually displaying better and clearly defined clusters, its evaluation scores are poorer than that of K-means clustering.

Comparing both plots and its scores, I have chosen to use the clusters produced by SOM clustering as it produces clearer clusters. The evaluation scores of both algorithms are poor and would not significantly affect the relations of the papers in each cluster.

Evaluating the Project

Successes

The success of this project is related to whether the application can meet the goals set out at the start of the project. And this application was able to meet 3 of the 4 goals: 1) Aid researchers in sieving through all the textual data in a piece of literature. 2) Group related literature together, helping researchers narrow their scope. 3) Create a data visualization tool to view the related literature.

The application was useful and able to create connections between related literature despite using different clustering algorithms independently. Surface-level examination of the clusters also shows connections between the data points in each cluster. This means that the entire process the project underwent is effective in generating clusters of related academic papers from the dataset used. Other successes would include having the project completed on Jupyter notebook, which allows the results to be shown without reprocessing the entire program, making the application "portable".

Failures

The application was not able to summarize the textual data presented, failing to meet one of the goals of this project. The current summary this project provides is from every academic paper's abstract, which would not be present in other forms of literature. Further use of NLP techniques with combination of deep learning models may be required to achieve this goal. Furthermore, the poor results of the clusters represent subpar clustering attempts and did not produce clearly defined clusters. The poor performance of the clustering makes this application difficult to use for researchers, individual discretion must be applied when using the application, further limiting the effectiveness of this project. There is also a high possibility of false positives, as it is very difficult to separate the different subjects/topics.

Limitations

Due to dataset used, academic papers often do not address only 1 topic, but the clustering algorithms can only classify under 1 topic. This creates inconsistencies and inaccuracies in the results of

the clustering algorithms. The NLP techniques used are designed in English, and this limits the pool of literature that can be used in this application to those only written in English.

Future Developments

There are many future developments that can be conducted on the different processes of this project to potentially improve the performance of the application. However, these future developments would require additional time to process, and additional evaluation must be conducted.

Step 2 processing can be developed using other text representation methods such as bag-of-words or singular value decomposition (SVD) instead of TF-IDF. While keeping all other algorithms used the same, the performance of the clustering algorithm can be evaluated and determined if the use of another text representation method affects the performance of the application.

Step 3 processing can be developed with other unsupervised classification algorithms. However, specific types of clustering algorithms known as overlapping clustering algorithms can be an area to be further explored. Overlapping clustering algorithms allow for clustering to consist of multiple topics, which is one of the major issues faced in this project. Developing the application with an overlapping clustering algorithm could produce better results for this project.

For step 4, adding additional features such as a search engine to find keywords and specific clusters can be implemented to help with the searching process. Or creating a 3-dimensional plot of the clusters and its data points could also provide a better understanding and experience when using this application to search for relevant literature.

Chapter 6 – Conclusion

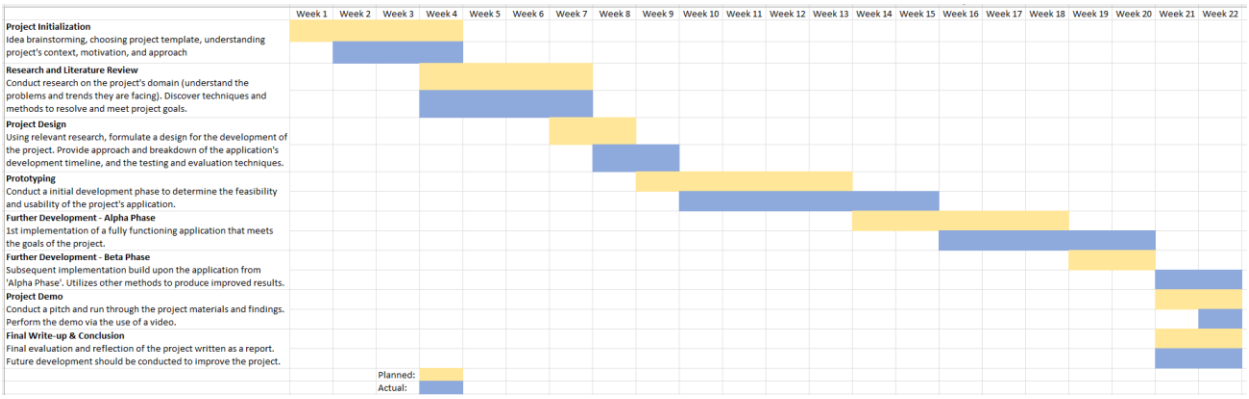
Summary

This project aimed to create a deep learning model and a software application that can help summarize and cluster literature together. The primary models used for this project are Natural Language Processing (NLP) and Machine Learning (ML) algorithms and techniques to process textual data to be clustered into groups. The dataset used for this project was a huge dataset of academic papers obtained from Kaggle, titled: "Covid-19 Open Research Dataset (CORD-19)".

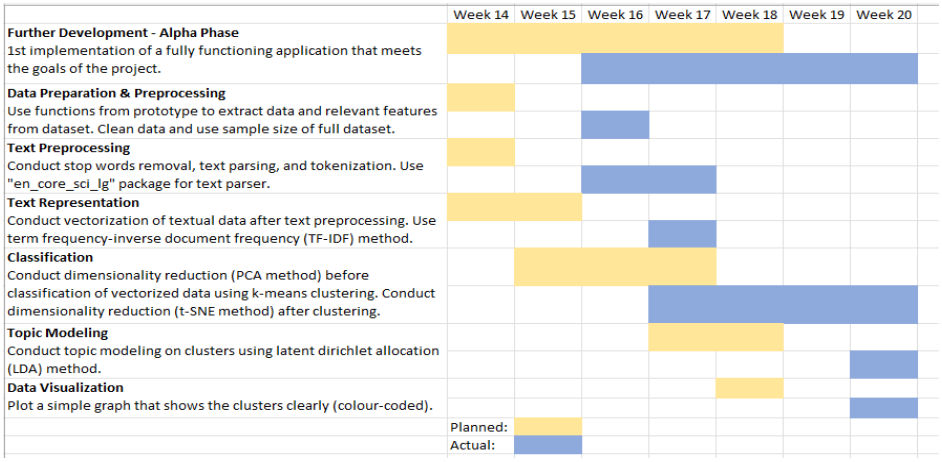
The application managed to read and store the data from the dataset, clean and use NLP techniques to further process the data. Machine learning techniques were then utilized to help cluster the literature files and were successfully plotted on an interactive graph. Users can directly interact with the graph to view the different literature and have its metadata shown for the user's convenience.

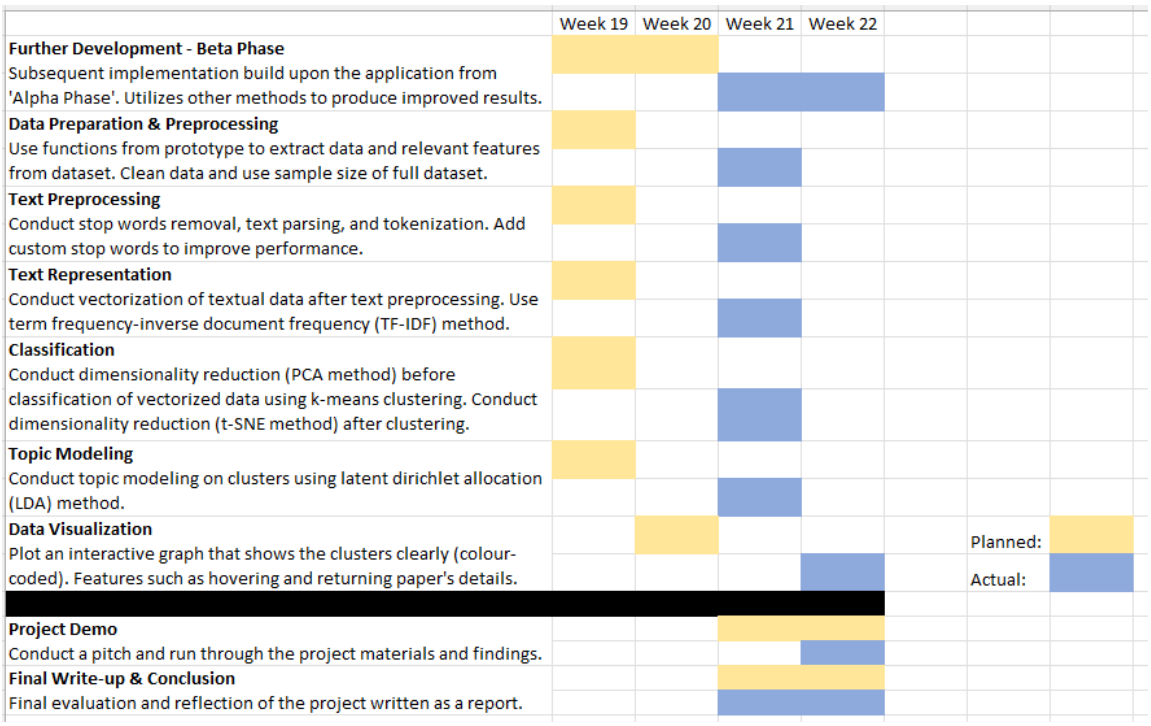
Timeline and Future Adaptations

With the tight timeline for this project, I have learnt that it is important to provide more time than expected for each component of the project. This would provide buffer time in the schedule such that when a component overruns the timeline, the buffer time can aid in that situation.



As seen in the Gantt chart above, I took too long and overrun my schedule during the prototyping stage by 2 weeks. These 2 weeks cascaded down to the rest of my project, potentially affecting the results of the project and the report. The 2 Gantt charts below are for the breakdown of my activities during the Alpha and Beta development phase respectively.





Future Work

If this project can produce good and useful results after future development on certain extensions as stated in chapter 5, there is potential for this project to be commercially viable. If further developments prove successful with this software, this can be used by researchers or students to help reduce their time spent on preliminary research and literature review. Schools and universities can use this application to help their students with this process.

Using other NLP and ML methods to fit general-purpose use instead of the specified methods used in this project can improve effectiveness of the clustering model on other forms of literature. Large-language models (LLMs) such as ChatGPT and Google's Bard/Gemini are very useful tools that this project can also utilize to improve its performance. Using LLMs to help conduct step 2 could produce better output for the clustering algorithms.