SUMMARIZING BIOMEDICAL EVIDENCE

Sahithi Kodali

Bachelor of Engineering
Software Engineering

School of Engineering
Macquarie University

June 27 2021

Supervisor: Dr.Diego Molla-Aliod

ACKNOWLEDGMENTS

I would like to express my sincere and deepest gratitude to my thesis supervisor Dr.Diego Molla-Aliod for providing me the opportunity to work on this research project and guiding me in every step of my research journey by providing insightful comments, valuable feedback and encouragement. I would be forever in debt for all the time and support provided in my very first research project.

I would also like to extend my heartfelt thanks to Dr.Gaurav Gupta for all the guidance and mentoring throughout my thesis, ensuring that I met the deadlines by providing assistance in times of hardship.

Special thanks to my family and friends for supporting me in the difficult phases encountered throughout my thesis, and encouraging me to give my best in the hard times of life.

STATEMENT OF CANDIDATE

I, Sahithi Kodali, declare that this report, submitted as part of the requirement for the award of Bachelor of Engineering (Honours) in the School of Engineering, Macquarie University, is entirely my own work unless otherwise referenced or acknowledged. This document has not been submitted for qualification or assessment in any academic institution.

Student's Name: Sahithi Kodali

Student's Signature: K.Sahithi

Date: 27 June, 2021

ABSTRACT

In the recent years, immense growth of information in various formats like text, images, sources on web etc. can be observed evidently. To handle the enormous growth of information, the field of Human Language Technologies (HLT) aims at developing machines that can understand and interpret natural languages. Text Summarization (TS) is a task which is focused on extracting the important and relevant information from a given set of documents in order to reduce the tedious task of reading the whole documents. These summaries act as a surrogate for whole documents, thus enabling the complex intelligent systems to reduce the processing time, while handling the vast information efficiently. The approaches of summarizing documents varies based on specific tasks and needs in different domains. Lately, the research in summarizing the Biomedical documents has geared up to enhance the healthcare system. BioASQ organization hosts challenges related to the Semantic Indexing and Question Answering tasks to develop solutions that can improve the access of information to the experts in the biomedical domain. In this thesis, the Query based Extractive Summarization techniques are discussed with a focus on the state-of-the-art BERT architecture. Pre-trained models are utilized to analyze if use of BERT based models alone, to reduce the complexity of the model produce better performing summaries. The systems built based on this research idea are discussed in this thesis, along with showcasing some of the best ROUGE scores obtained in BioASQ submissions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Human Language Technologies (HLT) is an interdisciplinary field which involves a wide range of activities with an end goal of enabling people to communicate with the machines efficiently using natural language or natural communication skills [105]. The common challenge in these tasks are dealing with natural language, understanding its nature and the context it has been developed in. Every language has its own structure, properties and phenomena which is to be considered while developing these machines [57]. Adding to this, due to the rapid growth of the Internet, there has been a tremendous increase in the information available in different formats like videos, images, etc. along with the text, curving this research into a complex horizon. To handle this massive availability of information, HLT has intelligent applications ranging from simple, general tasks dealing with whole documents to complex tasks dealing with specific words in the documents. Few of the well known intelligent applications based on HLT are Information Retrieval (IR), Text Classification (TC), Question Answering (QA) and Text Summarisation (TS) defined as follows.

1. **Information Retrieval:** Information Retrieval is the method of retrieving relevant information to a specific need, given a collection of large documents usually in the text format [64].

2. **Text Classification:** Text Classification is the method of classifying documents into specific categories as required given a predefined set of documents. Thus, it can also be termed as Text Categorisation [94].

3. **Question Answering:** Question Answering is the method of providing precise answers to simple and complex questions in Natural language. This is carried on by accessing a collection of documents or a database related to the question [100].

4. **Text Summarization:** A method of producing a brief summary of the document or required sections of the document by selecting and condensing the important information only. These summaries act as a surrogate of the original document [43].

Further in this chapter we will discuss the scope of this thesis in section 1.1 and structure of this thesis in section 1.2.

1.1 Scope of the thesis

This thesis extensively discuss the query based extractive summarization approaches that are proposed in the BioASQ Task-B Question Answering challenge. The main goal of this task is to produce answers to the questions related to Biomedical domain as summaries. These BioASQ approaches are further discussed in chapter 3. Biomedical Question Answering (QA) is a complex and challenging task due to the requirement of domain expertise and the vast amount of unstructured data available. Text Summarization can make this task of Question Answering much easier and can produce better results by handling the information available efficiently. To understand the approaches combining the Text Summarization with Question Answering known as Query-based summarization, this thesis explains Text Summarization and its evolution, types of summarization techniques and various approaches used to generate summaries with a special focus on Extractive Summarization and Question Answering approaches.

Adding to this, we will also discuss the approaches and techniques used by different systems submitted in the BioASQ Task-B challenge with a special focus on the architectures based on BERT and BioBERT — A state-of-the-art in Natural Language Processing (NLP) tasks like Summarization and Question Answering. Further, the experiments conducted in this thesis to handle the research question *"Can pre-trained models based on BERT and BioBERT alone, reducing the architecture complexity produce better summaries?"* are discussed, along with evaluating the results obtained by these experiments in BioASQ submissions.

1.2 Structure of the Thesis

This thesis is structured in six chapters. *Chapter 1* includes introduction and scope of the thesis. *Chapter 2* includes the background and research work required for the thesis with a detailed description of Text Summarization and its approaches, followed by introduction to BioASQ tasks. *Chapter 3* details the approaches and architectures used by various systems in BioASQ Task B challenge to produce Query based Extractive summaries with a special focus on Macquarie University contribution, which is followed by introducing BERT/ BioBERT and their pre-defined models. *Chapter 4* details the experiments conducted to handle the research question of the thesis. This includes the details of the software setup and libraries used, data collection and processing, training, testing and evaluation of the model. *Chapter 5* discuss the results obtained by the experiments conducted and analysis of their performance. Concluding this thesis, *Chapter 6* explains the output of the thesis work and the future work that can be conducted to improve the results produced.

Chapter 2

Background and Related Work

In the previous chapter, the scope of the thesis with its structure is discussed. This chapter explains the background and prior research conducted, explaining the Text Summarization and its approaches in section 2.1 followed by introducing the BioASQ challenges in section 2.2. This chapter has major of its contents based on the paper by Lloret & Pastor [59] along with the updates of recent research work.

2.1 Text Summarization

Text Summarization (TS) aims at producing a brief summary of the document or required sections of the document by selecting and condensing the important and relevant information only [43]. Though it evolved in the late 50's, the research in TS has showed immense progress in the recent years by proposing a wide range of techniques to tackle TS challenges. Generating an automated summary is a challenging tasks and comes with many issues like redundancy, temporal dimensions, co-reference etc. These problems appear to be more challenging when dealing with a set of documents termed as Multi-document Summarization. Summaries often present the lack of logic or consistency known as coherence and research to overcome this has been fuelled in the last few years. This resulted in the development of combined approaches that can identify the relevant content and merge them into new fragments of information, thus reducing the lack of coherence [59].

With the evolving changes in the society, TS has to adopt to the changing requirements. The introduction of Web 2.0 (Social Web) led to the birth of new types of information resources like websites, blogs, social media, forums where people have the right to speech to express their own ideas towards any issue or product. This change in information resources led to the introduction of new types of TS known as *Sentiment based summaries* generated based on the user opinions. An important aspect to be concerned about in Text summarization is Evaluation. Evaluation is a difficult task due to the uncertainties in deciding what information to be included in a summary and this process is certainly challenging when it needs to be automated [74]. It has been shown that the methods for evaluating summaries automatically correlate well with the human evaluation, but issues

while evaluating the quality of summary like redundancy, grammar and coherence stills needs to be addressed. Few of the automatic evaluation metrics for TS are ROUGE [56] and AutoSummENGG [34], with the use of ROUGE metrics in this thesis.

Further, the factors to be considered to generate TS and types of TS, approaches used to generate summaries, approaches combining TS with QA and the evaluation metrics are discussed in this section.

2.1.1 Overview

Text Summarization (TS) process varies depending on different factors. One of the well known existing taxonomy states three classes of context factors that influence the process of TS namely;

1.) *Input factors*: Deals with the aspects related to the source like genre, language etc.

2.) *Purpose factors*: Deals with the aspects of target audience and end use of summary.

3.) *Output factors*: Deals with the aspects of the style of summary and its extent of coverage. Usually driven by the Purpose factors [43].

The second taxonomy suggested is based on the relevant aspects like characteristics of — source document, summary as text, use of summary, thus including factors related to coherence and subjectivity level of summaries produced [42]. The third taxonomy is focused on the approach adopted to generate the summaries in order to classify the summarization systems, which says a problem can be solved using different approaches at three levels namely;

1.) *Surface level:* Aims at representing the information of summary in terms of shallow features like frequency counts.

2.) *Entity level:* Aims at building the internal representation of words in the document i.e model the entities and their relationships thus providing patterns of connectivity.

3.) *Discourse level:* Aims at modelling the global structure of the document which includes concerns about the format of the document, subtopics in documents. It attempts to capture the structure of different types of text involved like Narrative (personal experience and stories) or Argumentative (deliver an idea, usually with proofs or evidence) [61].

The first two taxonomies based on factors and characteristics classify summaries with respect to different criteria related to the nature known as fine-grained granularity, while the third taxonomy based on the levels group the summary approaches based on types of features and techniques used to generate the summaries. The problem with dealing with fine-grained granularity occurs when classifying the summarization system based on

their defined criteria of characteristics occurs while sharing multiple characteristics in a single system, turning this classification complex and unclear. The problem with the third taxonomy dealing with levels is the purity of classification. It is shown that the current classification systems rely on *hybrid approaches* combining different level features [59]. There are different types of summaries based on various factors as shown in Table 2.1.

Table 2.1: Factors affecting the type of summary produced [59]

Media	Text, Images, Video, Speech, Hypertext
Input	Single, Multi document
Output	Extract, Abstract, Headline
Purpose	Generic, Personalized, Query-focused, Update, Sentiment-based, Indicative, Informative, Critical
Language	Mono, Multi and Cross Lingual

This thesis focus on *Multi-documents* as input, to generate the *Extractive summaries* as output with the purpose of summaries being *Query focused*. Combining these, the goal of this thesis is to explore the state-of-the-art in generating *Query-based Extractive summaries* from multiple documents and conduct research to improve current performance.

There has been a noticeable research on summarizing text documents specifically newswire, by extracting relevant sentences from documents to generate summaries known as Extractive Summarization. However, with the emerging changes in the society and huge sources of information available, new scenarios like literary text (stories and books), image captioning (short description of image), patents, web 2.0 (blogs, reviews, social media etc). have developed. To handle these novel domains, new types of summarization techniques are developed, based on the objectives of the summaries that are produced.

i. *Personalized summaries* are specific to the user interests and requires prior analysis of the user profile .

ii. *Update summaries* are generated by assuming that the user has a background knowledge of the topic, hence provides only the latest updates related to the topic.

iii. *Sentiment summaries* are produced based on the subjectivity, polarity seen in the document and are related to the feelings/opinion of person about a product, service or place etc.

iv. *Survey summaries* provide a generalized overview of a topic or entity, which are generally long due to the inclusion of important facts concerning the topic.

v. *Abstractive summaries* produced by extracting the relevant sentences along with adding new text to handle coherence [59].

Abstractive and Extractive Summarization are differentiated further, as it is essential to understand the differences between these two approaches considering their importance in this thesis.

Abstractive and Extractive Summaries

Extractive Summaries are produced by extracting the relevant sentences from documents without any changes to the sentences. This lacks coherence and is prone to dangling anaphora. To overcome this problem, Abstractive summaries are produced by extracting the relevant sentences along with adding new text. This is essential to handle the repetition of information and, the methods suggested to generate these summaries are sentence compression [113], sentence fusion [12], natural language generation [85]. Evaluation of abstractive summaries are shown to be 78% more coherent than extractive summaries [59]. Producing abstracts for Biomedical documents is explored using SemRep[1] by identifying semantic predicates [30]. Machine Learning approaches using different types of classifiers are experimented and Natural Language Generation (NLG)[2] module is also applied to produce these summaries [111]. Other approaches like using information extraction system and identification of patterns [18] are also experimented to produce natural looking Abstractive summaries.

Though Abstractive summaries has the capability to produce better quality summaries, the underlying complexities in the natural language makes this task both challenging and difficult [35]. Due to these reasons, Extractive summaries and the techniques used to generate these summaries are focused in this thesis.

2.1.2 Approaches to generate summaries

In this section, the approaches used in various studies to generate Extractive summaries are discussed . The process of generating summaries can be divided into three main stages namely; topic identification, topic interpretation and the summary generation [84].

1.) *Topic Identification:* Involves identifying the main theme of the document by determining the structure of the texts and topics included.

2.) *Text Interpretation:* Involves distinguishing between relevant and irrelevant information to derive meaning of content in text,

3.) *Summary generation:* Aims at generating the summary by extracting and combining all the relevant information obtained from the first two stages.

The first two stages involves producing the extracts from the text i.e extract the sentences in the same way as they appear in the original document. Hence, most of the approaches focus on the first two stages due to the complexity of generating a final summary [59].

The five different types of extractive summary approaches based on diverse techniques are as follows.

[1] https://semrep.nlm.nih.gov/
[2] https://pypi.org/project/nlg/

1.) *Statistical-based approaches* generate summaries based on numerical factors like frequency count of the words in a sentence, to extract the relevant sentences in a document as summary [65].

2.) *Topic based approaches* produce summaries by identifying cue words like "In conclusion", thus determining the relevance of sentence by the words or phrase the sentence contains [26].

3.) *Graph based approaches* utilize the Graph based ranking algorithms, where the nodes of the graph are the words or sentence and the relationships (Synonymity, Anonymity etc.) between the words are given through the edges links [28].

4.) *Discourse based approaches* exploit the rhetorical or discourse relation between sentences in a document i.e determine the logical flow [63].

5.) *Machine Learning Approaches* are produced by utilizing the machine learning algorithms to generate summaries.

Further Machine Learning (ML) approaches are discussed in detail, due to the vital role payed by ML algorithms in query-based extractive summarization techniques used in this thesis, particularly related to the Deep Learning (DL) frameworks.

Machine Learning Approaches

Binary classifiers [51], Hidden Markov Models [22] and Bayesian methods [10] are the first machine learning methods used for Text Summarization. A single document summarizer known as NetSum [98], used RankNet [15] as a learning algorithm to produce extracts from the news-wire documents based on the neuronal nets and score the sentences to extract the highest scored sentences as summary. Though the common features considered from ML approaches are the keywords and the position of the sentence, a new set of features have evolved based on the query logs and Wikipedia information i.e. the sentences consisting of the Wikipedia entities and queries were given more importance. FastSum [92] is a multi-document summarizer which ranks the sentences using Support Vector Regression (SVR), a ML algorithm that uses Least Angle Regression for the selection of features. Features which are word-based, phrase-based, semantic based along with the position of the sentences or named entities are used to train the SVM classifier automatically in this approach [55].

Before the emergence of Deep Learning approaches to handle Extractive summaries, extractive summarization approaches based on supervised and semi-supervised methods has been proposed. The supervised approaches used the labelled data and are based on SVM algorithm, while semi-supervised approaches consist of a Naive Bayesian Classifier and probabilistic SVM co-trained to deal with unlabelled data. SVM algorithm is used to extract the relevant information to be included based on a query in a query-focused summarization [32, 110]. These classifiers showed some noticeable improvements in the field

of Text summarization in early research. Though use of the Machine learning algorithms for Text Summarization has an advantage of testing the performance on a high number of features learnt, the downside is the requirement of a huge training corpus or data consisting of annotated source documents or human written summaries with clear details of what is and not important for a summary, to obtain efficient results.

Adding to these, due to the immense increase in the information available, several recent advancements can be observed in the Machine Learning approaches. Techniques using Deep Learning and automatic dimensionality reduction have become prevalent, thus producing much better representation of learning. The use of pre-trained and fine-tuned language representation models like BERT, BioBERT showed state-of-the-art results in many summarization approaches and techniques [37]. The latest advancement in the approaches based on Deep Learning are discussed in Section 3.2 in Chapter 3.

2.1.3 Text Summarization with Question Answering

Text Summarization (TS) can be applied to intelligent systems like Information Retrieval (IR) [83], Text Classification (TC) [49] and Question Answering (QA) for reducing the processing time of these applications by providing summaries, thus eliminating the need of dealing with whole documents. This sort of evaluating the summaries indirectly is known as extrinsic evaluation further discussed in this section. Adding to this, summaries represent the important information in a document, thus allowing better efficiency of the systems. Further, the ways of using TS with Question Answering to understand the generation of query-focused summaries is explained.

Question Answering (QA) is the process of answering simple or complex questions in natural language automatically [100]. There has been a wide research in working with integration of Question Answering and Summarization approaches. An approach of multi-document QA summarizing was suggested using the topic signatures [72]. However, this was not sufficient as Topic signatures contained emphasized information of the text only, while summaries made by human includes other information along with the emphasized information. Approaches combining Information Retrieval and the Text Summarization techniques were experimented to answer the biomedical domain questions [23]. Questions are tackled by extracting information related to the question and clustering them to extract a short summary for each cluster. The final summary including the title, main intervention and top scoring sentences are produced using the Machine Learning Techniques.

BioSquash system [96] is a medical domain system based on a generic summarizer, which aims at answering a question by summarizing multiple biomedical documents. It has four main components namely; Annotator, Concept Similarity, Extractor and Editor modules each performing specific tasks to generate a final summary. *QAAS system* [101] is a combination of TS and QA systems and is based on a generic multi-document summarizer in which multiple compression rates are coupled with the QA system to reduce the

space when compared to a full document, thus improving the number of correct answers obtained. Generic summarizer is also used to identify the informative text span in the documents, but due to the identification of few limitations, generic summarizer is adopted to the query-focused tasks through query expansion and re-scoring the sentences generated as answer. Some of the other approaches are using QA for TS in an inverse fashion. In these approaches, QA system determines the importance of sentences using the scores and a set of queries, whose output is integrated into a generic multi-document summarizer to generate the final summary [73].

Considering the change in current format of information available, techniques to improve the inclusion of semantic relations, ontology based systems and diverse ranking algorithms have developed to handle Question Answering [50]. The use of pre-trained language models have showed state-of-the-art results in handling the Question Answering tasks [24, 54]. Few of the approaches and techniques used by different systems to tackle these query based summaries in the BioASQ challenge context are discussed in Chapter 3.

2.1.4 Text Summarization Evaluation

TS evaluation computes the goodness of summary generated by evaluating its quality and informativeness [44]. The two main types of methods known are:

1.) *Intrinsic Evaluation* focus on the summary itself i.e content information and comparing to human-models which are the reference summaries [62].

2.) *Extrinsic Evaluation* focus on the effectiveness of summarisation system on other applications like IR, TC, QA [39].

To evaluate the informativeness and quality of the summary there are several metrics used, which are further discussed.

Informativeness Evaluation

Some of the well known metrics for this evaluation are precision (fraction of the actual relevant sentences retrieved to the total number of relevant sentences retrieved), recall (fraction of the actual relevant sentences retrieved to the total number of relevant sentences existing) and F-measure (a harmonic mean of Precision and Recall) [53, 75]. Other metrics used are Relative Utility [86] to determine suitability of sentence, Factoid scores [99], Pyramid method through Summary Content Units (SCU) [31,79] that needs manual effort hence known as *Semi-automated methods*. Few *Automated methods* that can be used are QARLA framework [9], Basic elements [41], ParaEval [114], AutoSummENG [34], DEPEVALsumm [77], GEMS [46], WordNet and HowNet eval [106] and ROUGE [56] scores.

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is an automatic evaluation metric that obtains Precision, Recall and F1 score. It uses the co-occurrence of n-grams

and compares summary content with reference summaries to check the number of n-grams of words that are in common. This is based on the hypothesis that *'texts sharing similar words/phrases indicate the texts similarity in their meaning'*. Based on the types of n-grams, variants like ROUGE-1 (unigrams), ROUGE-2 (bigrams), ROUGE-L (Longest Common sub-sequence), ROUGE-SU (Skip-bigram plus unigram), ROUGE-SU4 etc. can be computed [59]. In this thesis, ROUGE-2 and ROUGE-SU4 scores are used to evaluate the summaries produced by comparing with the human produced summaries discussed in Section 4.5 in Chapter 4.

Quality Evaluation

The problem with the methods discussed above is, they do not consider the important aspects like coherence or non-redundancy. To assess these, the methods using FAN Protocol which take features like number of anaphora deprived referents, tautological aspects and legibility [67], MLUCE Protocol, tools accessing linguistic quality [21], readability factors [80], coherence [11], Centering Theories [40], automatic evaluation techniques [104] are experimented. Further investigation on structure and coherence aspects based on lexical and semantic aspects are also investigated [103] to obtain better quality summaries.

The main challenge occurs when evaluating the Extractive summaries with the human written ones due to their abstract nature of summaries. To evaluate these, ROUGE scores are computed and compared to human written summary score as reference, which showed better performance [59]. When manual evaluation is performed involving different people, it still exhibited a bias due to subjectivity. Though ROUGE is shown to be a good evaluation tool for automated summaries, there are few criticisms against it [58] which can be considered in future research. The evaluation based on ROUGE scores (with extensive focus on ROUGE-SU4 scores) can be observed in the summarization approaches discussed in Chapter 4.

2.2 BioASQ

Previously various concepts related to Text Summarization have been discussed. This thesis is focused on query based extractive summarization approaches proposed by several systems in the BioASQ Task B challenge. BioASQ[3] hosts a series of challenges related to the large-scale biomedical Semantic Indexing and Question Answering, starting in the year 2013. The goal of BioASQ is to enable the medical experts to easily access the information related to any specific diagnosis or conditions in an efficient way [102]. BioASQ hosts challenges related to three tasks namely;

[3] http://bioasq.org

Task-A: Focused on Large-scale Biomedical Semantic Indexing.

Task-B: Focused on Biomedical Semantic Question Answering (includes intelligent applications like Information Retrieval, Text Classification and Text Summarization etc.).

Task-MESINESP: Focused on Semantic Indexing in Spanish.

Task-B is further divided into two sub-tasks/phases, namely;

Phase-A: Focused on Identification of the relevant information.

Phase-B: Focused on Question Answering Tasks.

In this thesis, various approaches used by different systems proposed in Task-B Phase-B Question Answering challenge are analyzed. The observations made are considered as the research base to develop a system that can provide better results.

2.3 Conclusion

Concluding this chapter, a discussion on Text Summarization and their types, approaches used to generate summaries with special focus on Extractive Summarization, approaches combining TS with QA and evaluation of summaries, with a basic overview of BioASQ challenge is showcased. Further, Query based Extractive Summarization approaches in BioASQ context are discussed in Chapter 3.

Chapter 3

Query-based Extractive Summarization Approaches in BioASQ

In the previous chapter, various Text summarization approaches along with an introduction to BioASQ challenge has been discussed. This chapter explains Query-based Extractive Summarisation approaches and techniques used by different systems in the past four editions of BioASQ to produce ideal Answers with a special focus on Macquarie University suggested systems, due to the similar framework used in the runs of BioASQ.

Query-based Extractive Summarization is the process of generating answers by extracting the relevant sentences related to a query, from a given text document or set of sentences without paraphrasing or adding additional words. Task-B Phase-B Question Answering tasks involves four types of questions namely; questions with answer as *Yes/No*, answers in the form of a *List*, answers in the form of single terms or facts known as *Factoid* and answer in the form of a *Summary* with the goal of obtaining the Exact and Ideal answers. This thesis focus on generating Ideal answers as summary. Ideal answers are the expansion of exact answers, including more information in the form of full sentences. The approaches and experiments range from simple experiments to complex approaches based on Deep Learning, Reinforcement Learning frameworks using Regression and Classification setups. These involved the use of pre-defined language models ranging from simple word2vectors, tf-idf to the use of variations of BERT, BioBERT, Siamese networks, LSTM. The systems developed are evaluated using the ROUGE metrics Recall, Precision and F1 scores with the major use of ROUGE-SU4 F1 score.

This chapter begins with discussing the baseline approaches in Section 3.1, Deep Learning approaches in Section 3.2 and Reinforcement Learning approaches in Section 3.3. This is followed by a detailed explanation of BERT and its deriving architectures, along with BERT based pre-defined models used in the thesis experiments in Section 3.4.

3.1 Baseline Approaches

Few baseline approaches used in the systems submitted to BioASQ returns the top 50 and 100 snippets for the question using the Support Vector Regression model. Two different approaches used are, the Greedy approach which extracts the highest ranked sentences until the maximum length constraint is reached, while the other adds Integer Linear Programming to this approach including the sentence position feature. The results showed that Integer Linear Programming obtained better results due to better optimization [33].

Similar to this, a baseline approach *'Trivial'* proposed by Macquarie system simply returned the first *'n'* snippets of a question which is observed to outperform the approaches in other domains like news as well [14]. Improving this, an approach termed as *'simple'* is suggested where *'n'* snippets that are most similar to the question are returned by determining the Cosine similarity between the question and the snippets, whose vectors were computed using word2vec and tf-idf followed by Singular Value Decomposition (SVD). It is observed that the first trivial approach obtained the best results, however inclusion of similarity obtained through word2vec obtained the next best results compared to vectors computed using tf-idf [68]. The results of these baseline approaches can be compared to the Deep Learning and Reinforcement Learning approaches discussed further.

3.2 Deep Learning Approaches

The system proposed by **Carnegie Mellon University**, in the fifth edition of BioASQ used a system pipeline consisting of three stages as follows.

1. *Question-sentence relevance ranker:* Three aspects are considered to determine the ranking for the relevant sentences i.e. *Granularity* as to choosing abstracts or snippets of the question, *Similarity metrics* computed using tf-idf or soft/hard positional constraints or jaccard similarity followed by the incorporation of *Biomedical tools and Ontologies* like UMLS Metathesaurus [93] and SNOMEDCT [97] for concept expansion by identifying the biomedical concepts from sentences.

2. *Sentence selection:* Ranking is followed by sentence selection using algorithms like Agglomerative clustering and Maximal Marginal Relevance (MMR) [17] to handle redundancy.

3. *Sentence Tiling:* The final stage of generating summary using Concatenation or Compression is known as Sentence Tiling [29].

Through various experiments combining these methods above, it was observed that the selection of snippets rather than abstracts improved the ROUGE score. MMR is shown to be more effective in selecting sentences, with more relevance to question compared to agglomerative clustering. It also introduced a novel similarity metric of combining the semantic information through word embeddings and tf-idf statistics, which when used

with soft positional constraints are shown to produce better results [19].

Improving this in the sixth edition, *Ontology based retrieval system* is added to the current best system. In this improved system, the questions were pre-processed using NLP standard techniques like POS taggers, Named Entity Recognition (NER) and Medical Entity Recognition (MER), which is followed by Ontology Retrieval to obtain the relevant snippets of a question. These snippets are ranked using a ranking module and sent to the sentence selector which is based on the best system obtained previously (*MMR algorithm and soft positional constraint*), thus obtaining the final summary using selected sentences. The Ontology Retrieval Module is a graphical model developed to improve the recall by utilizing entity and relation extraction methods to compare questions and answers. The ranking module is based on LEarning TO Rank (LETOR) [82] method where two approaches were experimented i.e. RankSVM [16] and List-wise Neural Ranking with estimation of BM25 scores between snippets and ideal answer. This approach also involved the use of Pointer Generator Coverage (PGC) [95] which is shown to produce high scores to abstract summaries due to the use of sequence models. Fine tuning this network on BioASQ data is shown to produce efficient results than pre-training the network on CNN/Daily Mail. Overall, a neural ranking module along with the domain specific features like fine tuning PGC is shown to obtain better results [50].

The system named **UNCC** proposed in the sixth edition of BioASQ showed best ROUGE scores compared to the other systems. Methods based on MetaMap and UMLS use conceptual representations to generate the summary [88]. These approaches used *Lexical chaining* (to determine the words that are semantically related) to predict the sentence similarity and rank. This is done my segmenting the sentences (using Stanford CoreNLP) and passing these to a *MetaMap tool*, which identifies all the biomedical entities and returns their name and semantic type. The scores are determined by grouping semantic types and biomedical entities, and calculating their intersection, thus selecting the highest scored sentences for final summary. MetaMap also plays role in tokenizing, connecting to UMLS to acquire mappings required, along with word sense disambiguation [13, 76].

Further, a broad discussion on the architectures used in the systems proposed by **Macquarie University** can be observed. These systems are analyzed in detail due to the similar Neural Nets framework used in architectures, while improving them in every edition of BioASQ using different techniques. The system proposed by Macquarie University in the fifth edition of BioASQ focused mainly on the Regression and Deep Learning approaches. The regression approaches are based on the Support Vector Regression (SVR) algorithm, with its setup and features based on the best results of BioASQ 3b run by Malakasiotis et al. [60]. The F1 scores of ROUGE-SU4 evaluation metric of each candidate sentence are used as target scores to train the SVR system. Instead of using just snippets as input like in baseline approaches, all the sentences in the source document along with the information of whether a sentence is a snippet or not are considered as input.

The steps followed in the general framework of summarising are:

i. The input text i.e. source documents or snippets given are split into candidate sentences and each sentence is scored separately.

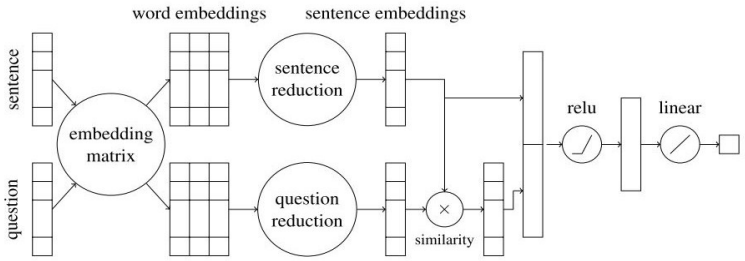
ii. The *'n'* sentences with the highest score are returned as summary, where the value of *'n'* is depended on the type of the question and is determined based on the observations [68].

The features used to train the system are as follows [68]:

- tf-idf vector of the candidate sentence based on the text of questions, ideal answers and snippets.
- Cosine similarity between tf-idf vectors of the question and the candidate sentence.
- Smallest cosine similarity between tf-idf vectors of candidate sentence and each of the snippets related to question (Not used in 3b run [60])
- Cosine similarity between sum of word2vec embeddings of words in question and the candidate sentence respectively.
- Pairwise cosine similarity between word vectors of question and candidate sentence, computed using word2vec embeddings. The features selected are: mean, median, maximum, minimum of all pairwise cosines similarities and means of the highest two, highest three, lowest two, lowest three [68].
- Weighted pairwise similarities by multiplying each word vector with its tf-idf and compute pairwise cosine similarities whose mean, median, maximum, minimum, mean of highest two, highest three, mean of lowest two and mean of lowest three are used as features [68].

Following this, a Deep Learning approach using Regression setup is experimented. The source documents in PubMed abstracts format are split into sentences in the pre-processing stage which are fed as token identifiers to the system. The architecture of this system is shown in Figure 3.1.

Figure 3.1: Architecture of the Deep Learning approach under the regression setup [68]



As shown in Figure 3.1, the pre-processed candidate sentence and question are fed into the system to convert into word embeddings using the *embedding matrix*. These are further reduced into sentence embeddings combining the word embeddings using a *reductor*. The similarity between sentence embeddings of sentence and question are compared to obtain the similarity vector. This is combined with the sentence embeddings for the final regression setup involving one hidden layer ReLU (Rectifier Linear Units) and a linear combination. Using the pre-trained word2vec, the word embeddings are produced and encoded into embedding matrix. Optimising weights by backpropogation did not show any improvement, hence a constant embedding matrix of size 100 is chosen after experimenting. The sentence embeddings are obtained using three different approaches as follows.

i. *Mean* of word embeddings.

ii. *Convolutional Neural Networks (CNN)* applied to word sequences to extract important 'n' grams.

iii. A bi-directional variant of *Long Short Term Memory (LSTM)*, a complex Recurrent Neural Networks (RNN) where four distinct set of weights are used. Two weights assigned to each of the forward and backward LSTM chains due to the difference in behavior of the words in question and candidate sentence, while the other to consider the context of the left and right sides of each word as well.

The similarity between the candidate sentence and the question is calculated using two methods: weighted dot product and a method used by Yu et al. [112] which allows interaction between sentence components by allowing the weights to learn by backpropogation [68]. Improving this in the sixth edition, Deep learning architectures under the same regression setup are experimented with an additional feature of sentence position (due to the better performance in trivial- top n sentence approach) and the input data is restricted to only the snippets relevant to the question. This is shown to improve results and hence these features are carried on to the following editions [70].

In the seventh edition of BioASQ, the system proposed by Macquarie University involved classification approaches i.e, the training procedures are based on the classification setups rather than regression setups. The general framework for summarising can be structured as follows:

i. Training the classifier to predict the target label of the input sentence i.e summary or not summary.

ii. Select the sentences predicted as 'Summary' to produce the final summary.

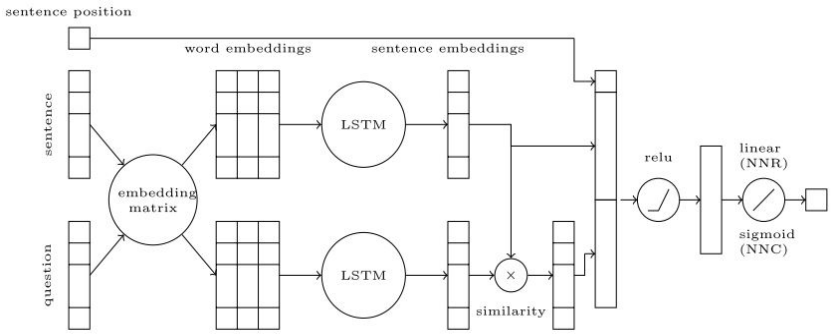
iii. If the selected sentences are less than n, n sentences with higher probability of 'summary' label are selected, where n is obtained based on empirical observations [69].

The two approaches used for labelling the training data are:

1.) *Threshold t*: All the sentences in input text with a ROUGE-SU4 F1 score above threshold t are labelled as 'summary'.

2.) *Top m*: The m input text sentences with highest ROGUE-SU4 F1 score are labelled as 'summary'.

Figure 3.2: Architecture of the Deep Learning approach under the regression and classification setup [69]



The variations used in this architecture are the Neural Regressor using Regression setup and Neural classifier using top 5 labelling method (ROUGE-SU4 F1 score as the classification labels and regression labels respectively). The Regressor approach used Support Vector Regression and the Classifier approach used Support Vector Classification, while the input features are same as used in the Regression approach with an addition of the position of the snippet as feature. The 'top 5' labelling approach showed better performance and the reason for this is expected to be the use of all sentences instead of just the snippets. A sigmoid activation function is added to the final layer of the architecture and the loss function used is cross-entropy. The architecture of this system is as shown in Figure 3.2 [69].

Analysis of the systems

Comparing the systems above, it can be observed that the systems using domain specific language models obtained better results when compared to the usage of other domain irrelevant models [50]. Adding to this, the use of MetaMap in the system 'UNCC' is shown to be problematic when MetaMap does not capture all the biomedical entities that are needed to be grouped [13], which can be future research scope. Through the observations based on Macquarie University improvements, it can be said that Classification setups produced better results than regression frameworks. This is expected to be so due to the ROUGE values ranging between 0 and 1, thus matching the probability values of sigmoid activation function of classifiers [69].

Most of the systems proposed have a minor change like change in the 'similarity functions' when calculating the similarity between the question and snippet vectors [19]. From this observation, it can be said that similarity metric/function plays a key role in the performance of the model and incorporating the context features in these functions can produce better performance of the summarization systems. However, it is worth observing that the better scores of ROUGE do not assure the better scores of manual evaluation. Most of the systems that attained high ROUGE-SU4 scores appear to produce lower performance in terms of the Readability metric of Manual evaluation scores in BioASQ. This can be addressed in future research to produce better manual evaluation scores by incorporating advanced semantic and context related algorithms.

3.3 Reinforcement Learning Approaches

In the sixth, seventh and eighth editions of BioASQ, along with the Deep learning approaches Macquarie University experimented with Reinforcement approaches based on a proof-of-concept prototype used to train a global policy (a neural network using tf-idf features encoding the question, candidate sentence and context) using REINFORCE algorithm. After experimenting with Deep learning approaches where the model is trained on the sentences annotated individually to obtain a summary of top 'n' sentences, an upper bound of these results are considered in two way. The general approach of returning the *'k'* snippets with highest individual ROUGE score or returning the combination of *'k'* snippets with the highest collective ROUGE scores, by calculating the scores of every combination of snippets. It is shown that the *'k'* snippets with highest individual scores outperformed the collective score method [70].

Reinforcement Learning (RL) allows training the system iteratively by considering the ROUGE scores of summaries produced, as an evaluation metric. A summary is extracted using the current policy and the policy is updated based on the feedback from ROUGE scores of the summary produced. The global policy is trained on the training set of development data which is implemented as a simple neural network with a hidden layer using a variant of REINFORCE [109]. The policy predicts the probability of not selecting a sentence by learning a set of parameters through a neural network. The parameters and the input for the network is obtained by concatenating the features determined based on the experiments [109], along with adding the length of summary [70]. The features considered are detailed below.

i. tf-idf vector of each candidate sentence.

ii. tf-idf vector of the entire input text.

iii. tf-idf vector of the summary that has been generated so far.

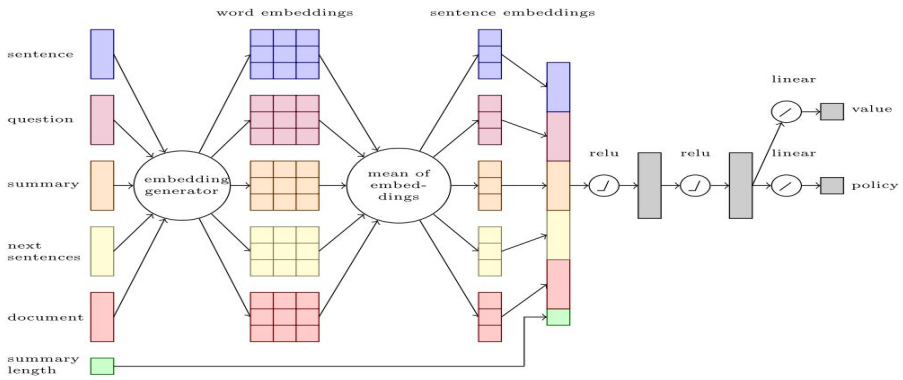
iv. tf-idf vector of the candidate sentence that are not yet processed.

v. tf-idf vector of the question.

vi. The length of the summary generated so far.

The added feature of length of summary generated so far is shown to improve the learning time of the policy, thus producing better results [70]. Observing the better results of RL approaches with human evaluations, few improvements are incorporated in the following seventh edition. The architecture using same features with a minor change of the use of word2vec to compute word embeddings, instead of tf-idf vector features are used in the system. Along with these, both Python ROUGE and Perl implementation variants are experimented which showed no improvement [69]. In the eight edition, a new approach is experimented using Proximal Policy Optimisation (PPO) instead of REINFORCE algorithm, due to the ability of PPO to penalise changes to the policy, obtaining better learning curve than REINFORCE.

Figure 3.3: Architecture of Reinforcement Learning approach using PPO [25]



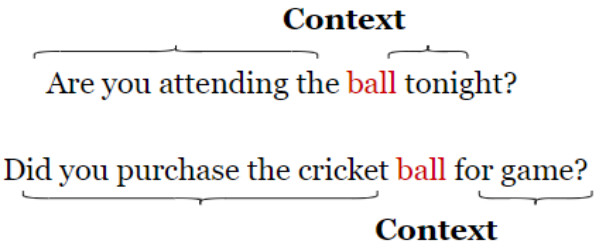
As shown in the architecture in Figure 3.3, the input features are as stated previously and the sentence embeddings are generated by computing mean of word embeddings obtained using word2vec. A layer obtained by concatenating the sentence embeddings and the length of the summary is fed to the Multi-Layer Perceptron with two hidden layers, whose outputs are the stochastic policy function and the value function with a liner activation function for each. These represent the predicted future rewards and action probabilities of the two actions of classifying 0 or 1. The RL approach using REINFORCE classifies the sentences into either 1 or 0, indicating if the sentence is included in the summary or not respectively, based on the reward of the ROUGE-SU4 F1 scores directly [25].

3.4 BERT Architecture

Bi-directional Encoder Representations from Transformers (BERT) is a pre-trained language representation model that overcomes the limitation of the models being unidirectional by using a Masked Language Model (MLM). In this model, few tokens are hidden to predict the embedding based on the left and right words context [24]. Consider the

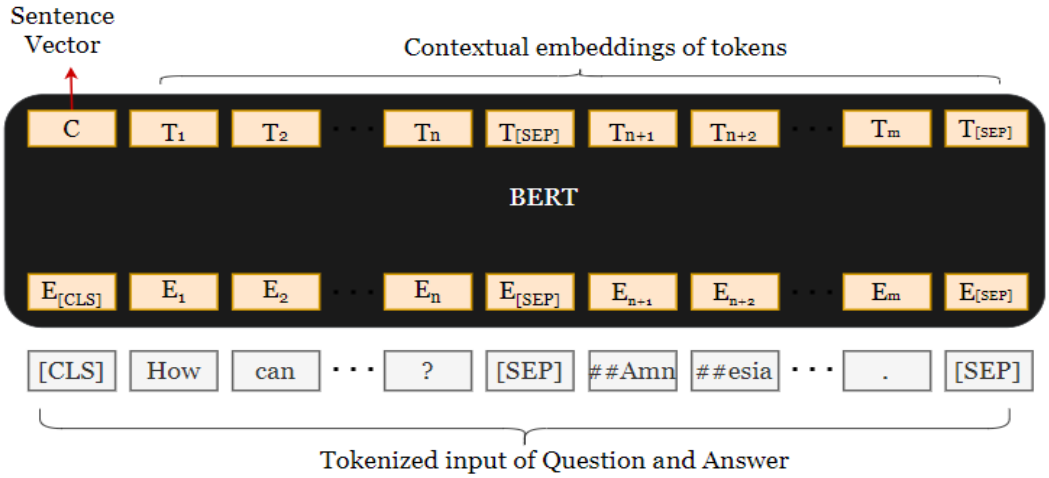
example in Figure 3.4, where the word 'ball' has two different meanings — an event and a sports ball. BERT model considers the words to the right and left side of the word focused, to provide an embedding based on the context.

Figure 3.4: Example of using multiple meaning words in a sentence



The inputs in the form of tokens are sent into the BERT model to obtain contextual embeddings of the words in the sentence, which are further used on the downstream tasks as needed. Classification token '[CLS]', added in the beginning of each input sentence passed to the model is a sentence vector, that acts as a sentence representative. This token provides the probability of the sentence that can be used in Classification and Next sentence prediction tasks. This diagrammatic overview of how the inputs are given to the complex BERT model to obtain contextual embeddings can be seen in Figure 3.5.

Figure 3.5: Overview of input and output in BERT model [38]



BERT architectures are complex. The black box of the BERT model architecture involves a series of encoder representations from transformers. Each encoder layer takes input from the previous encoder layer and outputs an intermediate representation of the word. Each encoder layer has an attention layer followed by a feed-forward network. Attention layer

assigns an embedding to the word by assessing its importance in the sentence. This architecture can be seen in Figure 3.6. The self-attention process involves a series of mathematical calculations to compute the embeddings which can be further studied in the paper by Jay Alammar [8]. The number of encoder layers depends on the chosen model configuration.

Models

Models are intelligent architectures based on diverse algorithms that takes specific input data, learn the features of the data in order to predict the outputs for unseen data known as testing data.

A model architecture is built based on the task requirements. Pre-defined models are used to perform experiments in this thesis. These models are trained on vast amounts of data and the best weights are stored. These saved models can be further fine-tuned on the task specific data. There are various benefits of using pre-defined models compared to building a model from scratch and training it. Some of these are as listed below.

- Reduce the time spent on making design decisions like how many layers are needed, order of layers, pooling, calculating various statistical, mathematical operations etc. in order to find the best suitable model architecture [20].
- Avoiding the lack of budget and computational resources required to train model on huge data corpus, along with avoiding the lack of enough data availability for efficient model training [20].
- The major task of fine-tuning the model parameters to determine the best weights of the model can be avoided, as the best weights are already made available [20].

BertForSequenceClassification

BertForSequenceClassification is a pre-defined model provided by HuggingFace Transformers [1]. This model involves a classification/regression head and can be used for either of the tasks. In this thesis, this model is used with a classification head and the architecture of the model can be defined by importing a pre-trained model.

The model begins by taking word and token type embedding as inputs, along with the position embeddings. The essential inputs required for the model are as follows.

1. *Word indices/Input ID's:* The tokens obtained after tokenizing are converted into numerical format for the machine to understand. These are known as Word indices/Input ids/embedding for the model.

[1] https://huggingface.co/transformers/

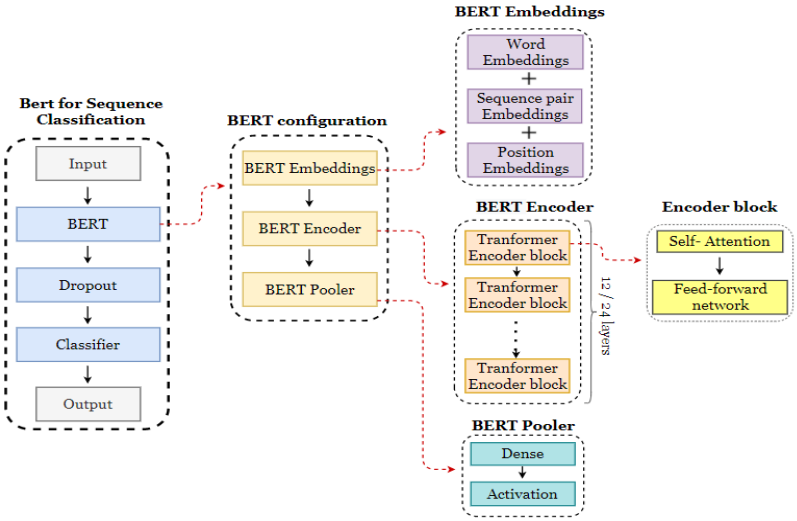
2. *Attention masks:* Masking is performed on the words that are used to pad in order to maintain the same length for sentences. This way the words that does not have meaning are not given attention. Hence, attention masks are in the from 1's and 0's with '1' representing the actual word and '0' representing the padded words.

3. *Segment ID's/Token type ID's:* Segment ID's are used to differentiate the first sentence from a sentence following it. This will help the model to learn the difference between the input sentences. These are in the form of alternating sequences of 0 and 1, changing when a new input sentences starts.

4. *Position embeddings:* These are the sequential numbers given to the tokens for determining their position in the given input sentence.

The method of providing the inputs and the explanation of the architecture of the model used in this thesis can be observed in Section 4.3 in Chapter 4. The architecture of BertForSequenceClassification model can be seen in Figure 3.6.

Figure 3.6: Internal working of BertForSequenceClassification model



The BertForSequenceClassification model begins with a BERT layer that take inputs to obtain contextual embeddings, followed by a Dropout layer that assists in adding noise to network to reduce over-fitting of the model and ends with a final classifier layer to categorize/score the sentences as required. The 3 layers in BERT configuration have different functionalities. The embeddings are processed in the embedding layer and forwarded to the series of encoder layers with attention layer inside each encoder which works as explained previously. BERT Pooler performs pooling using Tanh() activation function which provide outputs in the range (-1,1).

The number of Transformer encoder blocks depends on the chosen predefined models for the task. With BERT obtaining state-of-the-art results as discussed in the previous chapters, the pre-trained models based on BERT are chosen for this thesis experiments and are explained further.

3.4.1 BERT pre-defined models

BERT based pre-trained models are provided by the Transformers library. These models can be imported and fine-tuned on the task specific data. Two of the pre-trained models chosen for this thesis experiments are bert-base-uncased and bert-large-uncased.

Bert-base-uncased

Bert-base-uncased is a model trained on lower-cased English text on 4 cloud TPUs for 4 days. It consists of 12 bert layers (transformer encoder blocks) with 768-hidden layers, 12 attention heads and 110 million parameters [1].

As explained before, model begins with an embedding layer where the word embeddings, token type embeddings along with position embeddings are summed up and sent as input. These undergo computation in every transformer encoder layer to yield a new intermediate representation of the words in sentence by attention. These are normalized and forwarded to an intermediate feed-forward network where a drop out is performed. This occurs similarly in all the 12 encoder layers and then a whole dropout is performed with a linear classifier function used as the final classification layer. The model consists of two classes for classification and hence the output layer has two output probabilities.

Bert-large-uncased

Bert-large-uncased is a model trained on lower-cased English text on 16 TPUs for 4 days. It consists of 24 bert layers (transformer encoder blocks) with 1024-hidden layers, 16 attention heads and 336 million parameters [1].

Bert-large-uncased model internal working is similar to the bert-base-uncased model with the only difference in number of encoder layers. It consists of a series of 24 encoder layers. Bert-large-uncased usually takes longer time to be fine-tuned and occupies quite vast memory due to its large architecture.

3.4.2 DistilBERT pre-defined models

DistilBERT is an architecture provided by transformers library and is a distilled version of BERT. Distilbert-base-uncased pre-trained model is chosen for this thesis experiments. The large version of Distilbert is not experimented, due to the not so great results obtained using bert-large.

Distilbert-base-uncased

This model is obtained from a checkpoint of bert-base-uncased model. It consists of 6 encoder layers with 768-hidden layers, 12 attention heads and 66 million parameters [1]. Distilbert-base-uncased model takes less computation time, but obtains nearly good results as bert. Hence, this model is considered for this thesis experiments due to its simplicity compared to the bert model.

3.4.3 BioBERT pre-defined models

BioBERT models are built by fine-tuning the BERT model using a huge corpora of Biomedical documents. Due to its focus on Biomedical information, it can be termed as a domain-specific language representation model. DMIS-lab [4] provides these pre-trained models that can be imported and further fine-tuned on task specific data. Two of the pre-trained models chosen for this thesis experiments are biobert-base-cased and biobert-large-cased.

Biobert-base-cased

Biobert-base-cased (BioBERT-Base v1.1) is a bert-base-uncased model fine-tuned on PubMed medical documents and readily provided by DMIS lab [1].

Biobert-large-cased

Biobert-large-cased (BioBERT-Large v1.1) is a bert-large-uncased model fine-tuned on PubMed medical documents and readily provided by DMIS lab [1].

The architectures of the pre-defined models discussed above can be seen in Table 3.1.

Transformer Libraries	Pre-trained model	Architecture
BertForSequenceClassification	Bert-base-uncased	12 transformer encoder layers , 768-hidden layers, 12 attention heads, 110 million parameters
	Bert-large-uncased	24 transformer encoder layers , 1024-hidden layers, 16 attention heads, 336 million parameters
AutoModelForSequenceClassification	Biobert-base-cased	12 transformer encoder layers , 768-hidden layers, 12 attention heads, 110 million parameters
	Biobert-large-cased	24 transformer encoder layers , 1024-hidden layers, 16 attention heads, 336 million parameters
DistilbertForSequenceClassification	Distilbert-base	6 transformer encoder layers , 768-hidden layers, 12 attention heads, 66 million parameters

Table 3.1: Bert pre-defined models architectures used in the thesis experiments

Further in this section, various systems based on BERT and BioBERT architectures, submitted in the BioASQ challenges previously are discussed and analyzed to suggest further research improvements [54].

3.4.4 Systems and Approaches

The system **unipi-quokka** proposed in the seventh edition of BioASQ used BERT, BioBERT models where each encoder is focused on an input while using it for the Stanford Question Answering Dataset (SQUAD) [81]. BioBERT model is employed and fine-tuning is performed to obtain better results. This system involved few pre and post processing, Augmentation and Evaluation techniques to determine the span of the answer in the snippets using ROUGE scores [90].

The system **Bio-AnswerFinder** proposed by UCSD in the eight edition of BioASQ involved a three processing phases namely; Question Processing phase, Document Processing phase and the Answering phase each allocated to specific tasks. This is followed by ranking the answer candidate sentences after question focus entity type filtering, using a weighted-Relaxed Word Mover's istance (wRWMD) [52] similarity for clustering purposes on the word/sentence embeddings in an optimized way. This is followed by re-ranking the sentences by fine-tuning BERT classifier to obtain the final summary. Re-ranking the sentences was also experimented using fine-tuned BioBERT classifier which showed no difference to using a fine-tuned BERT classifier [78].

The system **PA** proposed by collaborating Russian universities worked on showing the high performance of baseline approaches compared to other approaches in few cases. This system involved methods of Document and Snippet Retrieval, where documents were retrieved by identifying relevant document candidates using BM25 algorithm and re-ranked them using variations of BERT (trained to preform logistic regression) by fine-tuning with BioASQ documents to extract high scored documents. The relevant snippet spans are retrieved from these high ranked documents and ranked using statistical, neural approaches. The baseline statistical ranking approach extracts entities related to snippets and questions to compute the score of relevance. Other re-ranking methods like word2vec based on cosine similarity, BERT similarity, BERT relevance and document scores based on question-sentence pair relevance score and their position. This systems also experimented a abstractive summarization method based on BioMed-RoBERTa [36] to enhance the readability of the summaries generated, however this obtained lower scores. The snippet re-ranking exhibited some good ROUGE-SU4 F1 scores. It is observed that there is no incorporation of semantic indexing which can be included using biomedical ontologies to improve the performance in future work [47].

The system proposed by **DAIICT** university, India in the eight edition of BioASQ challenge has shown good ROUGE-SU4 F1 scores and high Recall scores. This system focused on implementing a query-graph based summarization techniques, including named-entity information due to their better performance in previous studies [71]. UMLS, an ontology knowledge source is used by this system to generate a graph of candidate entities from the question and semantically connected entities to it, followed by using similarity measures to determine the importance of entities. The baseline approaches used are based on the

standard graph based techniques TextRank [66] and LexRank [28]. Improving these, the UMLS based query specific graphs are generated to obtain importance of words and match sentences using cosine similarity. A method called Query Sentence Matching (QSM) is introduced to compare all the sentences to the query in order to obtain the top sentences, in combination with UMLS to obtain important words. The results obtained using QSM were shown to be better compared to the use of UMLS [91]. From these observations, it can be said that QSM can contribute in a great way to improve the summaries without the inclusion of UMLS. Other ontological knowledge resources can be explored in future work to obtain better summaries.

The system **NCU-IISR** proposed by National Central University (NCU) in the eight edition of BioASQ used BioBERT as a pre-trained language model and logistic Regression model. The logistic Regression model was inspired by the Macquarie University system [68] which followed the similar steps of splitting the input snippets into sentences and scoring them to obtain the top n sentences. Following this, BioBERT is used to replace the features used in this pre-trained model with word embeddings. The snippets after splitting are used to calculate the ROUGE-SU4 F1 score between the candidate sentences and question, to determine the positive and negative instance that are used as training set for logistic regression model. The BioBERT model is fine-tuned by providing both question and candidate sentence as input as the same time to obtain embeddings that are fine-tuned. The [CLS] token embeddings which is a representation of the relation between candidate sentence and question, is also used a feature to determine ROUGE-SU4 F1 scores in testing [54]. This is followed by a ReLU activation layer using the loss function Mean Squared Error (MSE). However, all the instances/questions were not used in the process of fine-tuning the model. This is shown to be considered as future work along with expanding the inputs from snippets to the full abstracts provided. Different activation and loss functions can also be experimented as future work [38].

The system named as **DMIS** proposed in the eight edition of BioASQ is based on the proven improvement in performance of QA when the learning relationships between the sentence pairs is induced. This system used BioBERT to transfer the knowledge of Natural Language Inference (NLI) to biomedical Question Answering, thus reducing the complexity of requiring domain specific information. This means the pre-trained language models are used for these tasks and are fine-tuned on a target task further. Adding to this, a Sequential transfer learning is also experimented to improve the performance of these tasks [45].

The system proposed by **Macquarie university** in the eight edition of BioASQ challenge used variants of BERT and BioBERT with the overall architecture similar to the seventh edition shown in the figure 3.2, with the regression setup using the training data labelled with the ROGUE-SU4 F1 scores of each candidate sentence. The objective function to be optimised is Mean Squared Error (MSE) and the classification setup is used. The labels of the top 5 sentences with highest ROUGE-SU4 F1 score is labelled as '1', while

the remaining are labelled as '0', with the objective function to be optimised as binary cross-entropy. The experiments using variants of BERT and BioBERT explained below and consists of the final layer as a classification layer. The sigmoid activation function and binary cross-entropy are the objective functions to be optimised [25].

1. **BERT untrained:** The embedding generator is replaced by BERT using pre-trained model provided by HuggingFace[2]. The word embeddings obtained are now affected by the context and the weights are not updated during training. The average of the word embeddings is considered as the embedding reductor of each sentence and candidate sentence (as recommended by HuggingFace).

2. **BERT trained:** The embedding generator and reductor are same as explained in BERT untrained above, except that the weights are updated i.e fine-tuned in the training stage.

3. **BERT LSTM:** The embedding generator is same as in BERT untrained but the reductor is a bidirectional LSTM chain as in Classification setup.

4. **BioBERT untrained and BioBERT LSTM:** The embedding generator and reductor are same as in BERT untrained and BERT LSTM respectively, except that the BERT model is trained using biomedical documents as provided by the developers of BioBERT [54].

Other approaches using Siamese variants are also experimented. Siamese network is generally used in applications that include comparisons between documents like determining the semantic similarity etc [89]. The general idea used by this system is to *'use the same processing module for each of the two documents by sharing the weights of encoding component that generates the embeddings of the documents [25]'*. The two variants used are:
1.) LSTM that implements a usual Siamese Network by incorporating the sharing of weights of embedding reductors of both candidate sentence and question to generate sentence embeddings using the exact same processing.
2.) Sentence Bert (SBERT) which uses BERT in a Siamese setup to compute the similarity between a candidate sentence and question.

These architectures are shown below in the figure 3.7, 3.8. The three setups experimented in the approaches using sBERT in Figure 3.8 are classification, regression, multitask (both included) with both the Classification and Regression setups using Classification labels [25].

Regression: In this setup, the cosine similarity between candidate sentence and question is calculated and Mean Square Error (MSE) is used as the objective function to be optimised.

[2] https://huggingface.co/

Classification: In this setup, the embeddings of the candidate sentence and question are concatenated with element-wise absolute difference between the candidate sentence and question with an added softmax layer and binary cross-entropy as the objective function to be optimised.

Multi-task: Both the regression and classification setups are jointly optimised in the training stage of this setup. Either of the regression or classification label is used in the prediction stage [25]

Figure 3.7: Base architecture used by the MQ system with the inclusion of using a Siamese Network [25]

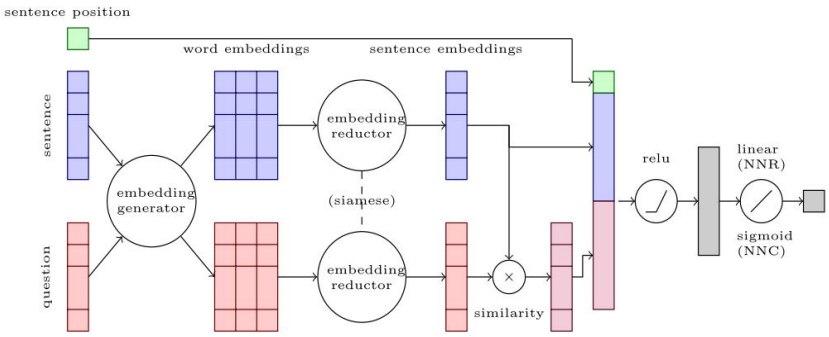
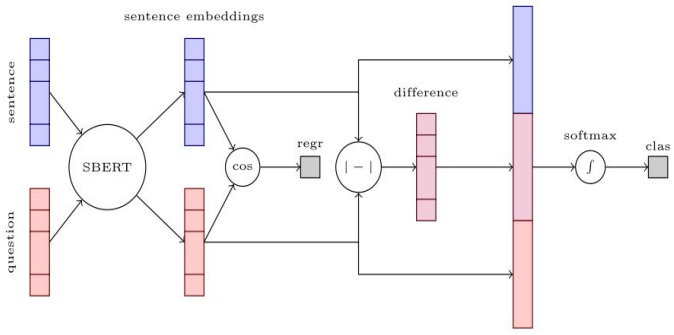


Figure 3.8: Architecture using sBERT in Siamese setup to generate sentence embeddings of question and candidate sentence [25]



It was observed that BERT produced the worst results and untrained BERT has no improvement compared to classification method. This led to the use of the BERT untrained only with LSTM approach, along with untrained BioBERT based variants experimented in the same way. Overall, the architecture using BERT or BioBERT to compute the word embeddings followed by LSTM-based sentence reductor obtained good results [25].

3.4.5 Analysis

Analyzing the systems above, it can be said that the pre-trained language models like BERT and Bio-BERT can enhance the performance of Query based extractive summarization techniques making these the state-of-the-art approaches. It can also be observed that inclusion of semantic knowledge and use of biomedical ontologies like UMLS can produce better summaries. Improvement in the summaries generated using architectures involving pre-trained models along with fine-tuning them on domain specific and task related documents are observed to produce better results. This can be due to the consideration of the context of words before and after the usage of word to produce embeddings.

Similar to the analysis in Deep Learning approaches, the change in similarity function has showed good improvement in the ROUGE-SU4 F1 scores [52] and hence these functions can further be investigated along with fine-tuning the pre-trained language models using domain-specific features. There is also a progress shown in the use of graph-based methods for extracting summaries, hence this can be considered as a starting point to explore more graph based architectures based on named entities and relationships [91].

It can also be observed that the system with simple BERT based architectures may produce better results than the complex BERT architectures in few scenarios [38] , which is taken as the focus of research work in this thesis. We also observed that the use of Transfer Learning produced better results for exact answers [45]. This is also considered in this thesis work to obtain better results.

3.5 Conclusion

Concluding this chapter, various approaches using Deep Learning, Reinforcement Learning used by the several systems to generate Query based Extractive Summaries in the BioASQ challenge are discussed. The systems based on BERT and BioBERT are discussed and analyzed, along with clearly stating the scope of research work it led to in this thesis.

Chapter 4

Methodology

In the previous chapters, we discussed Text summarization techniques, Query-based extractive summarization approaches used in the previous systems submitted in BioASQ along with working of BERT/BioBERT and the systems based on BERT/BioBERT in BioASQ. An analysis of these systems is also performed to understand the current state-of-the-art, leading to observing the scope of research work in this thesis.

Based on the research on BioASQ systems in the previous years, a few key observations are made to produce better summaries as below.

i. Classification approaches perform better than Regression approaches.

ii. Transfer Learning i.e using pre-trained language models obtained better results.

iii. BERT and Bio-BERT based model architectures in combination with other summarizing algorithms obtained state-of-the-art results.

These observations led to working on the research question "Can simple models built using BERT and BioBERT architectures obtain better results?". This can also be framed as "Is complexity of the model hindering the better performance of the model?". Thus, this thesis involves working with simple pre-trained BERT based models without including any external algorithms. However, it is to be observed that the model is not simple, since BERT by itself has a complex architecture.

Further, the models built to solve this research question are discussed and analyzed with a detailed explanation of data processing, model architectures, training and testing models along with performance measures and metrics.

4.1 Software and Libraries

For implementing the code in this thesis a specific set of software and libraries have been used. These need to be version checked to keep them suitable for the current

operating system used, which is Windows 10. The security installations needs to be checked as well, for safety of the code. Various software and libraries used for the successful implementation of the code in this thesis are discussed further in this section.

4.1.1 Programming setup and environment

A specific coding platform needs to be setup in order to build the code. Google Collab and Visual Studio Code (VSC) are used to write the code scripts and execute respectively. Visual Studio Code allows writing coding scripts in a easy fashion. However, due to limited resources of GPU on local computer, Google Collab with a 12GB NVIDIA Tesla K80 GPU is chosen to execute the code by importing from VSC [48]. The entire coding setup is done on a Windows 10 operating system.

4.1.2 Python

Python language is chosen to write the code in this thesis. Among the various reasons for choosing this language, the wide availability of libraries and frameworks that support the Deep Learning and Machine learning implementations stood as the major reason. Along with this, Python is a general purpose high level language with an easily understandable syntax, and hence can be justified to use for wide range of code implementations.

4.1.3 PyTorch

Torch is an open source Machine learning library that provides a range of algorithms required for Machine and Deep learning implementations. PyTorch is an open source Machine Learning library built using Python on the Torch library.

PyTorch utilizes tensors i.e. n-dimensional arrays as data structure. The important use of tensors is that they can be implemented on both GPU and CPU, making them efficient to use due to high speed executions. Another open-source library TensorFlow also uses tensors similarly, however, PyTorch is preferred in this thesis due to its capability of learning quickly and rapid model building capacity [108].

4.1.4 HuggingFace Transformers

HuggingFace Transformers[1] is a software built on PyTorch and TensorFlow libraries, that provides various pre-defined models trained on huge text corpus. PyTorch Transformers is a library built on PyTorch and provides a diverse range of pre-trained models and pre-trained weights that can be imported, and fine-tuned on the required data. This is known as Transfer learning and can be customized based on the specific task. Using transfer learning through pre-trained models has various advantages as discussed in Section 3.4 in Chapter 3 [2].

[1] https://huggingface.co/transformers/

4.1.5 Other Libraries

Along with the use of PyTorch-Transformers, various Python libraries like Numpy, Pandas, Matplotlib and Scikit-learn are also used.

Numpy

Numpy is a powerful library used to work with n-dimensional arrays, matrices and algebraic expressions. It is similar to Tensors with the major difference of no back-end availability of GPU for Numpy arrays [3].

Pandas

Pandas library plays a critical role in data manipulation and is capable of building in-memory 2D table objects known as data-frames which are widely used in Python Data science projects. It also provides the capability to converting data of different formats into the required format, thus assisting in easy data handling [107].

Matplotlib

Matplotlib library provides the data visualization and graphical plotting functions. It allows the use of static, dynamic and interactive data visualization techniques. These are valuable in statistical analysis of data, visualizing model performance by plotting the results obtained using performance metrics [6].

Scikit-Learn

Scikit-learn, also known as sklearn, provides a wide range of Machine Learning algorithms including Classification, Regression, K-neighbours etc. In this thesis, sklearn is used to calculate the chosen performance metrics, due to its capability of computing mathematical and graphical calculations easily through readily available functions.

4.1.6 Limitations

The selection of the software and libraries for this thesis as stated above has few limitations/drawbacks discussed below.

i. The use of Google Collab relies on a stable internet connection. It also has a 12-hour limitation of GPU utilization and has run-time disconnection when the computational facilities are not used for a specific amount of time. However, the availability of 12GB GPU overrides its disadvantages.

ii. National Computational infrastructure (NCI)[2] is a cloud facility that allocates GPU resources based on the requirements. However, the recent changes of the versions of

[2] https://nci.org.au/

python libraries, compatibility and the time constraints of this thesis led to a non usage of this cloud facility.

iii. The libraries and packages imported in this thesis need to be checked for their current versions and compatibilities to ensure error-free implementation of the codes as desired.

4.2 Data collection and Pre-processing

Data collection is one of the major steps in Machine or Deep learning projects. The data availability and unbiased data plays a key role in acquiring better model performance.

BioASQ provides data-sets each year and the dataset provided in 8b run in 2020 is chosen for the code building. For every run each year, 5 batches of test data are provided as well. These data-sets readily provided by the BioASQ organization are utilized in this thesis. The data provided is in JavaScript Object Notation (JSON) format and needs to be pre-processed before training the model. The data consists of a list of questions with multiple snippets i.e answer texts and additional details related to questions like question ID, type etc. The 'text' in snippets indicate the answer text of respective question as shown in Figure 4.1.

Figure 4.1: Example snippet of data in JSON format

4.2.1 Data pre-processing

Data pre-processing plays a key role in transforming the raw data into a required structured format for further use. This step is essential as, the data usually is in an unstructured format and in order to add meaning to the data, it needs to be pre-processed into a valid format. This involves few steps and are detailed below.

i. Firstly, all the libraries and data sets are to be imported successfully into the programming environment.

ii. The format of the dataset needs to be checked. The training and testing dataset are in JSON format, hence the data is converted into Comma-separated values (CSV) file or data-frame and saved for further use.

iii. The data is split into training, validation data and checked for any missing values. In this thesis, the data is split based on the question ID's randomly, as every question ID has multiple snippets or answer texts. The major requirement of splitting the data into validation/test data as the first step is that, the model should be given unseen data while validating, for better learning and validation of the model.

iv. Data cleaning is one of the important steps in data pre-processing performed to get rid of all the irrelevant data for the model. Since the data chosen is already in a standard format, this step is not essential. Along with these the punctuation and stop words need not be removed, as the tasks performed in this thesis require contextual understanding. The presence of punctuation and symbols can make the model learn better in terms of contextual and sentimental analysis where needed.

Tokenization

Tokenization is the process of splitting the text into small meaningful text. In this thesis experiments, the input sentences are split into small meaningful words known as tokens using 'BERT tokenizer'.

BERT tokenizer is built using Word-Piece algorithm to perform tokenization. Word Piece tokenization is a sub-word tokenization algorithm which adds the word into the list of vocabulary whose size is pre-defined, based on maximising the probability of training data until the maximum vocabulary size is reached. It also splits long words into frequently used smaller words. These tokens are further passed into the model in desired format (like embedding) for training the model. An example of tokenizing a sentence using this algorithm by BERT tokenizer can be seen in Figure 4.2 [27].

Figure 4.2: Tokenizing sentence using BERT tokenizer based on Word-piece algorithm

4.2.2 Training and Testing data

The dataset is usually split into three sub-sets; namely Training data, Validation data and Testing data. Since we are provided with testing data separately, the training data is split into training and validation at a regular 80:20 ratio, however it can be changed based on the project requirements.

Training data allows the model to learn the features of input data, while Validation data is used to validate the performance of the model on unseen data. Based on the results, the model with good performance on unseen data is selected for testing on test data.

4.2.3 Labels

Labels are known as ground truth and are vital in evaluating the model. These are the ground truth provided to the model, as to how it should predict the output by learning through given input.

In this thesis, the labels are predicted based on ROUGE-SU4 scores. ROUGE-SU4 is a metric used to measure the performance of the model discussed in detail in the section 4.5. The scores of the answer snippets for each question are calculated and the top 5 answer snippets with highest scores are labelled '1', while the remaining answer snippets are labelled '0'. Similarly, the labels are determined for every question answer snippets as shown in Figure 4.3.

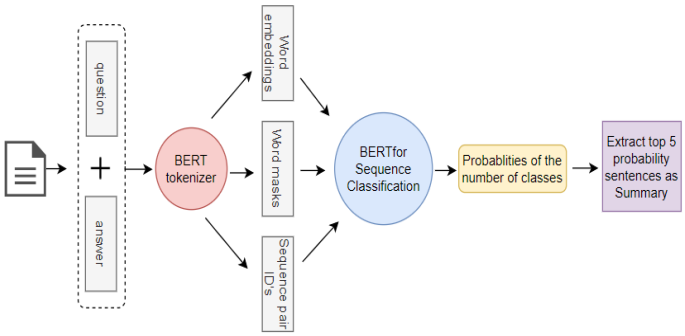
Figure 4.3: Example of data with labels marked based on the ROUGE-SU4 scores

qid	pubmedid	SU4	sentence text	SU4_labels	question
0	55031181e9bde6!	0.52892	The non-Mendelian inheritance (	1	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.52892	The non-Mendelian inheritance (	1	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.25117	The majority of the identified ge	1	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.25117	The majority of the identified ge	1	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.2	In this study, we review the iden	1	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.18015	In the etiology of Hirschsprung d	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.18015	In the etiology of Hirschsprung d	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.1375	Furthermore, mutations in the R	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.12406	RET, GDNF, EDNRB, EDN3, and S	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.09399	Hirschsprung disease (HSCR) is a	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.09399	Hirschsprung disease (HSCR) is a	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.09399	Hirschsprung disease (HSCR) is a	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.07826	The inheritance of Hirschsprung	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.07826	The inheritance of Hirschsprung	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.07782	BACKGROUND: RET is the major	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.06615	Therefore, HSCR has become a r	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.05584	Chromosomal and related Mend	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.05582	Differential contributions of rare	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.05405	Coding sequence mutations in e.	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.04292	For almost all of the identified H	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
0	55031181e9bde6!	0.03957	On the basis of a skewed sex-rat	0	Is Hirschsprung disease a mendelian or a multifactorial disorder?
1	55046d5ff8aee20	0.54918	the 7 known EGFR ligands (EGF, I	1	List signaling molecules (ligands) that interact with the receptor EGFR?
1	55046d5ff8aee20	0.4	EGFR ligands epidermal growth	1	List signaling molecules (ligands) that interact with the receptor EGFR?
1	55046d5ff8aee20	0.35114	mammalian EGFR ligands includi	1	List signaling molecules (ligands) that interact with the receptor EGFR?
1	55046d5ff8aee20	0.28449	the epidermal growth factor rec	1	List signaling molecules (ligands) that interact with the receptor EGFR?
1	55046d5ff8aee20	0.2653	Among EGFR ligands, heparin-bir	1	List signaling molecules (ligands) that interact with the receptor EGFR?
1	55046d5ff8aee20	0.23776	Among EGFR ligands, heparin-bir	0	List signaling molecules (ligands) that interact with the receptor EGFR?
1	55046d5ff8aee20	0.237	oluble amphiregulin (AR), transfc	0	List signaling molecules (ligands) that interact with the receptor EGFR?

4.3 Model architecture

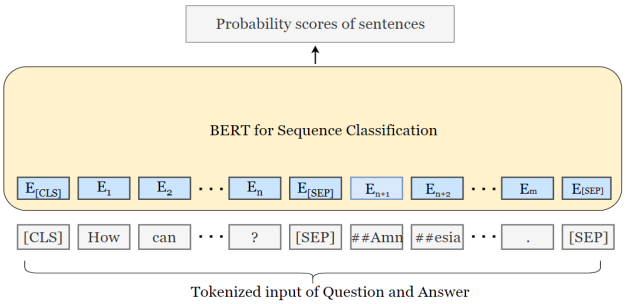
The models chosen for this thesis experiments are pre-trained models based on BERT in conjunction with the BertForSequenceClassification model provided by HuggingFace. The three models used for the experiments are BERT based, DistilBERT and BioBERT based. These model internal architecture and the working of layers are explained in detail in Section 3.4 in Chapter 3. The overall model architecture implemented in this thesis can be seen in Figure 4.4.

Figure 4.4: Architecture of the model implemented in the thesis



The input in the form of a unique string of question and the respective answer snippet are combined and sent to the bert tokenizer. The question and answer snippet are joined using special tokens '[CLS]' at the beginning and '[SEP]' in between the question, answer and at the end of text. This can be observed in Figure 4.5.

Figure 4.5: Simple architecture of the BertForSequenceClassification model



These inputs are sent to a BERT tokenizer where the sentences are broke down into smaller words i.e tokens. The tokens i.e words produced are processed to obtain word embeddings, mask embeddings, sequence ID embeddings and sent to the BertForSequenceClassification transformer model as explained in Section 3.4 in Chapter 3 in detail. These inputs are

processed through the neural network (which is trained), to obtain the probability scores of the sentences. These probabilities are further sorted to obtain the top five sentences as summary for a question.

The training and validation of the model to obtain the scores, through which summary is generated in testing phase is further discussed in this section.

4.4 Training and Testing the model

After building or importing the model architecture, the model is trained using training data followed by validating through a validation data-set. Once the model is trained, the model is tested using testing data to obtain the outputs as needed based on the task requirements. The method of training, validating and testing performed in this thesis experiments are discussed further in this section.

4.4.1 Training and Validating models

The training data is converted into the desired input format and processed to obtain embeddings as explained above. These embeddings are converted into tensors and are now used to train the model to learn the features to predict outputs. Some of the important parameters required while training the model are as shown in Table 4.1.

Parameter	Description	Value
Epochs	Indicates the number of times the models needs to run over the entire given dataset.	4
Batch	Indicates the number of divisions a dataset needs to be split into in one epoch.	32
Learning rate	Indicates the rate of change in the weights of the neural network based on the loss obtained.	$2e-5$
Number of warm-up steps	Indicates the number of steps until which the learning rate reaches a low value as '0', before the scheduling function starts.	100
Number of total steps	Indicates the total number of steps in the training phase of the model.	1000

Table 4.1: Parameters used for Training the model

The number of epochs chosen is '4'. However, the value chosen is usually bigger in number when the model needs to find the best weights while being trained from scratch. Since the model used is pre-trained and consists of best weights by default, 4 epochs are sufficient for fine-tuning tasks. The batch chosen is '32', which means 32 batches of data is sent in each epoch for training the model.

After every epoch, the loss is calculated using *'CrossEntropyLoss'*[3] function, comparing

[3] https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html

the predictions obtained by the model with the actual target labels. Logits/Predictions help in measuring the correctness of the model, while Loss indicates how bad are the predictions made by the model after training and is a key indicator for assessing the model learning. Based on the loss calculated, the optimizer and scheduler back propagate to update the weights after each epoch, based on the loss and learning rate, to obtain the best weights for the model.

Optimizer: Optimizer plays a key role in adjusting the weights of the neural network in the model by giving few parameters that indicate how the model needs to be learnt to improve the performance. The goal is to find the best weights that can minimize the loss. Optimizer has multiple parameters that are required when optimizing the weights during back-propagation of the network with the major ones being Weight decay and Learning rate. Weight decay a regularization method known as L2 regularization, applied to the weights of a neural network as a penalty to overcome the possibility of over-fitting. Over-fitting happens when the model memorizes the noise in the training data, leading to improper predictions [7].

Scheduler: Scheduler assist the optimizer by providing a schedule on how the learning rates need to be adjusted by providing a step value [5].

The chosen optimizer is *'AdamW optimizer'*, where the weight decay is fixed and is independent on the learning rate. A *'get_linear_schedule_with_warmup'* scheduler is chosen in this thesis where the learning rate is linearly increased after the warm-up phase. The number of warm-up steps are '100' among the '1000' total number of steps for training the model. The learning rate is '2e-5' as stated in Table 4.1. The parameters chosen in this thesis for the optimizer and scheduler are the best parameters retrieved through various external research experiments results.

Along with training in each epoch, the trained model is validated on validation data to check its performance on unseen data. In validation process, the validation data is sent to the model in the from of tensors in required format and the loss is calculated in each epoch to predict the performance of the model. The model with the best validation accuracy is saved along with the weights for future evaluation.

Saving models

The model with the best weights obtained is saved for further use. This is essential because the time and computational resources utilized to train the model are high. If we want to use the trained model on unseen test data at a later time, training the model is a waste consumption of resources and time. To overcome this, the saved models weights can be loaded in seconds/minutes for testing or reviewing the performance of the model.

4.4.2 Testing models

Once the model is trained and validated, the model is tested on testing data which is unseen data as well. The testing dataset is pre-processed and the inputs are provided to the model similar to the training phase, without showing the labels to the model. The learnt model predicts the scores of the sentences through the features of data learnt in training phase.

During this phase, the code can be structured based on what is required in the task. In this thesis, the top 5 sentences with maximum scores for each question are returned as summary. To implement this, a set of steps are followed as below.

1. Each time only the answer snippets of a question are sent separately to the model by joining question and each answer.

2. The logits/prediction probabilities obtained for each sentence are sorted using sort function to obtain the indices of the top 5 sentences with maximum score. If the number of sentences are less than five, all the sentences are returned as summary.

BioASQ Submission

BioASQ has a specific format to submit the results in the form of JSON and with few constraints on what part of the data needs to be submitted. Hence, the summaries obtained are structured in the format accepted by BioASQ for further performance evaluation on human generated summaries.

4.5 Model performance measures and metrics

To measure the performance of the model, specific evaluation metrics are chosen that can determine how the model is performing. The evaluation metrics chosen in this thesis experiments are Accuracy and F1 score.

Accuracy

Accuracy is the percentage measure of the correctness of the model on test data. It is the ratio of correct predictions to the total number of predictions by the model [87].

F1score

F1-Score is a statistical measure calculated as a Harmonic mean of Precision and recall. Precision is the ratio of correct positive predictions made by the model to the total number of positive predictions determined by the model. Recall is the ratio of correct positive predictions made by the model to the actual positives in the dataset given [87].

Note: A prediction is said to be positive if the predictions made correctly or incorrectly are same as the given ground truth/labels. If the predictions are not same as in ground truth/labels, it is said to be negative.

4.5.1 BioASQ ROUGE scores

Along with the metrics chosen, BioASQ provides ROUGE-2 and ROUGE-SU4 scores by comparing the summaries generated with human generated summaries. These metrics are also used in this thesis to assess the performance of the model.

ROUGE-2

ROUGE-2 is the measure of overlap of bi-grams between the model generated summary and human generated summary. Bi-grams are the adjacent pair of words/tokens in a sentence [56].

ROUGE-SU4

ROUGE-SU4 is the measure of overlap of bi-grams with a maximum skip distance as four. Skip distance is the distance that can be skipped between the next choice of bi-gram [56].

Among the metrics discussed above, F1-score and Rouge-SU4 are mainly focused while evaluating the performance of the model in this thesis.

Chapter 5

Results

In the previous chapter, the methodology and methods used in this thesis with their detailed explanation have been discussed. In this chapter, the results obtained by the models based on pre-trained BERT, BioBERT, DistilBERT architectures are discussed and analyzed to determine the best performing model.

5.1 BERT pre-trained models

BERT pre-trained models used in the thesis experiments are bert-base-uncased and bert-large-uncased. The accuracy and F1-score obtained by these models in the validation phase are shown in Table 5.1.

Model	Accuracy	F1-Score
Bert-base	0.7060	0.5963
Bert-large	0.7055	0.5958

Table 5.1: Accuracy and F1-score of BERT pre-trained models

It can be observed that bert-base model obtained a better accuracy and F1-score, compared to the bert-large model. Further, using the test data provided by BioASQ in 5 different batches, the summaries are extracted and submitted to BioASQ. The ROUGE-2 and ROUGE-SU4 scores obtained by these models in the five batches can be seen in Tables 5.2 and 5.3 respectively.

Model	ROUGE-2				
	Batch-1	Batch-2	Batch-3	Batch-4	Batch-5
Bert-base	0.6488	0.5843	0.5467	0.5121	0.6258
Bert-large	0.5884	0.5775	0.5044	0.4757	0.569

Table 5.2: ROUGE-2 scores of BERT pre-trained models in 5 batches

Model	ROUGE-SU4				
	Batch-1	Batch-2	Batch-3	Batch-4	Batch-5
Bert-base	0.6529	0.5927	0.5586	0.5223	0.62679
Bert-large	0.5986	0.5869	0.5216	0.4884	0.5737

Table 5.3: ROUGE-SU4 scores of BERT pre-trained models in 5 batches

It can be observed that bert-base model obtained better scores in all the batches when compared to bert-large. The difference in the scores is quite noticeable and hence can be said that, bert-base model produces better results when evaluated on human generated summaries.

5.2 DistilBERT pre-trained models

DistilBERT pre-trained models used in the thesis experiments are Distilbert-base-uncased. The accuracy and F1-score obtained by these models in the validation phase are shown in Table 5.4.

Model	Accuracy	F1-Score
Distilbert-base	0.7056	0.5961

Table 5.4: Accuracy and F1-score of DistilBERT pre-trained models

Using the test data provided by BioASQ in 5 different batches, the summaries are obtained and submitted to BioASQ. The ROUGE-2 and ROUGE-SU4 scores obtained by these models in the five batches can be seen in Tables 5.5 and 5.6 respectively.

Model	ROUGE-2				
	Batch-1	Batch-2	Batch-3	Batch-4	Batch-5
Distilbert-base	0.596	0.5442	0.5052	0.5054	0.5849

Table 5.5: ROUGE-2 scores of DistilBERT pre-trained models in 5 batches

Model	ROUGE-SU4				
	Batch-1	Batch-2	Batch-3	Batch-4	Batch-5
Distilbert-base	0.6076	0.5508	0.5223	0.5139	0.591

Table 5.6: ROUGE-SU4 scores of DistilBERT pre-trained models in 5 batches

5.3 BioBERT pre-trained models

BioBERT pre-trained models used in the thesis experiments are biobert-base-uncased and biobert-large-uncased. The accuracy and F1-score obtained by these models in the validation phase are shown in Table 5.7.

Model	Accuracy	F1-Score
Biobert-base	0.7057	0.5962
Biobert-large	0.7055	0.5960

Table 5.7: Accuracy and F1-score of BioBERT pre-trained models

It can be observed from the table above that, biobert-base model obtained better accuracy and F1-score compared to the biobert-large model. However, the difference is quite minimal. Further, using the test data provided by BioASQ in 5 different batches, the summaries are extracted and submitted to BioASQ. The ROUGE-2 and ROUGE-SU4 scores obtained by these models in the five batches can be seen in Tables 5.8 and 5.9 respectively.

Model	ROUGE-2				
	Batch-1	Batch-2	Batch-3	Batch-4	Batch-5
Biobert-base	0.6221	0.5657	0.4995	0.4841	0.5936
Biobert-large	0.5838	0.5532	0.5122	0.5094	0.5771

Table 5.8: ROUGE-2 scores of BioBERT pre-trained models in 5 batches

Model	ROUGE-SU4				
	Batch-1	Batch-2	Batch-3	Batch-4	Batch-5
Biobert-base	0.6261	0.5727	0.5165	0.4954	0.596
Biobert-large	0.5896	0.5589	0.5306	0.5226	0.5816

Table 5.9: ROUGE-SU4 scores of BioBERT pre-trained models in 5 batches

It can be observed that biobert-base model obtained better scores in three batches when compared to biobert-large, while it is nearly close to large model in remaining two batches. However, since the difference in score is quite noticeable it can be said that biobert-base model produce better results when evaluated on human generated summaries.

5.4 Discussion

Based on the results obtained, a mean and standard deviation of the ROUGE-2 and ROUGE-SU4 scores obtained by the models in 5 batches are calculated. These values can be observed in Table 5.10.

Model	Mean ± StDev (ROUGE 2)	Mean ± StDev (ROUGE SU4)
Bert-base	0.5835 ± 0.0559	0.5907 ± 0.0522
Bert-large	0.5430 ± 0.0499	0.5538 ± 0.0469
Biobert-base	0.5530 ± 0.0596	0.5613 ± 0.0545
Biobert-large	0.5471 ± 0.0351	0.5567 ± 0.0298
Distilbert-base	0.5471 ± 0.0428	0.5571 ± 0.0413

Table 5.10: Mean ± Standard Deviation of the ROUGE-2 and ROUGE-SU4 scores obtained by models in 5 batches

It can be observed that, the mean ± standard deviation of the bert-base model is higher when compared to the other models. This is followed by the biobert-base model with next highest scores. The graphical representation of the scores can be seen in Error bar graph Figures 5.1 and 5.2.

Figure 5.1: Error bar graph for ROUGE-2 scores of the models

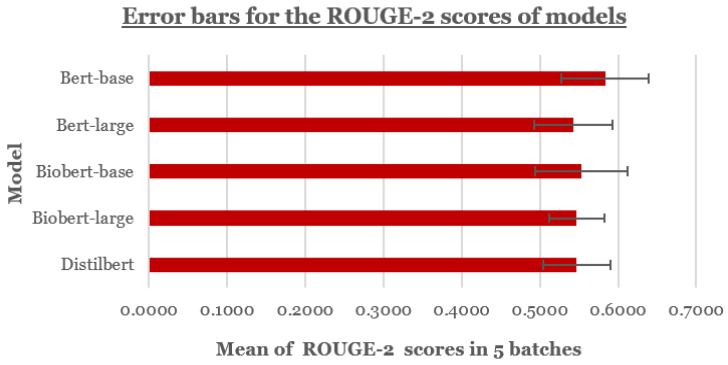
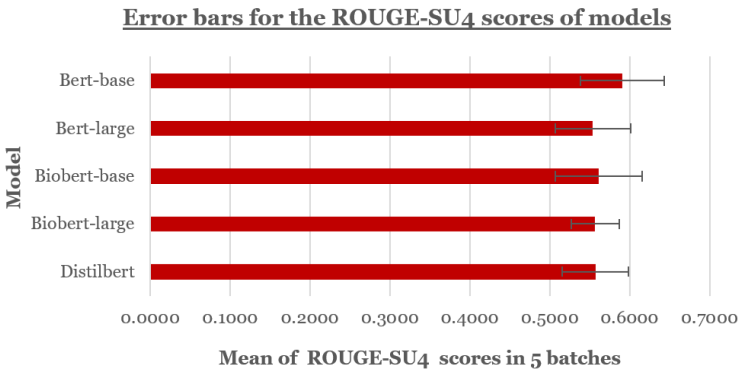


Figure 5.2: Error bar graph for ROUGE-SU4 scores of the models



Along with this, BioASQ provides a leaderboard which displays the system scores comparing with other submitted systems. The system/model built in this thesis stood as the top system in three batches and among the top 3 in the other two batches. These are shown in Figure 5.3 below.

Figure 5.3: Leader-board of BioASQ submissions

System Name	Rouge-2	Rouge-SU4
Current Submission	0.6488	0.6529
MQ-3	0.5789	0.5834
MQ-2	0.5671	0.5732

System Name	Rouge-2	Rouge-SU4
Current Submission	0.5843	0.5927
MQ-2	0.5244	0.5339

System Name	Rouge-2	Rouge-SU4
Current Submission	0.5467	0.5586
MQ-2	0.5481	0.5580
MQ-3	0.5394	0.5491
MQ-1	0.5222	0.5336

System Name	Rouge-2	Rouge-SU4
pa-base	0.5291	0.5321
MQ-3	0.5162	0.5220
Current Submission	0.5121	0.5223

System Name	Rouge-2	Rouge-SU4
system of teamdaiict	0.6473	0.6445
DAIICT_QSM	0.6428	0.6392
Current Submission	0.6258	0.6267
DAIICT_lex	0.6257	0.6239

Thus, in this way the results of the thesis experiments can be evaluated. To assess the performance of the model on testing data Accuracy, F1-score metrics are used. ROUGE scores are utilized to evaluate the summaries generated, by comparing with human generated summaries. Based on these observations using various metrics, it can be said that bert-base model stood out with best performance compared with other systems.

Chapter 6

Conclusion and Future work

In the previous chapter the results of the thesis experiments are discussed along with a detailed analysis of the metrics chosen for evaluating the model performance in this thesis.

Based on the results, it can be said that BERT models when used in a simple way can produce some of the best results, when compared to the current best systems in BioASQ. This can be termed as an interesting finding and shows the effectiveness of BERT. BERT has a complex architecture with multiple layers with every layer consisting of high functionalities. This concept is taken as the lead to the research of working with BERT architectures alone. This is proved by the better performance results obtained by the Bert-base and Biobert-base models when compared to the large architectures.

Fine-tuning Hyper parameters

It can also be observed that the use of pre-defined models reduced the need of vast computational resources to train the model. However, fine-tuning the hyper-parameters of the model do require minimal availability of computational resources leading us to the future work that can be done on this research output.

The parameters used are the best parameters provided by various researchers for the BERT fine-tuning tasks. The hyper-parameters can be changed and the model can be fine-tuned to obtain better results using Biobert-base and Biobert-large models. This hyper-parameter fine-tuning couldn't be performed due to limited computational resources on Google Collab, and the version change in the NCI[1] cloud computational resources platform. In the limited time of 13 weeks, achieving this research output was challenging and hence fine-tuning of the hyper-parameters can further be worked in future for achieving potentially better results.

[1] https://nci.org.au/

Considering Question Type

Along with this, currently the summary produced is a 5-sentence summary which is irrespective of the type of the question. The data consists of questions which require single line answers and including this feature can improve the scores when evaluating with the human generated summaries.

Regression and abstractive Summarization Techniques

Adding to this, some of the algorithms that were not used in this thesis are Regression and Abstractive summarizing techniques. The ways of combining the state-of-the-art BERT/BioBERT architectures with Regression or Abstractive Summarizing techniques might generate better summaries. This is one such idea that can be carried on in future.

Finally, concluding this thesis report it can be stated that simple BERT based architectures produced some of the best results, when evaluated on Human generated summaries. These findings can be considered for further experiments on ideas discussed above in order to improve the quality of the summaries generated in the future.

Appendix A

Abbreviations

HLT	Human Language Technology
IR	Information Retrieval
TC	Text Classification
TS	Text Summarisation
QA	Question Answering
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
NLG	Natural Language Generation
BERT	Bi-directional Bidirectional Encoder Representations from Transformers
BioBERT	Bi-directional Bidirectional Encoder Representations from Transformers for Biomedical Text Mining
ML	Machine Learning
DL	Deep Learning
RL	Reinforcement Learning
SVR	Support Vector Regression
SCU	Summary Content Units
PPO	Proximal Policy Optimisation
MLM	Masked Language Model
wRWMD	weighted-Relaxed Word Mover's Distance
SQUAD	Stanford Question Answering Dataset
QSM	Query Sentence Matching
MSE	Mean Squared Error
NLI	Natural Language Inference
LSTM	Long Short Term Memory
SVD	Singular Value Decomposition
MMR	Maximal Marginal Relevance
POS	Parts Of Speech
NER	Named Entity Recognition
MER	Medical Entity Recognition
PGC	Pointer Generator Coverage
LETOR	LEarning TO Rank

CNN	Convolutional Neural network
RELU	Rectifier Linear Units
RNN	Recurrent Neural Networks
MLM	Masked Language Model
VSC	Visual Studio Code
JSON	JavaScript Object Notation
NCI	National Computational Infrastructure
CSV	Comma Separated Values
GPU	Graphics Processing Unit
CPU	Central Processing Unit

Appendix B

Code

The code built for the thesis experiments can be accessed through the GitHub link Summarization_of_Biomedical_Evidence_Sahithi_Kodali_GitHub

Bibliography

[1] "HuggingFace—Predefined models," 2020, [Online; accessed 20-June-2021]. [Online]. Available: https://huggingface.co/transformers/pretrained_models.html

[2] "HuggingFace—Transformers," 2020, [Online; accessed 20-June-2021]. [Online]. Available: https://huggingface.co/transformers/index.html

[3] "Pandas vs Numpy," 2020, [Online; accessed 20-June-2021]. [Online]. Available: https://www.javatpoint.com/pandas-vs-numpy

[4] "BioBERT," 2021, [Online; accessed 21-June-2021]. [Online]. Available: https://github.com/dmis-lab/biobert

[5] "Dive into deep Learning," 2021, [Online; accessed 27-June-2021]. [Online]. Available: https://d2l.ai/chapter_optimization/lr-scheduler.html

[6] "Matplotlib," 2021, [Online; accessed 21-June-2021]. [Online]. Available: https://pypi.org/project/matplotlib/

[7] "Weight-decay explained," 2021, [Online; accessed 27-June-2021]. [Online]. Available: https://paperswithcode.com/method/weight-decay

[8] J. Alammar, "The Illustrated Transformer," 2018. [Online]. Available: https://jalammar.github.io/illustrated-transformer/

[9] E. Amigó, J. Gonzalo, A. Penas, and F. Verdejo, "QARLA: a framework for the evaluation of text summarization systems," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2005, pp. 280–289.

[10] C. Aone, M. E. Okurowski, and J. Gorlinsky, "Trainable, scalable summarization using robust NLP and machine learning," in *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, 1998, pp. 62–66.

[11] R. Barzilay and M. Lapata, "Modeling local coherence: An entity-based approach," *Computational Linguistics*, vol. 34, no. 1, pp. 1–34, 2008.

[12] R. Barzilay and K. R. McKeown, "Sentence fusion for multidocument news summarization," *Computational Linguistics*, vol. 31, no. 3, pp. 297–328, 2005.

[13] A. Bhandwaldar and W. Zadrozny, "UNCC QA: biomedical question answering system," in *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*, 2018, pp. 66–71.

[14] R. Brandow, K. Mitze, and L. F. Rau, "Automatic condensation of electronic publications by sentence selection," *Information Processing & Management*, vol. 31, no. 5, pp. 675 – 685, 1995, summarizing Text. [Online]. Available: http://www.sciencedirect.com/science/article/pii/030645739500052I

[15] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 89–96.

[16] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon, "Adapting ranking SVM to document retrieval," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006, pp. 186–193.

[17] J. Carbonell and J. Goldstein, "The use of MMR, diversity-based re-ranking for reordering documents and producing summaries," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 1998, pp. 335–336.

[18] G. Carenini and J. C. K. Cheung, "Extractive vs. NLG-based abstractive summarization of evaluative text: The effect of corpus controversiality," in *Proceedings of the Fifth International Natural Language Generation Conference*, 2008, pp. 33–41.

[19] K. Chandu, A. Naik, A. Chandrasekar, Z. Yang, N. Gupta, and E. Nyberg, "Tackling biomedical text summarization: Oaqa at bioASQ 5b," in *BioNLP 2017*, 2017, pp. 58–66.

[20] F.-D. Cioloboc, "Why use a pre-trained model rather than creating your own?" Jan. 2019. [Online]. Available: https://medium.com/udacity-pytorch-challengers/why-use-a-pre-trained-model-rather-than-creating-your-own-d0e3a17e202f

[21] J. Conroy and H. T. Dang, "Mind the gap: Dangers of divorcing evaluations of summary content from linguistic quality," in *Proceedings of the 22nd international conference on computational linguistics (Coling 2008)*, 2008, pp. 145–152.

[22] J. M. Conroy and D. P. O'leary, "Text summarization via Hidden Markov Models," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 406–407.

[23] D. Demner-Fushman and J. Lin, "Answer extraction, semantic clustering, and extractive summarization for clinical question answering," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 2006, pp. 841–848.

[24] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[25] Diego Molla and Christopher Jones and Vincent Nguyen, "Query Focused Multi-document Summarisation of Biomedical Texts," 2020.

[26] H. P. Edmundson, "New methods in automatic extracting," *Journal of the ACM (JACM)*, vol. 16, no. 2, pp. 264–285, 1969.

[27] Eram Munawwar, "A comprehensive guide to subword tokenisers," Dec. 2020. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-subword-tokenisers-4bbd3bad9a7c

[28] G. Erkan and D. R. Radev, "LexRank: Graph-based lexical centrality as salience in text summarization," *Journal of artificial intelligence research*, vol. 22, pp. 457–479, 2004.

[29] K. Filippova, E. Alfonseca, C. A. Colmenares, Ł. Kaiser, and O. Vinyals, "Sentence compression by deletion with LSTMs," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 360–368.

[30] M. Fiszman, T. C. Rindflesch, and H. Kilicoglu, "Abstraction summarization for managing the biomedical research literature," in *Proceedings of the computational lexical semantics workshop at HLT-NAACL 2004*, 2004, pp. 76–83.

[31] M. Fuentes, E. González, D. Ferrés, and H. RODRiguez, "Qasum-talp at duc 2005 automatically evaluated with a pyramid based metric," in *HLT-EMNLP Workshop (DUC 2005), Vancouver, Canada*, 2005.

[32] M. Fuentes Fort, E. Alfonseca, and H. Rodríguez Hontoria, "Support vector machines for query-focused summarization trained and evaluated on pyramid data," 2007.

[33] D. Galanis, G. Lampouras, and I. Androutsopoulos, "Extractive multi-document summarization with integer linear programming and support vector regression," in *Proceedings of COLING 2012*, 2012, pp. 911–926.

[34] Giannakopoulos, George and Karkaletsis, Vangelis and Vouros, George and Stamatopoulos, Panagiotis, "Summarization system evaluation revisited: N-gram graphs," *ACM Transactions on Speech and Language Processing (TSLP)*, vol. 5, no. 3, pp. 1–39, 2008.

[35] S. Gupta and S. Gupta, "Abstractive summarization: An overview of the state of the art," *Expert Systems with Applications*, vol. 121, pp. 49–65, 2019.

[36] S. Gururangan, A. Marasović, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey, and N. A. Smith, "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks," *arXiv preprint arXiv:2004.10964*, 2020.

[37] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation Learning on Graphs: Methods and Applications," *CoRR*, vol. abs/1709.05584, 2017. [Online]. Available: http://arxiv.org/abs/1709.05584

[38] J.-C. Han and R. T.-H. Tsai, "NCU-IISR: Using a Pre-trained Language Model and Logistic Regression Model for BioASQ Task 8b Phase B."

[39] L. Hasler, "From extracts to abstracts: human summary production operations for computer-aided summarisation," in *Proceedings of the RANLP 2007 workshop on computer-aided language processing (CALP)*, 2007, pp. 11–18.

[40] Hasler, Laura, "Centering Theory for Evaluation of Coherence in Computer-Aided Summaries," in *LREC*, 2008.

[41] E. H. Hovy, C.-Y. Lin, L. Zhou, and J. Fukumoto, "Automated Summarization Evaluation with Basic Elements." in *LREC*, vol. 6. Citeseer, 2006, pp. 899–902.

[42] E. Hovy and C. Y. Lin, "Automated multilingual text summarization and its evaluation," in *Technical Report. Information Sciences Institute, University of Southern California*, 1999.

[43] K. S. Jones *et al.*, "Automatic summarizing: factors and directions," *Advances in automatic text summarization*, pp. 1–12, 1999.

[44] K. S. Jones and J. R. Galliers, *Evaluating natural language processing systems: An analysis and review*. Springer Science & Business Media, 1995, vol. 1083.

[45] J. Kang, "Transferability of Natural Language Inference to Biomedical Question Answering," *arXiv preprint arXiv:2007.00217*, 2020.

[46] R. Katragadda, "GEMS: generative modeling for evaluation of summaries," in *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2010, pp. 724–735.

[47] A. Kazaryan, U. Sazanovich, and V. Belyaev, "Transformer-Based Open Domain Biomedical Question Answering at BioASQ8 Challenge."

[48] A. Kazemnejad, "How to do Deep Learning research with absolutely no GPUs - Part 2," 2019. [Online]. Available: https://kazemnejad.com/blog/how_to_do_deep_learning_research_with_absolutely_no_gpus_part_2/

[49] S. J. Ker and J. N. Chen, "A Text Categorization Based on a Summarization Extraction," in *ACL-2000 Workshop on Recent Advances in Natural Language Processing and Information Retrieval*, 2000, pp. 79–83.

[50] A. N. Kumar, H. Kesavamoorthy, M. Das, P. Kalwad, K. Chandu, T. Mitamura, and E. Nyberg, "Ontology-based retrieval & neural approaches for BioASQ ideal answer generation," in *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*, 2018, pp. 79–89.

[51] J. Kupiec, J. Pedersen, and F. Chen, "A trainable document summarizer," in *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, 1995, pp. 68–73.

[52] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *International conference on machine learning*, 2015, pp. 957–966.

[53] Lalmas, Mounia and van Rijsbergen, Keith, "A logical model of information retrieval based on situation theory," in *14th Information Retrieval Colloquium*. Springer, 1993, pp. 1–13.

[54] Lee, Jinhyuk and Yoon, Wonjin and Kim, Sungdong and Kim, Donghyeon and Kim, Sunkyu and So, Chan Ho and Kang, Jaewoo, "BioBERT: a pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, 09 2019. [Online]. Available: https://doi.org/10.1093/bioinformatics/btz682

[55] S. Li, Y. Ouyang, W. Wang, and B. Sun, "Multi-document summarization using support vector regression," in *Proceedings of DUC*, 2007.

[56] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, 2004, pp. 74–81.

[57] E. Lloret, *Text summarisation based on human language technologies and its applications*. Universidad de Alicante, 2011.

[58] E. Lloret, L. Plaza, and A. Aker, "The challenging task of summary evaluation: an overview," *Language Resources and Evaluation*, vol. 52, no. 1, pp. 101–148, 2018.

[59] Lloret, Elena and Palomar, Manuel, "Text summarisation in progress: a literature review," *Artificial Intelligence Review*, vol. 37, no. 1, pp. 1–41, 2012.

[60] P. Malakasiotis, E. Archontakis, I. Androutsopoulos, D. Galanis, and H. Papageorgiou, "Biomedical question-focused multi-document summarization: ILSP and AUEB at BioASQ3," 01 2015.

[61] I. Mani and M. Maybury, "Introduction, advances in automatic text summarization," *MIT Press, Cambridge*, 1999.

[62] I. Mani, *Automatic summarization*. John Benjamins Publishing, 2001, vol. 3.

[63] W. C. Mann and S. A. Thompson, "Rhetorical structure theory: Toward a functional theory of text organization," *Text*, vol. 8, no. 3, pp. 243–281, 1988.

[64] C. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008. [Online]. Available: https://books.google.com.au/books?id=t1PoSh4uwVcC

[65] V. McCargar, "Statistical approaches to automatic text summarization," *Bulletin of the american society for information science and technology*, vol. 30, no. 4, pp. 21–25, 2004.

[66] R. Mihalcea and P. Tarau, "TextRank: Bringing order into text," in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004, pp. 404–411.

[67] J.-L. Minel, S. Nugier, and G. Piat, "How to appreciate the quality of automatic text summarization? Examples of FAN and MLUCE protocols and their results on SERAPHIN," in *Intelligent Scalable Text Summarization*, 1997.

[68] D. Mollá, "Macquarie University at BioASQ 5b – query-based summarisation techniques for selecting the ideal answers," in *BioNLP 2017*. Vancouver, Canada,: Association for Computational Linguistics, Aug. 2017, pp. 67–75. [Online]. Available: https://www.aclweb.org/anthology/W17-2308

[69] D. Molla Aliod and C. Jones, *Classification Betters Regression in Query-Based Multi-document Summarisation Techniques for Question Answering: Macquarie University at BioASQ7b*, 03 2020, pp. 624–635.

[70] Mollá, Diego, "Macquarie University at BioASQ 6b: Deep learning and deep reinforcement learning for query-based summarisation," in *Proceedings of the 6th BioASQ Workshop A challenge on large-scale biomedical semantic indexing and question answering*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 22–29. [Online]. Available: https://www.aclweb.org/anthology/W18-5303

[71] M. Moradi and N. Ghadiri, "Different approaches for identifying important concepts in probabilistic biomedical text summarization," *Artificial intelligence in medicine*, vol. 84, pp. 101–116, 2018.

[72] T. Mori, M. Nozawa, and Y. Asada, "Multi-answer-focused multi-document summarization using a question- answering engine," *COLING '04: proceedings of the 20th international conference on computational linguistics*, p. 439–445, 2004.

[73] Mori, Tatsunori and Nozawa, Masanori and Asada, Yoshiaki, "Multi-answer-focused multi-document summarization using a question-answering engine," *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 4, no. 3, pp. 305–320, 2005.

[74] A. Nenkova, "Summarization evaluation for text and speech: issues and approaches," in *Ninth International Conference on Spoken Language Processing*, 2006.

[75] Nenkova, Ani, "Summarization evaluation for text and speech: issues and approaches," in *Ninth International Conference on Spoken Language Processing*, 2006.

[76] M. Okumura and T. Honda, "Word sense disambiguation and text segmentation based on lexical cohesion," in *COLING 1994 Volume 2: The 15th International Conference on Computational linguistics*, 1994.

[77] K. Owczarzak, "DEPEVAL (summ): dependency-based evaluation for automatic summaries," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009, pp. 190–198.

[78] I. B. Ozyurt, A. Bandrowski, and J. S. Grethe, "Bio-AnswerFinder: a system to find answers to questions from biomedical texts," *Database*, vol. 2020, 2020.

[79] R. J. Passonneau, "Formal and functional assessment of the pyramid method for summary content evaluation," *Natural Language Engineering*, vol. 16, no. 2, p. 107, 2010.

[80] E. Pitler and A. Nenkova, "Revisiting readability: A unified framework for predicting text quality," in *Proceedings of the 2008 conference on empirical methods in natural language processing*, 2008, pp. 186–195.

[81] Pranav Rajpurkar and Robin Jia and Percy Liang, "Know What You Don't Know: Unanswerable Questions for SQUAD," *CoRR*, vol. abs/1806.03822, 2018. [Online]. Available: http://arxiv.org/abs/1806.03822

[82] T. Qin, T.-Y. Liu, J. Xu, and H. Li, "LETOR: A benchmark collection for research on learning to rank for information retrieval," *Information Retrieval*, vol. 13, no. 4, pp. 346–374, 2010.

[83] D. Radev and W. Fan, "Automatic summarization of search engine hit lists," in *ACL-2000 Workshop on Recent Advances in Natural Language Processing and Information Retrieval*, 2000, pp. 99–109.

[84] D. R. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization," *Computational linguistics*, vol. 28, no. 4, pp. 399–408, 2002.

[85] D. R. Radev and K. R. McKeown, "Generating natural language summaries from multiple on-line sources," *Computational Linguistics*, vol. 24, no. 3, pp. 469–500, 1998. [Online]. Available: https://www.aclweb.org/anthology/J98-3005

[86] Radev, Dragomir R and Tam, Daniel, "Summarization evaluation using relative utility," in *Proceedings of the twelfth international conference on Information and knowledge management*, 2003, pp. 508–511.

[87] R. Rajpurohit, "Model Evaluation," May 2018. [Online]. Available: https://medium.com/@rakeshrajpurohit/model-evaluation-8e432c7d8a84

[88] L. Reeve, H. Han, and A. D. Brooks, "BioChain: lexical chaining methods for biomedical text summarization," in *Proceedings of the 2006 ACM symposium on Applied computing*, 2006, pp. 180–184.

[89] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. [Online]. Available: https://www.aclweb.org/anthology/D19-1410

[90] M. Resta, D. Arioli, A. Fagnani, and G. Attardi, "Transformer models for question answering at bioasq 2019," in *Machine Learning and Knowledge Discovery in Databases*, P. Cellier and K. Driessens, Eds. Cham: Springer International Publishing, 2020, pp. 711–726.

[91] J. Sankhavara and P. Majumder, "Query-focused biomedical text summarization in BioASQ 8B."

[92] F. Schilder and R. Kondadadi, "FastSum: fast and accurate query-based multi-document summarization," in *Proceedings of ACL-08: HLT, short papers*, 2008, pp. 205–208.

[93] P. L. Schuyler, W. T. Hole, M. S. Tuttle, and D. D. Sherertz, "The UMLS Metathesaurus: representing different views of biomedical concepts." *Bulletin of the Medical Library Association*, vol. 81, no. 2, p. 217, 1993.

[94] F. Sebastiani, "Machine learning in automated text categorization," *ACM Comput. Surv.*, vol. 34, no. 1, p. 1–47, Mar. 2002. [Online]. Available: https://doi.org/10.1145/505282.505283

[95] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," *arXiv preprint arXiv:1704.04368*, 2017.

[96] Z. Shi, G. Melli, Y. Wang, Y. Liu, B. Gu, M. M. Kashani, A. Sarkar, and F. Popowich, "Question answering summarization of multiple biomedical documents," in *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 2007, pp. 284–295.

[97] M. Q. Stearns, C. Price, K. A. Spackman, and A. Y. Wang, "SNOMED clinical terms: overview of the development process and project status." in *Proceedings of the AMIA Symposium*. American Medical Informatics Association, 2001, p. 662.

[98] K. Svore, L. Vanderwende, and C. Burges, "Enhancing single-document summarization by combining RankNet and third-party sources," in *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 2007, pp. 448–457.

[99] S. Teufel and H. V. Halteren, "Evaluating information content by factoid analysis: human annotation and stability," 2004.

[100] P. Thompson, "Advances in open domain question answering, edited by tomek strzalkowski and sanda harabagiu," *Computational Linguistics*, vol. 33, no. 4, pp. 597–599, 2007. [Online]. Available: https://www.aclweb.org/anthology/J07-4007

[101] J.-M. Torres-Moreno, P.-L. St-Onge, M. Gagnon, M. El-Beze, and P. Bellot, "Automatic summarization system coupled with a question-answering system (qaas)," *arXiv preprint arXiv:0905.2990*, 2009.

[102] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos, Y. Almirantis, J. Pavlopoulos, N. Baskiotis, P. Gallinari, T. Artires, A. C. Ngomo, N. Heino, E. Gaussier, L. Barrio-Alvers, M. Schroeder, I. Androutsopoulos, and G. Paliouras, "An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition," *BMC Bioinformatics*, vol. 16, p. 138, Apr 2015.

[103] R. Vadlapudi and R. Katragadda, "On automated evaluation of readability of summaries: Capturing grammaticality, focus, structure and coherence," in *Proceedings of the NAACL HLT 2010 student research workshop*, 2010, pp. 7–12.

[104] Vadlapudi, Ravikiran and Katragadda, Rahul, "Quantitative evaluation of grammaticality of summaries," in *International Conference on Intelligent Text Processing and Computational Linguistics*. Springer, 2010, pp. 736–747.

[105] G. Varile, R. Cole, R. Cole, A. Zampolli, J. Mariani, H. Uszkoreit, A. Zaenen, S. Bird, B. Boguraev, M. Kay *et al.*, *Survey of the State of the Art in Human Language Technology*, ser. Linguistica computazionale. Cambridge University Press, 1997. [Online]. Available: https://books.google.com.au/books?id=WlHD141lKw4C

[106] Wang, Chan and Long, Lixia and Li, Lei, "HowNet based evaluation for Chinese text summarization," in *2008 International Conference on Natural Language Processing and Knowledge Engineering*. IEEE, 2008, pp. 1–6.

[107] Wikipedia contributors, "Pandas (software) — Wikipedia, The Free Encyclopedia," 2021, [Online; accessed 21-June-2021]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Pandas_(software)&oldid=1028883828

[108] Wikipedia contributors, "PyTorch — Wikipedia, The Free Encyclopedia," 2021, [Online; accessed 20-June-2021]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=PyTorch&oldid=1028827477

[109] R. J. Williams, "Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning," *Mach. Learn.*, vol. 8, no. 3–4, p. 229–256, May 1992. [Online]. Available: https://doi.org/10.1007/BF00992696

[110] K.-F. Wong, M. Wu, and W. Li, "Extractive summarization using supervised and semi-supervised learning," in *Proceedings of the 22nd international conference on computational linguistics (Coling 2008)*, 2008, pp. 985–992.

[111] J. Yu, E. Reiter, J. Hunter, and C. Mellish, "Choosing the content of textual summaries of large time-series data sets," *Natural Language Engineering*, vol. 13, no. 1, p. 25, 2007.

[112] L. Yu, K. M. Hermann, P. Blunsom, and S. Pulman, "Deep Learning for Answer Sentence Selection," *CoRR*, vol. abs/1412.1632, p. 9, 2014. [Online]. Available: http://arxiv.org/abs/1412.1632

[113] D. Zajic, B. J. Dorr, J. Lin, and R. Schwartz, "Multi-candidate reduction: Sentence compression as a tool for document summarization tasks," *Information Processing & Management*, vol. 43, no. 6, pp. 1549–1570, 2007.

[114] L. Zhou, C.-Y. Lin, D. S. Munteanu, and E. Hovy, "ParaEval: Using paraphrases to evaluate summaries automatically," in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, 2006, pp. 447–454.