Final Year Project Interim Report
Semantic Search Engine – Industry Project

Team Members:
Adya Singh (3035452666)
SM Nazim Uddin Shoikot (3035445821)

Supervisor
Dr. Anthony Tam

Submission Date
24th January 2021
Abstract

The objective of this project is to build a Semantic Search Engine for the industry partner – Azeus Systems Limited. With a high volume of data in store, locating documents can become a tedious and time-consuming process when done manually. Therefore, search engines are the ideal solution for this use case as they allow users to enter search queries and retrieve relevant results. However, traditional search engines are typically designed to perform keyword matches which may not always return model results. Thus, the Semantic Search Engine bridges this gap by attempting to understand the intent of the user to retrieve the documents. This project collects, processes, and organizes data, accounting for similarity and nature of the original company dataset. Bidirectional Encoder Representation from Transformers (BERT) and various Sentence-BERT models are explored to create a Natural Language Processing model that retrieves documents effectively and efficiently. It combines that model with popular technologies such as ReactJS and Django to deliver a user-friendly stand-alone web-application for searching materials. Partial collection of data, extraction of text, and implementation of database have been completed, although they continue to be revised. A basic user-interface and preliminary functional NLP models have also been developed. However, substantial work is yet to be done to complete the software, including data processing, further development of web-application, and optimization of NLP models. During the project, several difficulties related to nature and size of the original dataset were encountered. The model also currently has several limitations related to large documents, dataset size, and retrieval time. Nevertheless, the project is currently on schedule and is making headway into building the solution.
Acknowledgements

We would like to thank my supervisor Dr. Anthony Tam for guiding and advising us in the process and giving us the opportunity to work on this project. This project would not have been possible without his suggestions and constructive feedback. We are also grateful for the support given to us by the project’s industry partner, AZEUS Systems Limited, throughout the development of this project.
Table of Contents

Abstract ................................................................................................................................. 2
Acknowledgements ............................................................................................................. 3
List of Figures ...................................................................................................................... 5
List of Tables ...................................................................................................................... 5
Abbreviations .................................................................................................................... 6

1 Introduction .................................................................................................................... 7
  1.1 Background ................................................................................................................ 7
  1.2 Project Objectives ..................................................................................................... 8
  1.3 Scope .......................................................................................................................... 8
  1.4 Deliverables ............................................................................................................... 9
  1.5 Outline ....................................................................................................................... 9

2 Methodology .................................................................................................................. 9
  2.1 Data .......................................................................................................................... 10
    2.1.1 Data collection .................................................................................................... 10
    2.1.2 Data preparation ................................................................................................. 11
    2.1.3 Splitting dataset to test addition of new documents ........................................ 11
    2.1.4 Text Extraction ................................................................................................. 11
    2.1.5 Database design ................................................................................................. 12
    2.1.6 Elasticsearch ..................................................................................................... 12
  2.2 Natural Language Processing Model ......................................................................... 12
    2.2.1 Google Dialogflow ............................................................................................ 12
    2.2.2 Bidirectional Encoder Representation from Transformers (BERT) ................. 13
    2.2.3 Variants of BERT ............................................................................................... 15
    2.2.4 Limitations of BERT ......................................................................................... 16
    2.2.5 Sentence-BERT (SBERT) .................................................................................. 17
  2.3 Web Application ........................................................................................................ 17
    2.3.1 Front-end .......................................................................................................... 18
    2.3.2 Back-end ............................................................................................................ 18
  3.1 Specification of the project ....................................................................................... 20
  3.2 Data and Database ................................................................................................... 20
  3.3 Text Extraction ........................................................................................................ 21
  3.4 Preliminary user-interface ........................................................................................ 22
  3.5 Experimentation of SBERT with different core models ...................................... 22
  3.6 Results ...................................................................................................................... 23
List of Figures

Figure 1: MLM learning [2] ....................................................................................................................... 14
Figure 2: MLM and NSP Training [3] ......................................................................................................... 14
Figure 3: Twin network architecture of SBERT [5] .................................................................................... 17
Figure 4: Django Architecture .................................................................................................................. 19
Figure 5: Layout of the basic user-interface ............................................................................................ 22

List of Tables

Table 1: Approximate distribution of documents in dataset........................................................................ 10
Table 2: Comparison of BERT, RoBERTa, and DistilBERT [4] ................................................................. 16
Table 3: Sample data and structure of table ............................................................................................... 21
Table 4: Summary of results of two models using threshold approach ..................................................... 24
Table 5: Summary of results of two models using relevance approach ................................................... 25
Table 6: Project Schedule ......................................................................................................................... 27
Abbreviations

AI – Artificial Intelligence
API – Application Programming Interface
BERT – Bidirectional Encoder Representation from Transformers
MLM – Masked Language Model
NLP – Natural Language Processing
NSP – Next Sentence Prediction
OCR – Optical Character Recognition
RoBERTa - Robustly Optimized BERT Approach
SSE – Semantic Search Engine
SBERT - Sentence-BERT
1 Introduction

This section first introduces the background and context of the project. It then discusses the primary objective, scope, and specific deliverables of the project before providing an outline for the remaining sections of the report.

1.1 Background

Efficient search of materials is at the core of the daily lives of individuals and businesses, making search engines and their underlying algorithms an area of focus. A traditional search engine is typically built to serve keyword searches, wherein it uses page ranking algorithms for search result retrieval followed by manual selection by users. However, the primary problem is that this approach does not evaluate the user’s intent. As such, it may be unable to fulfil the user's intended query. Moreover, it may often display a large number of irrelevant results, increasing the user’s browsing time before finding relevant results. These problems are usually particularly evident in searches using polysemy words which have multiple meanings depending on the context [1].

In an effort to solve such problems, Semantic Search Engines (SSEs) such as DuckDuckGo and Hakia have become increasingly popular. SSEs relate the words of a query with each other to understand the intent of the user. An SSE creates syntactic as well as semantic relations between different resources within the dataset, which helps it to search the data and provide accurate and meaningful results more effectively and efficiently.

Although keyword-based search engines still dominate public use, SSEs are particularly appealing to businesses. Company data is highly sensitive, so an internal search engine is ideal for maintaining data privacy. Since such data is likely to be domain-specific, more focused semantics can be defined for their data, which can make their internal SSE more efficient than a general search engine.

One of the companies who have realized the potential of SSEs is this project’s industry partner – Azeus Systems Limited (referred to as “Azeus”, hereafter), a Business IT Solutions
provider. Internal users including directors, managers, and employees regularly need to refer to a variety of documents. Currently, the data is organized in a file system where users have to manually look through the documents. To streamline the process, the company wants to develop an intelligent semantic search engine that can efficiently search their database and retrieve the materials.

1.2 Project Objectives

The project’s primary objective is to allow internal users of the company to efficiently search and retrieve company documents using conversation-like language. It will understand the intent of the user query and use it to return the queried documents efficiently.

1.3 Scope

The project will focus on developing the following core features of the software:

- Finding documents and creating search queries similar to Azeus dataset
- Designing and implementing a database to organize data
- Extracting plain text from files such as PDF, Images, Excel and PowerPoint
- Designing and creating a user-friendly web application
- Creating an NLP model for efficient document retrieval
- Testing and optimizing the NLP model according to dataset

Subject to time constraints, feasibility, and discussion with the industry partner, the following additional features might be developed for the software:

- Suggesting and autocompleting search queries according to user search history and recent popular searches on the engine.
- Implementing a speech to text feature so that users can query using voice command.

An additional feature of integrating the SSE with other company software was discussed. However, the integration process had considerable complexity and the confidentiality of the code
base of the other software was a significant issue. Hence, it was later decided that it would not be feasible to include this feature under the scope of this project.

1.4 Deliverables

Upon completion of the project, the following deliverables will be completed:

- An intuitive and user-friendly stand-alone web application where users can search, view and download results
- An NLP model that can understand the intent of the user from the query, and finally find and return the documents or materials that is being searched

1.5 Outline

The remaining sections discuss different aspects of the project in detail. Section 2 specifies the methodologies that are being used in the project. It discusses the collection, preparation and processing of data. It also explains the various technologies being used for developing the NLP model and the web-application. Lastly, it specifies the metrics and techniques that will be used to tune and evaluate the model. Section 3 highlights the current progress and results in terms of specification design, data collection, database design, text extraction, models, and a basic user-interface. Then, Section 4 explains the limitations and difficulties encountered such as data unavailability and limited use of testing metrics. Building on that, Section 5 discusses the future work to be done on the data, NLP model and the web-application. Finally, Section 6 provides a conclusion that summarizes all details of the project thus far.

2 Methodology

This project is being carried out in two major phases. The first phase focuses on data collection, data preparation and creation of the NLP model. The second phase consists of optimizing the NLP model and web application development.
2.1 Data

Collection and preparation of data is an important part of the project. Data of appropriate nature must be collected and processed. Finally, it has to be stored and organized in an optimized way that allows efficient retrieval of information.

2.1.1 Data collection

Creating an effective SSE requires a large amount of data. Ideally, the datasets would be provided by Azeus but due to the sensitive nature of data, Azeus was unable to provide internal company data. However, the nature of the dataset on which the model would be run was discussed and then emulated by manual collection and processing of data from public sources. The approximate distribution of the different types of documents is shown in Table 1. Plain text was later extracted and analyzed from these documents of different formats.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Approximate Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDFs</td>
<td>60%</td>
</tr>
<tr>
<td>Emails</td>
<td>20%</td>
</tr>
<tr>
<td>Images</td>
<td>10%</td>
</tr>
<tr>
<td>Excel</td>
<td>5%</td>
</tr>
<tr>
<td>PowerPoint</td>
<td>5%</td>
</tr>
</tbody>
</table>

Table 1: Approximate distribution of documents in dataset

While collecting data from public sources, it was important to ensure that the additional datasets were within the linguistic domain of Azeus’ dataset since the model will be optimized for the linguistic domain of its documents. The data collected mainly consisted of annual reports of information technology companies and an email dataset of Enron Corporation. Images, Excel and Powerpoint documents were collected through manual scraping of company websites and other similar sources. The collected documents were in the form of PDF, Email text, Images, Microsoft Excel, and Microsoft PowerPoint.
Furthermore, sample search queries were created that were later used to test the performance of the model. The search queries were created instead of collected from similar publicly available datasets to ensure that the queries are relevant to the datasets that the model is operating on.

### 2.1.2 Data preparation

In order to better represent Azeus’ dataset, annual reports were separated into more specific sections such that each document could highlight a different part of the business. This was done as annual reports in itself are not typical of documents found in the internal company documents. Furthermore, the Enron corporation email dataset presented a large quantity of emails. Therefore, only emails with sufficient content were selected such that every email in the dataset consists of meaningful information that could potentially be useful to the management.

### 2.1.3 Splitting dataset to test addition of new documents

Ideally, at the time of project completion, the dataset would have at least 3000-4000 documents overall. In practice, however, new documents will continuously be added to Azeus’ dataset. It would be helpful to replicate that behavior in the NLP model. Hence, about 10% of the total dataset would be reserved for testing the addition of documents. The models will be tested prior to the addition of such documents as well as upon addition. This will be done to check if it has any adverse effects on the performance of the SSE. Any related concerns would be addressed and resolved before the final implementation to ensure the robustness of the model.

### 2.1.4 Text Extraction

Since documents cannot be used directly, it is important to perform text extraction. This process involves extracting plaintext from various file formats. In order to achieve this, the project used Apache Tika, which offers an elegant solution for extracting plaintext from an array of formats.
2.1.5 Database design

The database design for the project is fairly simple and flexible, with only one table storing the name, embedding, and the location of the document. As the project progresses and new features and larger files are integrated into the system, the design will be appropriately adjusted in order to accommodate the changes. The current implementation is in PostgreSQL as it is compatible with both Azeus’ general database and can be easily integrated into the Django framework being used at the server-side.

2.1.6 Elasticsearch

As retrieval time is of the essence for an SSE, the project will explore the Elasticsearch framework alongside the existing database for storing the dataset. This allows for inverted indexing, which facilitates efficient and effective document search. While Elasticsearch itself is typically used for keyword matching, integrating it with NLP models will allow the overall model to grasp the user’s intent and serve as an SSE.

2.2 Natural Language Processing Model

The most challenging part of the project is to accurately understand the intent of user search queries and relate it to concepts in the documents. Natural Language Processing (NLP) will be used for this purpose. Google Dialogflow, a simple NLP tool, is first assessed as an option. Bidirectional Encoder Representation from Transformers (BERT), its modifications of RoBERTa and DistilBERT, and Sentence-BERT models are then discussed in detail.

2.2.1 Google Dialogflow

Google Dialogflow focuses on understanding the intent of the user when the interactions are in the context of a chatbot framework. It can be effective for sentiment and intent analysis, which could be helpful for the SSE since the queries will be in conversation-like language. However, its intent analysis is reliant on manual input of entities which will be searched. Since the dataset of SSE will be dynamic and new documents will continuously be added, maintaining the effectiveness of Google Dialogflow might prove to be infeasible. Moreover, subscription
payments are necessary to employ the premium version of Google Dialogflow. Given that there are better open-source alternatives, choosing a paid version is unlikely to be a good choice. More importantly, it is very simplistic in nature and can only be treated as an API. So, there is only a limited scope of customizing it to the dataset, which would pose a problem at the later stages of the project when the models need to be fine-tuned.

2.2.2 Bidirectional Encoder Representation from Transformers (BERT)

Considering the limitations of Google Dialogflow, Bidirectional Encoder Representation from Transformers (BERT) model is suitable for this project as it can be readily tuned and optimized for the domain-specific dataset. To provide an overview, BERT is based on the transformer architecture. The original transformer uses both encoders and decoders in its network; however, BERT only uses multiple encoders stacked on top of each other to create a representation of the input text in the form of a vector that can be analyzed for similarity analysis and other uses. In its training, BERT uses both Masked Language Model (MLM) and Next Sentence Prediction (NSP).

A normal language model learns by using a sequence of words to predict the next words. MLM, on the other hand, masks certain words from a sentence and learns by trying to predict these words using the remaining words of the sentence. Figure 1 presents a visualization of this methodology. NSP receives a pair of sentences and learns by using the first sentence to predict whether the second sentence follows the first sentence. Both MLM and NSP are used bidirectionally to train the BERT model, as demonstrated in Figure 2. BERT has been pre-trained using a total of 16GB data which consists of 3.3 Billion words, with 2.5B coming from Wikipedia and the other 0.8B coming from BooksCorpus. It can further be trained with domain-specific data to fine-tune the model and improve performance.
There are other models such as Long short-term memory (LSTM) models which try to understand the sentence sequentially instead. There were experiments done with bi-directional LSTM models which focused on training the model in the forwards and backwards direction and eventually combining the two approaches. However, despite this variation, the model is based on a sequential understanding which can often leave out important contextual clues. The bidirectional approach using MLM and NSP models allows BERT to understand the context, nuances of the language, and overall meaning of a sentence more accurately [3]. Moreover, BERT processes words simultaneously which makes it faster than its counterparts such as LSTM, which usually processes words consecutively.
2.2.3 Variants of BERT

Several variants have been developed in an effort to further improve the performance of BERT in terms of accuracy, training time, and execution time.

2.2.3.1 Robustly optimized BERT approach (RoBERTa)

RoBERTa is a modification of the original BERT model and employs a different training methodology. Instead of using the NSP model in its training, it uses dynamic masking such that the words being masked changes during different training epochs. Moreover, it uses 160 GB of text for pre-training, which includes the same 16GB used for BERT, and an additional 144 GB from various other datasets. It also uses larger batch-size, which was found to be helpful in training [4].

2.2.3.2 Distil-BERT

Distil-BERT tries to reduce the training and prediction times of BERT by learning a distilled, or approximate, version of BERT. It takes the trained original BERT model and uses a technique called distillation where it uses a smaller network to approximate the output distribution of the larger BERT network. It uses only half the number of layers and parameters compared to BERT but can retain about 97% of BERT’s accuracy [4]. Naturally, the smaller network leads to faster training and prediction times, but it compromises accuracy of the final results.

2.2.3.3 Comparison of BERT variants

Transformer architecture-based BERT was a major breakthrough in NLP and it usually outperforms all of its counterparts in most NLP tasks. RoBERTa was trained using a modified methodology and with a much larger dataset, which can allow it to give a 2-20% improvement over BERT in terms of accuracy. Distil-BERT compromises some of the accuracy in order to approximate the results of the original BERT and quicken the training and prediction times. It is usually used when a very fast retrieval time is needed but accuracy can be compromised. The comparison between these 3 are summarized in Table 2.
<table>
<thead>
<tr>
<th></th>
<th>BERT</th>
<th>RoBERTa</th>
<th>DistilBERT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>Outperforms state-of-the-art in Oct 2018</td>
<td>2-20% improvement over BERT</td>
<td>3% degradation from BERT</td>
</tr>
<tr>
<td>Data</td>
<td>16GB BERT data (Book Corpus + Wikipedia) 3.3 Billion words</td>
<td>160GB (16 GB BERT data + 144 additional)</td>
<td>16 GB BERT data. 3.3 Billion words.</td>
</tr>
<tr>
<td>Method</td>
<td>BERT (Bidirectional Transformer with MLM and NSP)</td>
<td>BERT with dynamic masking and without NSP</td>
<td>BERT Distillation</td>
</tr>
</tbody>
</table>

*Table 2: Comparison of BERT, RoBERTa, and DistilBERT [4]*

### 2.2.4 Limitations of BERT

Despite BERT’s significant performance improvement in the NLP applications realm, original BERT can still be quite slow in practical applications. For instance, if the two most similar sentences need to be found in a dataset containing 10,000 sentences, each sentence would need to be compared to every other sentence, which would require 49,995,000 passes through BERT, requiring approximately 60+ hours. Similarly, for the case of semantic search, for every query, the query would need to be compared with each of the 10,000 sentences in the database, which would take 40+ seconds on a standard computer [5]. The time taken is too long for the SSE to be useful in the application. The primary problem with BERT is that it needs to compare every sentence to measure the similarity. BERT’s variants, namely RoBERTa and Distil-BERT, also suffer from the same problem as the sentences undergo similar processing in them. The problem of excessive computation and time taken can be solved by a modification of BERT, called Sentence-BERT.
2.2.5 Sentence-BERT (SBERT)

SBERT improves on it by pre-computing vector-embeddings for each sentence individually. It is a twin-network which allows sentences to be processed separately but through the same base-model with same parameters, making it seem like the same model is run multiple times. After processing it with the base network, it is passed through a pooling layer to create a fixed-size vector for sentences of varying lengths, as shown in Figure X. Pooling can be done according to MEAN, MAX, or the CLS token BERT generates by default. The vector embeddings can then be compared using measures such as cosine-similarity to find vectors, and thus sentences, that are most similar to each other. This modification drastically reduces the computations and the time taken for retrieving materials in the SSE. For the same problem discussed earlier, it reduces the time taken to compare and find similar sentences to merely 5 seconds [5].

![Figure 3: Twin network architecture of SBERT [5]](image)

2.3 Web Application

This project aims to provide users with a user-friendly interface for them to make queries and view the retrieved documents. It will be built with ReactJS in the front-end and Django web framework in the backend.
2.3.1 Front-end

The stand-alone web application is in the form of a user-friendly and minimalist interface. A simple search bar is present where the user can type in a query. Upon pressing the “search” button, a sorted list of relevant documents will be displayed. Azeus has provided access to an internal software called Convene that is being used in the company. The general layout follows the aesthetic and user-interaction style of the other internal software to maintain consistency and coherence of company software design. Upon receiving the results of a query, users will have the option to preview the documents to quickly understand the nature of the document. The documents can then be downloaded for further storage and use.

This project uses ReactJS as the front-end development technology. This is because it has become increasingly popular for front-end development and is particularly efficient for single page applications such as the one for this project. With fast handling of user events and simplified interactions with the Document Object Model, it can significantly reduce the overhead. Additionally, it has numerous useful pre-built components which make the development process convenient and fast.

2.3.2 Back-end

The back-end has been implemented using Django, a Python web framework which uses a Model View Template (MVT) architecture as shown in Figure 4. This architecture outweighs the typical Model View Controller architecture as it allows for a cleaner program, discarding the manual implementation of the Controller for bridging the model and view. Furthermore, Django offers heightened security in comparison to its competing counterparts like Node JS, which is an important consideration based on the sensitive nature of the documents. Also, Django’s heightened efficiency and performance speed is a vital aspect since it is important to minimize the time between the input query and display of retrieved results.
2.4 Testing

A critical part of the project would be to ensure the accuracy and efficiency of the model through extensive testing. For effective testing, exact metrics and evaluation schemes need to be defined.

There are two distinctive parts of the model - (1) identifying the intent of the user from the query given, and (2) providing relevant documents. Purely assessing whether the intent has been correctly recognized is a significantly subjective matter and it might be difficult to effectively measure model accuracy using that. However, the retrieved documents lend themselves to quantifiability. Hence, the testing will emphasize on the accuracy of the retrieved documents.

To carry out the testing, a number of search queries would be made at random. Each query would be designed with respect to an associated document in the database, such that the document is an expected result of the query. Ideally, the associated document would be first among the retrieved documents. However, this may only work for a perfectly optimized model. Therefore, a threshold of rank will be set. If the associated document can be found ranking higher than the threshold, it can be considered to have passed the test. The SSE will initially aim to display the intended searched document in the top 10 results, but the threshold may be adjusted.

An alternate approach to test the model would be to assess the average relevance of the documents retrieved. These scores would be calculated by manually labelling the retrieved documents. While this procedure is prone to subjectivity, it is able to give some insight into the variance in model performance.
Finally, it is also essential to assess the time taken for the results to be retrieved. This will be examined by calculating the difference between the time the “Search” button is pressed and the time at which the list of documents is displayed on the screen.

3 Current Progress and Results

Significant progress has been made in terms of design specifications, data collection and preparation, NLP model construction and web application development. The data has been prepared and text has been extracted from documents, different SBERT models have been experimented with, and a basic layout of the user-interface has been created. Finally, the results from the preliminary implementation are also discussed.

3.1 Specification of the project

As this project is an industry project, many of the specifications and details of the project were initially unknown or open for discussion. During the early stages of the plan, the team met with the company representative online to try and understand the requirements of the software. The feasibility of the features was discussed with respect to the time allotted, confidentiality and availability of company data, and support from the industry partner. A full specification was created, discussed, and finally approved.

3.2 Data and Database

After approval of the specification and a discussion about the nature of Azeus’ dataset, data was collected from public sources such that it could represent Azeus' dataset. Although most company documents used in day-to-day operation are usually not available on public sources, annual reports are readily available on websites such as annualreports.com. Hence, annual reports of information technology companies were collected and later separated into different sections to represent common documents in Azeus’ dataset. A number of relevant technology industry reports were also gathered through online searching. Moreover, work emails of 150 users of senior management were collected from the available Enron corporation email dataset
which contains 500,000 emails. The emails were cleaned in order to only keep emails with sufficiently large content that could be useful to the management.

Additionally, numerous documents were assessed, and sample queries were constructed to search for those particular documents. The search queries and the associated document for every query was then used to test the performance of the model. Until now, around 1100 documents have been collected, prepared, and processed to create the preliminary implementation of the model. Search queries have also been created to test the preliminary implementation. Data collection and preparation will remain an ongoing process to improve and further test the model.

A simple database has also been designed and implemented using PostgreSQL. As shown in Table 3, it consists of a table which stores the document name, the binary representation of the vector embedding for the content of the document, and lastly the relative path of the document for later access.

<table>
<thead>
<tr>
<th>Document Name</th>
<th>Embedding</th>
<th>File Relative Path</th>
</tr>
</thead>
</table>
| file.pdf      | b'3<\x80?\xa8\x02\xbf\xfd\x0e\xed>l\xf8\xad=\xa1^\xa1?\xe6\xf3\xa6>\x1fD\xdd>?
                   w\x16\xbf\xa2~?\... | directory/folder/file.pdf |

Table 3: Sample data and structure of table

3.3 Text Extraction

Another task completed in the first phase involved text extraction. This was done in order to obtain plain text from other formats as this will assist in training the NLP model. In documents which contain images or tables, the software only extracts headings/captions for analysis. Also, in the scenario that the PDF consists of scanned documents or images of text, Optical Character Recognition (OCR) is used to perform the text extraction. The project utilizes Apache Tika for this task, which offers a metadata extraction Application Programming Interface (API) including OCR extraction for both text and images. This is the ideal tool as it can be used across file formats, reducing implementation overhead to deal with each format individually.
3.4 Preliminary user-interface

A preliminary user-interface has been designed and created using ReactJS, following the aesthetic and user-interaction style of Convene and other internal software. Figure 5 shows the design of the user-interface, highlighting the search-bar that will be used for searching. After retrieval of documents, the title and a preview of the content of the document (yet to be implemented) will be shown to the user.

![Figure 5: Layout of the basic user-interface](image)

3.5 Experimentation of SBERT with different core models

Four different SBERT models were initially experimented with. These differed in the core model (BERT, RoBERTa, or Distil-BERT) that they used for embedding, and the size of the fixed-size vectors they outputted.

The first two were bert-base-nli-mean-tokens (BERT-base) and bert-large-nli-mean-tokens (BERT-large) respectively. Both use the original BERT at its core and a means pooling layer to encode the sentences into fixed-size vectors. The key difference between them is BERT-
base represents sentences using vectors of size 768 whereas the size is 1024 for BERT-large. Due to the smaller size of vectors, time taken for training and prediction is smaller for BERT-base. However, the documents in the dataset of this project tend to be comparatively large even after splitting into separate sections. As such, a larger vector size is able to better represent the “sentences” or documents, which lead to more accurate results in general. Despite the fact that the larger size of vectors increases the training and retrieval time of the model, the results are increasingly accurate, making BERT-large the preferred model compared to BERT-base.

Next, distilbert-base-nli-mean-tokens (DistilBERT-base) was implemented to see if similar accuracy could be achieved using Distil-BERT which uses a smaller network to approximate results. Although this model predicted results more quickly, it demonstrated a considerable decline in accuracy. Since accuracy was of greater significance at this stage and the small decrease in retrieval time did not justify the degree of lower accuracy, BERT-large outperforms this variant.

Finally, a roberta-large-nli-mean-tokens (RoBERTa-large) model was implemented. It created the embedded vectors using the modification of RoBERTa instead of the original BERT as discussed in earlier sections. Since it was established that a larger vector size was preferred for the project, RoBERTa-base, with vector size 768, was not experimented with. The performance of BERT-large and RoBERTa-large was very similar on the dataset, making it difficult to conclusively select one of the models at this stage. So, both the models will be further explored and a final model will be selected based on the assessed performance upon fine-tuning.

3.6 Results

According to research and exploration, BERT-large and RoBERTa-large were chosen to be the preferred models. Test cases were individually run for each of the two models and their performance was compared using two different metrics. This section summarizes the results.

3.6.1 Threshold Approach

A search query was designed for certain chosen documents such that upon entering the query, the respective document should be returned. If the associated document was among the 10
retrieved documents for that query, that search query qualified as a passed test case. Otherwise, it constituted a failed test case. The performance of both the models was the same using this approach, with each of them passing 80% of the test cases, as shown in Table 4. Average time taken for each of the models was also similar, with BERT-large taking 10.4 seconds on average and RoBERTa-large taking 10.3 seconds. The overall performance is acceptable considering the current stage of the project. However, the accuracy needs to be further improved. Moreover, as the size of the dataset increases, it might be challenging to maintain the performance in terms of both accuracy and retrieval time.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
<th>Average retrieval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT-large</td>
<td>80%</td>
<td>10.4s</td>
</tr>
<tr>
<td>RoBERTa-large</td>
<td>80%</td>
<td>10.3s</td>
</tr>
</tbody>
</table>

*Table 4: Summary of results of two models using threshold approach*

### 3.6.2 Relevance Approach

Since a binary assessment of the occurrence or lack thereof of the document may not necessarily be sufficient, an alternative approach was also considered. For every query, the top 10 documents were manually scored on a scale of 1-5, where 1 represents not relevant and 5 represents highly relevant. This process was repeated for several queries and the results are indicated in Table 5. Both BERT-large and RoBERTa showcased similar performance in terms of the average relevance score and true positives across all queries. True positives refer to documents which are ranked above 1, and therefore are somewhat relevant, which show in the query results. Therefore, while these results are not definitive of the better model, it does take into consideration an alternate testing metric.
### Table 5: Summary of results of two models using relevance approach

<table>
<thead>
<tr>
<th>Model</th>
<th>Average Relevance Score</th>
<th>True Positive</th>
<th>Average time to return results</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT-large</td>
<td>3.5/5</td>
<td>90%</td>
<td>10.5s</td>
</tr>
<tr>
<td>RoBERTa</td>
<td>3.6/5</td>
<td>90%</td>
<td>10.2s</td>
</tr>
</tbody>
</table>

### 4 Challenges and Limitations

Throughout the project, many challenges and limitations have arisen across the designing, planning and implementation phases.

#### 4.1 Size and nature of dataset

The most consequential challenge is the lack of data. Upon further discussion with the industry partner, it was determined that sharing the internal documents was not a feasible option due to confidentiality and sensitivity reasons. As a result, manual data collection had to be undertaken. However, since the SSE depends on the nature of the data, care must be taken to ensure that the nature and dependencies of the original data is not distorted by the new data. In order to ensure the acquisition of accurate and fully representative data, the data collection was done manually which is a significantly time-consuming process. Furthermore, it was important to collect a significant amount of data which was representative of a variety of different formats.

#### 4.2 NLP Model

While the model shows a good performance overall, there are certain limitations at play. Firstly, the model is inaccurate when dealing with larger documents. This is primarily due to the fact that common word embeddings are bound to be found with greater frequency in large documents even if the document as a whole is not highly relevant to the search query. As a result, the document is shown in the results irrespective of the relevancy. One proposed solution to resolve this issue is to divide larger documents into smaller segments which will be searched upon.
Another challenge would be to ensure that the performance of the model remains consistent despite large datasets. As more data is collected and added to the database, the model will be tested for accuracy and overall performance.

Additionally, it is important to ensure that the documents are retrieved in a short period of time. While the model itself is able to achieve this, the web application displays these results with a significant delay. Therefore, optimization of the web application along with the indexed database are likely to reduce this time immensely.

4.3 Testing

Following from the unavailability of data, testing the NLP model appeared to be a significant challenge. Ideally, metrics such as a confusion matrix would be suitable to assess the accuracy of the documents retrieved. However, as the data is unlabeled, this metric no longer remains feasible. Additionally, given the nature of the documents, it is difficult to automate the labelling process. The relevance testing methodology is inherently subjective as the relevancy ranking of documents is not objective. Nevertheless, if time permits, a small subsection of the dataset will be chosen, and manual labelling will be conducted with respect to each search query. Each document will be both - (1) given a binary label indicating whether or not it should appear in a given query and (2) given a general category which describes the overarching theme of the document. This will allow for a more objective measure while testing the performance of the SSE.

5 Future Work

Significant headway has been made in the first half of the project. The next phase will focus on optimizing the models, finalizing testing metrics and testing the models, completing the development of the web application and implementing an indexed database with Elasticsearch. Additionally, the data collection process will remain ongoing so as to improve the performance of the model. The completed items, and the schedule for the remaining work are outlined in Table 6.
<table>
<thead>
<tr>
<th>Tentative Completion Month</th>
<th>Tasks</th>
</tr>
</thead>
</table>
| September 2020            | 1. Initial meeting with industry partner – Complete  
2. Research as per industry partner requirements – Complete  
3. Project Plan – Complete  
4. Project Webpage – Complete |
| October 2020              | 1. Receive and analyze dataset from industry partner – Complete  
2. Create additional datasets – Complete  
3. Perform text extraction from documents – Complete  
4. Design database – Ongoing |
| November 2020             | 1. Implement initial database – Complete  
2. Begin Implementation of NLP model – Complete  
3. Begin implementation of web app architecture – Complete |
| December 2020             | 1. Begin implementation of web app frontend – Complete  
2. Complete preliminary implementation of NLP model – Complete |
| January 2021              | 1. First presentation – Complete  
2. Detailed interim report – Complete  
3. Integrate NLP model with web application – Ongoing |
| February 2021             | 1. Complete development of web application – Ongoing  
2. Manual Labelling of data  
(subject to time constraints) - To be done  
3. Perform testing and fine-tune model – Ongoing |
| March 2021                | 1. Implement indexed database - To be done  
2. Invite industry partner to test the prototype and address any concerns – To be done  
3. Consolidate all moving parts – To be done |
| April 2021                | 1. Final Report – To be done  
2. Final Presentation – To be done  
3. Final tested implementation – To be done |
| May 2021                  | 1. Project Exhibition – To be done |

*Table 6: Project Schedule*
6 Conclusion

This industry project aims to build a Semantic Search Engine for internal users of Azeus Systems Limited using effective and efficient technologies of NLP and Web Application development. Upon experimentation with various BERT models, the best performing models namely BERT-large and RoBERTa will be fine-tuned. A final model will be chosen based on the performance of the optimized models. Moreover, a user-friendly web-application will be developed using ReactJS and Django which allows users to query, view, and download documents through the final NLP model.

Significant progress has been made in terms of data collection, data preparation, text extraction, and database design. However, these are ongoing processes subject to revision, and more data is required to further improve the model. A basic implementation of NLP models as well as web application is also completed. A number of challenges in terms of representing the size and nature of Azeus’ dataset was faced in the absence of the actual company dataset. The model also currently has some limitations in dealing with size of dataset, large documents, and retrieval time. Nevertheless, more documents are being collected carefully considering Azeus dataset nature and the models are being optimized to mitigate the current hurdles. Additionally, the project is emphasizing on efficient data storage via Elasticsearch to ensure quick retrieval times.

The immediate next step is to begin optimizing the NLP model. This will involve adding domain specific vocabulary to each model. Additionally, the web application construction will remain ongoing and will also be optimized to minimize the time taken to retrieve the documents. Despite the difficulties, the project remains on schedule and project progress is expected to be at a faster pace since the overall foundation of the project has been established and solidified.
References


