UTILIZING THE TRANSFORMER ARCHITECTURE FOR QUESTION ANSWERING

MD TAHMID RAHMAN LASKAR

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO
NOVEMBER 2020

Abstract

The Question Answering (QA) task aims at building systems that can automatically answer a question or query about the given document(s). In this thesis, we utilize the transformer, a state-of-the-art neural architecture to study two QA problems: the answer sentence selection and the answer summary generation. For answer sentence selection, we present two new approaches that rank a list of candidate answers for a given question by utilizing different contextualized embeddings with the encoder of transformer. For answer summary generation, we study the query focused abstractive text summarization task to generate a summary in natural language from the source document(s) for a given query. For this task, we utilize transformer to address the lack of large training datasets issue in single-document scenarios and no labeled training datasets issue in multi-document scenarios. Based on extensive experiments, we observe that our proposed approaches obtain impressive results across several benchmark QA datasets.

Dedication

I dedicate this thesis to my father Dr. Aminur Rahman Laskar and my mother Jamila Akhter Laskar. It's only because of their great care and sacrifice, I have become what I am today. I express my gratitude to the Almighty Allah for blessing me with them as my parents. Alhamdulillah.

Acknowledgements

In the name of the Almighty Allah, the Most Merciful, may He be glorified and exalted. Only because of His immense blessings, it's possible for me to successfully complete my thesis-based Master's degree in Computer Science at York University.

I have a long list of people whom I would like to say thank you. And at first, I would like to give thanks to my supervisors, **Dr. Jimmy Huang** and **Dr. Enamul Hoque Prince** for giving me the opportunity to work under their kind supervision. Without their tremendous support, guidance, instruction, and motivation, I could never come this far and get my research papers accepted in several prestigious venues, such as COLING, LREC, CANADIAN AI, and AMMCS in the last 2 years. I would like to express my sincere gratitude to them for always believing in me and inspiring me in my research.

I would also like to thank my thesis committee member, **Dr. Aijun An** for always providing me with the necessary help whenever I need any, as well as for spending her valuable time on my thesis.

I would like to thank my beloved parents, who grew me up as a human being with great care since my childhood. I would also like to thank my siblings (Vaia and Shazia), my grandmother, and all of my relatives for always inspiring me to pursue higher studies.

Moreover, I would like to mention some more names explicitly, which I start with thanking **Mir Rayat Imitiaz Hossain** for recommending me to York University. I would also like to thank my junior friend **M Saiful Bari** who always helped me in understanding complex concepts of Deep Learning and Natural Language Processing. In addition, thanks to all my friends (including juniors and seniors) at York University residence as well as to the IUTians in Canada for always helping me to overcome any difficulties that I faced in the last 2 years.

I must acknowledge that it was a great experience for me to work at the *Information Retrieval and Knowledge Management Research Lab* at York University. I would like to thank all my fellow researchers working at this lab for their great co-operation. I would also like to thank the Natural Sciences & Engineering Research Council (NSERC) of Canada and the ORF-RE (Ontario Research Fund-Research Excellence) award in BRAIN Alliance for supporting my research in the last 2 years. Last but not the least, I would like to acknowledge Compute Canada for allowing me to use their computing resources.

Preface

This thesis is submitted to the Faculty of Graduate Studies in partial fulfillment of the requirements for a Master of Science Degree in Computer Science. The entire work presented here is done by the author **Md Tahmid Rahman Laskar** under the supervision of **Dr. Jimmy Huang** and **Dr. Enamul Hoque Prince**. Some parts of this thesis have been published or accepted for publication as:

- Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Xiangji Huang. WSL-DS: Weakly Supervised Learning with Distant Supervision for Query Focused Multi-Document Abstractive Summarization. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING 2020)*, pages 5647-5654, Barcelona, Spain (Online). December 2020.

- Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Xiangji Huang. Query Focused Abstractive Summarization via Incorporating Query Relevance and Transfer Learning with Transformer Models. In *33rd Canadian Conference on Artificial Intelligence*, pages 342–348, Springer. May 2020.

- Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, and Enamul Hoque. Contextualized Embeddings based Transformer Encoder for Sentence Similarity Modeling in Answer selection Task. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*, pages 5505–5514, Marseille, France. May 2020.

- Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Xiangji Huang. Utilizing Bidirectional Encoder Representations from Transformers for Answer Selection. In *Proceedings of the V International Applied Mathematics, Modeling and Computational Science Conference (AMMCS 2019)*, Waterloo, Ontario, Canada. August 2019.

Table of Contents

List of Tables

List of Figures

1 Introduction

Question Answering is a research area of computer science which is concerned with building systems that can automatically answer questions asked by humans in natural language [28]. It lies within the fields of Information Retrieval and Natural Language Processing. With the rapid growth of textual documents on internet, accessing information from web has become a challenging problem [168]. In a web search, users may require the answer of a specific query or the summary about a certain topic from various sources to fulfill their information needs [164]. Moreover, the risen popularity of virtual assistants[1] such as Google Assistant, Siri, Alexa, Cortana, and etc., in recent years have gained significant interest to build conversational agents[2] using state-of-the-art technologies [6]. Since the performance of such web search engines or conversational agents largely depends on a system that possesses good question answering capabilities, the researchers are focusing to solve

[1] https://en.wikipedia.org/wiki/Virtual_assistant

[2] Conversational agents are computer systems which utilize natural language processing to converse with a human.

Table 1.1: An example of the Answer Selection and Ranking Task. A question along with list of candidate answers are given. The text in bold font is the correct answer.

Question:

- Which country won the FIFA world cup 2018?

List of Candidate Answers:

- England have won the Cricket World Cup 2019.
- **France have won the FIFA world cup 2018.**
- France have won the FIFA world cup 2014.

Potential Ranking:

- **France have won the FIFA world cup 2018.**
- France have won the FIFA world cup 2014.
- England have won the Cricket World Cup 2019.

different question answering tasks [35], such as: (i) *Answer Sentence Selection*: where the relevant text span containing the answer is selected from the source document or from a set of candidate answer spans, (ii) *Answer Summary Generation*: where the answer is generated in natural language via analyzing the given source document(s). For answer sentence selection, one common problem is the *Answer*

Table 1.2: An example of the Query Focused Text Summarization Task to Generate Abstractive Summary.

Query: What is the benefit of reality shows?
Document: Even if reality shows were not enlightening, they generate massive revenues that can be used for funding more sophisticated programs. Take BBC for example, it offers entertaining reality shows such as total wipeout as well as brilliant documentaries.
Summary: Reality show generates revenues.

Selection and Ranking Task, where a question and a set of candidate answers are given, and the task is to rank the candidate answers based on their relevance with the question (see Table 1.1). For answer summary generation, one example could be the *Query Focused Text Summarization Task*, where a set of document(s) along with a query are given and the goal is to generate a summary from the source document(s) based on the given query [168]. The generated summaries in this task can be either extractive or abstractive [10, 113, 40, 168]. For the extractive case, relevant text spans are directly extracted from the source document(s). But for the abstractive case, the generated summaries can contain words which may not appear in the source document(s) (see Table 1.2).

In this thesis, we focus to study two types of question answering problems:

(i) the answer sentence selection, and (ii) the answer summary generation. For the answer sentence selection problem, we study the answer selection and ranking task. For the answer summary generation problem, we study the query focused text summarization task where our focus is to generate the summaries in natural language (i.e., generating abstractive summaries). In the following sections, we first discuss our motivation behind studying the above mentioned topics. Then we present our contributions in this thesis. In the final section, we describe how we organized the remaining chapters of this paper.

1.1 Motivation

In recent years, the outstanding success of Deep Neural Network [46] models in different natural language processing problems has drawn significant attention among researchers to also utilize the neural network architecture to build question answering systems [10, 22, 43, 114]. For instance, in the answer selection and ranking task, various neural network models are extensively utilized in recent times to measure the relevance between the candidate answer sentence and the given question [22, 23]. In these neural network-based models, the representations of the given question and the candidate answer sentences play a significant role to measure the relevance between them [22, 23, 36, 127]. To represent the input sentences, tradi-

tionally these neural network-based models use word embeddings[3] like GloVe [118] or Word2Vec [103] for their initial representations. Then the initial representations of the input sentences are given as input to the encoder of the neural models. The encoded representations of the input sentences produced by the encoder of the neural models are then utilized to measure the similarity between the question and the candidate answer [22, 23]. Finally, all the candidate answers for a given question are ranked based on their similarity score with the question.

However, the prior body of work for the answer selection and ranking task has some major limitations. For instance, most prior work for this task usually utilized the Recurrent Neural Network [55] architecture, while more recent neural models are rarely adopted for such tasks [172]. Recently, Vaswani et al. [150] proposed the *Transformer* architecture which significantly outperformed the widely used recurrent neural network models in various of natural language processing problems [150, 122]. Despite the superiority of the transformer architecture over the recurrent neural network models, it has been rarely evaluated on the answer selection and ranking task [122]. Furthermore, previous work for such tasks mostly concentrated on using word embeddings like GloVe or Word2Vec [118, 103, 22, 23, 113, 10, 161]. A critical limitation of GloVe or Word2Vec is that such embeddings can only provide fixed representation of a word and fail to capture the context when the word is used

[3]Word Embedding is the learned representation that maps words from a vocabulary to vectors of real numbers.

in different sentences [119]. Recently, the Deep Contextualized Word Representations (ELMo) [119] and the Bidirectional Encoder Representation from Transformers (BERT) [36] models have received lots of attention as these models can provide contextualized representations of each word in a sentence instead of the traditional fixed word embeddings produced by GloVe or Word2Vec. Thus, due to the advantages of the transformer model as well as the contextualized embeddings mentioned above, we are motivated to use them for the answer selection and ranking task.

For the query focused text summarization task, most of the recent state-of-the-art models also leveraged various deep neural models [113]. Note that for extractive summary generation, only the encoder of the neural models are required to extract the most relevant text spans from the source document(s) as the summary [168, 172, 114, 10, 113]. In contrast, for the abstractive summary generation, a decoder is also needed along with the encoder. Because in such scenarios, the encoder is used for the input document representation which is later utilized by the decoder to generate the answer or summary in natural language [114, 62]. Thus, the existing state-of-the-art models in different query focused summarization datasets where the requirement is to generate abstractive summaries utilize various encoder-decoder based neural network models [114, 113, 10]. Though the transformer model has been effectively utilized in various natural language generation problems in recent years [122], to the best of our knowledge, it has not been utilized for the

query focused abstractive text summarization task yet. Thus, our motivations to use the transformer architecture for the query focused summarization task are the following: (i) the transformer model is not utilized for such tasks yet [122], and (ii) due to the superiority of this model over the other neural models [150, 122].

Moreover, there are some important challenges that we aim to address for the query focused summarization task. Note that the query focused summarization task can be done in two scenarios: (i) *Single-Document Scenario*: where the goal is to generate a summary from a single source document. (ii) *Multi-Document Scenario*: where the goal is to generate a summary from a set of documents [10]. One major challenge in the single-document query focused summarization task is that the available datasets for such tasks are very small in size compared to the generic abstractive summarization datasets [113, 10, 138]. To address this issue, we utilize transfer learning via leveraging the transformer architecture. For the query focused summarization task in multi-document scenarios, one major challenge is that the currently available datasets for such tasks do not contain any training data [10]. To tackle this issue, we propose a weakly supervised learning model via utilizing various pre-trained transformer models. In the following section, we present our contributions in this thesis.

1.2 Contributions

As we mentioned earlier, in this thesis we study two types of question answering problems: *answer sentence selection* and *answer summary generation*. For answer sentence selection, we propose several new transformer-based methods for the *answer selection and ranking task* and conduct extensive experiments to set new state-of-the-art results in six answer sentence selection datasets. For answer summary generation, we propose several novel methods via utilizing the transformer architecture for the *query focused text summarization task* and observe new state-of-the-art results in benchmark datasets for such tasks. In the following, our contributions in this thesis in the answer selection and ranking task and the query focused text summarization task are stated.

Answer Selection and Ranking Task

- In the answer selection and ranking task, we investigate how to utilize the contextualized word embeddings by integrating them with the transformer encoder. For that purpose, we present two new approaches (fine-tuning-based and feature-based) via utilizing various contextualized embeddings, such as ELMo, BERT, and RoBERTA [75, 77, 79].

- We observe that combining contextual embeddings with the transformer encoder is very effective for answer selection. However, while comparing our

fine-tuning-based approach with the feature-based approach, we find that the fine-tuning-based approach significantly outperforms the feature-based approach in all datasets used for comparisons. Moreover, we observe a new state-of-the-art result in six benchmark question-answering datasets in terms of various evaluation metrics using our fine-tuning-based models [79].

Query Focused Text Summarization Task

- For the query focused text summarization task, we focus to generate abstractive summaries based on both single-document and multi-document scenarios.
- In the query focused single-document abstractive summarization task, we address the lack of availability of large training datasets by introducing a transfer learning approach via utilizing transformer-based models. In our proposed approach, we first pre-train our transformer-based model on a large generic abstractive summarization dataset. Then for the target dataset, we incorporate query relevance in the pre-trained model and fine-tune it for the query focused abstractive summarization task [76].
- In the query focused multi-document abstractive summarization task, one major challenge is that the currently available datasets for such tasks do not contain any training data. To tackle this issue, we utilize datasets similar to the target dataset as the training data. However, these training datasets only

contain multi-document gold reference summaries and do not contain the gold reference summary for each single-document in a document set. Thus, the effectiveness of utilizing supervised learning with neural summarization models cannot be utilized here due to the computational complexity of training such models in long text documents [92, 12, 26, 69, 174]. To address this issue, we propose a novel weakly supervised learning model [78].

- In our proposed weakly supervised learning model for query focused abstractive summarization in multi-document scenarios, at first we generate the weak reference summary of each single-document in a document set using various pre-trained transformer models. Then, we leverage the effectiveness of fine-tuning pre-trained generic summarization models for the query focused abstractive summarization task [76] via introducing an iterative approach. In our proposed iterative approach, we adopt a pre-trained transformer-based single-document summarization model and iteratively generate the summary of each single-document in a multi-document set. Finally, for each document set we utilize an answer selection model to select the most relevant sentences from the generated summary as the final summary.

- Experimental results in different datasets for the query focused text summarization task for both single-document and multi-document scenarios show

that our proposed approaches set new state-of-the-art results in terms of various evaluation metrics.

- In addition, to investigate the effectiveness of incorporating query relevance in transformer models for such tasks, we also propose a new attention mechanism. However, while investigating the effectiveness of utilizing query relevance using our proposed attention mechanism in transformer models, we surprisingly find that the Debatepedia dataset which is used for the query focused summarization task is more of a generic summarization dataset and the queries in this dataset have little to no effect in summary generation.

As a secondary contribution, our source codes used in this thesis have been made publicly available here: *https://github.com/tahmedge/Tahmid-MSc-Thesis-YorkU*.

1.3 Organization of the Thesis

Before moving to the next chapter, we give a brief description below regarding how the following chapters of this thesis are organised.

- We start the Chapter 2 with a brief introduction to machine learning and deep learning, followed by their applications in natural language processing and information retrieval. We then discuss various deep learning architectures followed by different word embedding techniques which are extensively used

in deep neural models in a wide range of natural language understanding and generation problems in recent years. Then we review the transformer architecture and various language models based on it. Finally, we review different question answering tasks (the answer sentence selection task and the answer summary generation task) that we have studied in this thesis.

- Then in Chapter 3, we first briefly discuss the background of the answer selection and ranking task that we study for the answer sentence selection problem in this thesis. In the same chapter, we describe our proposed approaches for this task along with the datasets that we used to evaluate the proposed models. Moreover, our experimental details as well as the results are also discussed in this chapter.

- In Chapter 4, we discuss the answer summary generation problem that we study in this thesis: the query focused text summarization task. As mentioned earlier, we study this task to generate abstractive summaries based on two scenarios: (i) Single-Document, and (ii) Multi-document. For both scenarios, we discuss the background, our proposed approaches, the datasets, the experimental settings, and the results in this chapter.

- Finally, the concluding remarks of this thesis as well as our plans for future work are discussed in Chapter 5.

2 Literature Review

This chapter begins with a brief introduction to Machine Learning [106], which is the foundation of the Deep Learning [46] architecture that we utilize in this thesis for various question answering systems. Then we conduct a literature review on deep learning that has been extensively used in recent years in a wide range of tasks in information retrieval and natural language processing. Then we discuss various Word Embedding [118, 103] techniques that are utilized with such deep neural networks. Afterwards, we discuss the Transformer [150] model which is based on the deep neural network architecture that we use in this thesis. Next, we discuss some recent language models[4] based on the transformer architecture. Finally, we review different question answering tasks that we study in this thesis: i) the *Answer Sentence Selection* task, and ii) the *Answer Summary Generation* task.

[4]A Language Model is a probability distribution over a sequence of words that aims at predicting the probability of a word appearing next in a text sequence.

2.1 Machine Learning

We have now entered the era of Big Data, where the amount of information on the internet is ever increasing [106]. Thus, it is required to have systems that can analyze this myriad of data automatically without any human intervention. This is where machine learning comes into play since it can automatically detect patterns in data to predict the future events as well as perform different kinds of decision making [106]. In particular, machine learning is a branch of artificial intelligence [133] that analyzes data to provide computer systems the ability to learn automatically from experience without any human intervention [106, 140]. All machine learning algorithms are required to be trained on a given dataset, which we call as the training data. Then the trained machine learning algorithms perform inference in new data that were not used during the training procedure. To perform machine learning, the following two approaches are mostly utilized [106]. These are:

- Supervised Learning: In supervised learning, each example in the training data contains the label of the correct answer. In such datasets, the machine learning algorithms learn a mapping from the inputs to the outputs to predict the labels in the unseen test data.

- Unsupervised Learning: In unsupervised learning, the examples in the train-

ing data don't contain any labels. In such datasets, the machine learning algorithms need to find interesting patterns in order to make predictions in the test data.

Apart from the above two main approaches, there are some other machine learning approaches that have been used [106], such as:

- Reinforcement Learning: In this approach, algorithms are trained to make sequence of decisions based on reward and punishment. Such algorithms are rewarded when their action is the desired one and punished when their action is the undesired one. In particular, reinforcement learning algorithms learn through trial and error to maximize the cumulative reward in order to attain their desired goals.

- Semi-Supervised Learning: It combines supervised learning with unsupervised learning via utilizing a small amount labeled training data with a large amount of unlabeled training data to train the machine learning algorithms.

- Weakly Supervised Learning: When the given datasets are unlabeled, weak supervision utilizes various noisy or imprecise sources to generate the labels for the unlabeled datasets to train the machine learning models.

2.2 Deep Learning

Deep learning is a sub-field of machine learning which is based on the Artificial Neural Network architecture. The artificial neural network is a machine learning model that mimics human brain to solve computational problems [106, 140]. It can be described as a directed graph where the nodes denote *neurons*[5] and the edges denote the connections between different neurons. The architecture of an artificial neural network usually consists of an input layer and a output layer along with some hidden layers in between them. Each connection in an artificial neural network has been assigned an *weight* and each neuron in different layers receives the weighted sum of the outputs of the neurons connected to its incoming edges as input. Note that the outputs of the neurons in different layers are usually calculated using a function called *activation function*. While training a neural network, the weights of the connections are updated which is done most commonly using a method called *back-propagation* [17] to minimize the errors made by the network. To calculate the errors, neural networks utilize a function called *loss function* (also known as *cost function*) that compares the output predicted by the neural network with the desired output [46, 140]. Afterwards, the back-propagation method tries to minimize the loss by calculating the gradient to update the weights [131, 46, 140]. Moreover, there are some parameters called *hyperparameters* that

[5]In a human brain, a neuron is the basic working unit.

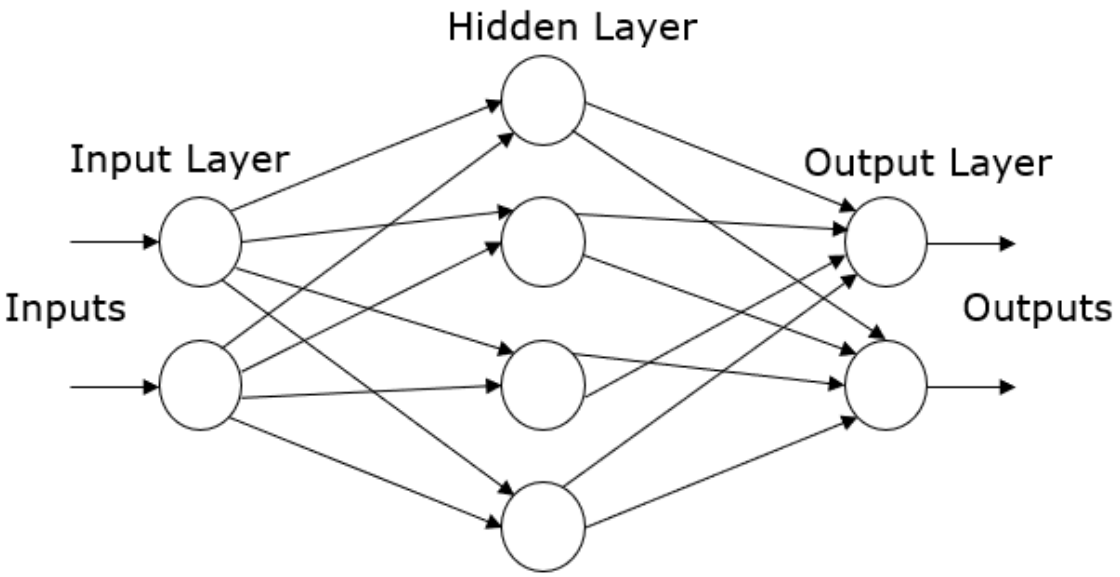


Figure 2.1: A feedforward neural network with one hidden layer. Both the input and output layer contain two neurons and the hidden layer contains four neurons.

have to be set before starting to train an artificial neural network. Some examples of hyperparameters are, *learning rate*: which determines the step size for the gradient to reach the minimum of a loss function, *batch size*: which denotes the number of examples used in each training iteration of the neural network, and etc.

While there are several types of artificial neural network, one of the simplest forms is the Feedforward network, where the underlying graph like structure doesn't contain any cycles [140, 46]. The feedforward networks usually contain an input layer, followed by some hidden layers and finally an output layer (an example

of a Fully Connected[6] feedforward network is illustrated in Figure 2.1). When the artificial neural network contains multiple hidden layers, it is called as Deep Neural Network. In recent years, deep neural network (i.e., deep learning) has gained significant popularity in various research areas of computer science, such as: Computer Vision [151], Natural Language Processing [172, 84], Information Retrieval [84], Bioinformatics [104] and etc. In areas such as Natural Language Processing (NLP) and Information Retrieval (IR), two of the most extensively used deep neural network architectures are the *Convolutional Neural Network* [70, 80, 57] and the *Recurrent Neural Network* [172, 84]. Since in this thesis, we are studying various question answering problems where the recurrent neural network models were mostly utilized in recent years [172, 22, 23, 113], we give a brief introduction to the readers about this architecture below.

Recurrent Neural Network (RNN): RNN is a special type of artificial neural network to process sequential information. It can be considered as multiple copies of the same network where each network passes information to the successor. For example, to process a sequence of text using RNN, at each time step, each word in the text sequence is sequentially given as input to the RNN network where the information of the previous word is also utilized for processing. Note that for

[6]In a full connected layer, each neuron in a layer has a connection to every neuron in another layer.

tasks such as text understanding, the layers that learn the encoded representation of the text sequence in RNN models are called encoders; while for tasks such as text generation, RNN models use additional decoder layers along with the encoder layers [172]. One major drawback of the standard RNN models is that in some text processing tasks, the future information of the input sentence may need to be known while processing the current state (e.g: for tasks such as text summarization, the overall context of the source sentence may be required to generate a summary). For such scenarios, the Bidirectional RNN was proposed which includes an additional recurrent layer in the network via duplicating the original recurrent layer [137]. In the first recurrent layer, the original input sequence is given as input. Whereas for the second recurrent layer, a reversed copy of the input sequence is given. In this way, the future information of the input is known via allowing the network to be trained simultaneously in both positive and negative time direction. However, when the text sequence is very long, there is a major issue called the *long-term dependency* problem occurs in RNN models due to the *vanishing* or *exploding gradient* problems that leads the RNN to lose relevant information at a given time step [54, 55, 56]. To address these issues, a special type of RNN called the Long Short-Term Memory (LSTM) was proposed [55]. The advantage of LSTM over the traditional RNN is that it has a special state call the cell state that consists of an input gate, an output gate and a forget gate. This cell helps LSTM to remember the relevant

information as well as forget the irrelevant ones allowing it to mitigate the vanishing or exploding gradient issues to tackle the long-term dependency problem in long text sequences. Moreover, various variants of LSTM were also proposed, of which some of the strongest variants are the Bidirectional LSTM (BLSTM) [148], and the Gated Recurrent Unit (GRU) [25]. Similar to the Bidirectional RNN, the BLSTM model was also proposed to take into account both the future and current information. However, there are some key architectural differences between the other LSTM variant the GRU architecture and the traditional LSTM architecture. For instance, in contrary to the traditional LSTM model, the GRU architecture doesn't have the output gate in its cell [25]. Instead, it has an update gate along with a reset gate which is a combination of the input gate and the forget gate used in LSTM [55, 25]. It should be noted that with lesser complexity than the LSTM, the GRU still provides almost similar performance compared to the LSTM in most tasks [27, 25]. Nonetheless, the performance of LSTM is still strictly stronger than GRU in many scenarios [14, 162].

2.3 Word Embeddings

Neural network architecture has been widely used in recent years to solve various NLP problems [172]. However, since neural networks are unable to process non-numerical values and can only process numerical representations, one crucial step

to process natural language text using neural models is to learn the numerical representation of each word in a text. This is where word embedding comes into play. It contains a set of vectors where each vector represents the distributed numerical representation of a word in a vector space [118, 103]. In the vector space, words having similar meaning are usually located closer since the similar words tend to have similar representations in the word embedding vectors. Below, we discuss different word embedding techniques that are used with deep learning-based models to solve various NLP tasks.

Traditional Word Embeddings: Over the years, word embedding vectors were created using various shallow neural networks which were later used by the deep learning models to represent the words in a given text sequence [172]. Utilizing such word embeddings has been responsible for achieving state-of-the-art performance in many NLP tasks [172]. However, there are some limitations while training a word embedding model in a given dataset. For example, if the size of the training data for the given task is small, the word embedding vectors may not contain the representations of many words that could appear in the test data. Moreover, learning the word embeddings for a given task may also take considerable amount of time when the size of the training dataset is large. To address these issues, instead of training the word embedding vectors from scratch for the target dataset,

utilizing Transfer Learning[7] via leveraging pre-trained word embeddings [118, 103] are mostly utilized by various deep learning models in recent years [172]. One major advantage of using pre-traiend word embeddings is that they are trained in a large dataset and thus it allows these embeddings to contain the vector representations of words which appear rarely or did not appear at all in the training dataset of the target domain [172, 103, 118]. Two of the most popular such word embeddings that have been used extensively in recent years are the Word2Vec [103] embedding and the GloVe [118] embedding [172]. Word2Vec embeddings are learnt using neural networks via utilizing two methods: i) *Common Bag Of Words (CBOW)* and ii) *Skip-gram* [103]. The *CBOW* method uses the context words of the target word as input to the neural network and tries to predict the target word based on these context words. For the *Skip-gram* method, the target word is given as input to the neural network which then learns to predict its surrounding context words. However, one major issue with Word2Vec is that it can only take the local context while being trained [172, 103, 118]. To address this issue, the GloVe model was proposed [118]. In contrary to the neural network-based architecture of Word2Vec, the GloVe model doesn't leverage any neural networks while being trained. The GloVe model addresses the local context issue in Word2vec via proposing a new global log bilinear regression model [118]. In this model, a global word-word co-

[7]It is a technique where the knowledge gained while solving one problem is applied to a different but related problem.

occurrence matrix was constructed by combining the global matrix factorization method and the local context window method. This global co-occurrence count allows the embedding of each word in GloVe to obtain the global information [118]. However, one common limitation with both Word2Vec and Glove is that they can only provide the fixed representation of a word and fail to capture its context when the word is used in different sentences [172]. To overcome this issue, the contextualized word embeddings have been proposed [172]. In the following, we review the contextualized word embeddings.

Contextualized Word Embeddings: Instead of the fixed word embeddings provided by the traditional word embeddings like GloVe or Word2Vec, the contextualized word embeddings provide the embedding of a word based on its overall context in a sentence. Thus, such word embeddings can capture multiple meanings of a word based on where it has been used in different sentences. One such contextualized embeddings is the Embeddings from Language Models (ELMo) [119], which is pre-trained on a vast amount of text data [18] for the language modeling task using a BLSTM [55] network. This bidirectional language modeling objective allows the ELMo model to have a sense of both the next and the previous words while predicting the target word. The ELMo model provides three layers of representations for each word: one layer provides a character based word representation

whereas the other two layers are the LSTM hidden states. The weighted sum of these three layers is the ELMo layer and the output of this layer is used as the contextualized word embeddings [119]. More recently, a new language model called the Bidirectional Encoder Representations from Transformers (BERT) [36] model was proposed which could also provide contextualized embeddings like ELMo. However, in contrary to the BLSTM network used in ELMo, the BERT model utilized the encoder of the transformer architecture [150] (we will discuss the transformer architecture in details later). Besides, the BERT model used a different language modeling objective called *masked language modeling* [147], along with utilizing the model for the *next sentence prediction* task during the pre-training stage [36]. Though there are several differences between BERT and ELMo, the BERT model was also pre-trained in large datasets similar to the ELMo model [119, 36].

2.4 Deep Learning Applications in NLP and IR

In this section, we give a brief review of different deep learning models applied in various areas of NLP and IR for tasks such as text understanding and text generation. For that purpose, we divide this section into two parts. First, we discuss the applications of deep learning models in various natural language understanding tasks. Then, we discuss their applications in various natural language generation tasks.

Natural Language Understanding: Sentiment analysis [94, 173], paraphrase identification [22, 23], named entity recognition [7], and answer sentence selection [43] are some examples of the natural language understanding task. The objective of such tasks is to predict the correct label of the given input. Thus, in scenarios when deep learning models are applied to these tasks, only the encoder part of the neural network is required to classify the encoded representation of the input text to the correct label. In recent years, various deep learning models were proposed for such tasks which significantly outperformed the previously used non-neural network based approaches [172]. More recently, state-of-the-models for the natural language understanding task leveraged the advantage of pre-trained language models [123, 124, 15, 119, 36, 96, 74, 30]. It was found that via *fine-tuning*[8] the transformer-based pre-trained language models, significant improvement in performance could be achieved in a wide range of natural language understanding problems [15, 36, 96, 74]. Note that the new state-of-the-art results using the pre-trained language models in several natural language understanding datasets also led to the release of various sophisticated benchmarks to evaluate the generalized effectiveness of these models [155, 156, 87, 117, 47]. One of the most notable such benchmarks is the General Language Understanding Evaluation benchmark (GLUE)[9]. The moti-

[8]Fine Tuning is a process where a neural model already trained for a given task is trained again for a similar task in a new dataset via updating the weights of the original trained model.

[9]https://gluebenchmark.com/

vation behind the GLUE benchmark is that, NLP models should be able to process language in such a way that they are not only used exclusively for a specific task or dataset. Rather, these models should be robust enough to a diverse range of language understanding tasks [155]. However, with fine-tuning various sophisticated pre-trained language models, the human baseline in the GLUE benchmark has already been outperformed [96, 74, 156]. Very recently, a new benchmark similar to GLUE named SuperGLUE[10] was introduced [156]. The SuperGLUE benchmark contains a new set of datasets for various natural language understanding tasks which are more complex than the tasks and the datasets used in the GLUE benchmark [156]. Though the overall human baseline in SuperGLUE is not yet surpassed, yet some models [96, 125] have already outperformed the human baseline in SuperGlue for several tasks along with providing the overall performance almost similar to the human level [125].

Natural Language Generation: For natural language generation, some of the examples are neural machine translation [5, 25, 24], abstractive text summarization [132, 110], and dialogue generation [85]. In such tasks, the outputs are required to be generated in natural language. Similar to the impressive performance of deep neural network models in various natural language understanding problems, models

[10]https://super.gluebenchmark.com/

based on this architecture also provide superior performance in a wide range of natural language generation tasks [172, 5, 25, 24, 132, 110]. However, contrary to the traditional architecture of neural models for the natural language understanding task where the requirement is only the encoder of a neural net, for the natural language generation task a decoder is also needed along with the encoder [25, 24, 143]. In recent years, RNN-based neural models that utilized the encoder-decoder framework[11] [25, 24, 143] obtained excellent results for natural language generation in numerous datasets. Despite being effective to generate text in natural language, the RNN architecture has a common limitation called the long-term dependency problem which occurs in long text sequences [172]. To tackle the long-term dependency issue, various variants of RNN such as LSTM [55], as well as the attention mechanism [5, 97] (see Footnote 11) have been used in recent years. Though these variants of RNN have provided improvement over the standard RNN architecture in several tasks [55, 5, 97], the long-term dependency issue was still present in these variants [150]. Very recently, a novel neural network architecture called the transformer was proposed for the language generation problem [150]. Note that the transformer model is also based on the encoder-decoder framework that utilizes the attention mechanism. However, for the transformer architecture, a new attention mechanism called self-attention was proposed [150]. Experimental results show that the trans-

[11]A detailed description of the encoder-decoder framework as well as the attention mechanism is given on Section 2.5.1

former model could address the long-term dependency issue more effectively than other neural models which allows it to set a new state-of-the-art result in the neural machine translation task [150]. Moreover, models based on the transformer architecture also provide state-of-the-art performance in many other natural language generation problems, such as: abstractive summarization [92], question generation [38], abstractive answer generation [114], and conversational response generation [178]. In the following section, we give a detailed description of the transformer architecture.

2.5 The Transformer Model

In this section, we review the transformer model [150], which is the deep learning architecture that we utilize in this thesis for various question answering tasks. To review the transformer model, we divide this section into three parts. First, we discuss the background of the transformer model by reviewing the encoder-decoder based deep learning architecture [25, 24, 143] as well as the attention mechanism [5, 97] since the transformer model is based on the encoder-decoder framework that utilizes self-attention [150]. Then we describe the architecture of the transformer model followed by discussing various transformer-based pre-trained language models [123, 124, 15, 119, 36] that provide impressive success in a wide range of NLP tasks in recent years.

2.5.1 Background

Since the transformer model is an encoder-decoder model [25, 24, 143] that utilizes the attention mechanism [5, 97], we give readers a brief introduction to the encoder-decoder models and the attention mechanism here.

The Encoder-Decoder Model: The encoder-decoder models [24, 25] are used for the sequence to sequence modeling tasks [143] such as neural machine translation [25, 24, 97] or abstractive text summarization [138, 110, 132]. Such models usually consist of two RNNs [172, 143]. One of these two RNNs is called the encoder, which encodes a sequence of symbols into a fixed-length vector representation. The other is called the decoder, which decodes the representation into another sequence of symbols. One major limitation with such encoder–decoder based RNN models is that, the encoder needs to compress all the information of the source sentence into a single fixed-length vector which is the encoded representation obtained from the final hidden state of the encoder [24, 25]. However, this approach leads to rapid performance deterioration when the length of the input sentence increases [25, 5]. This is because in a long text sequence, the long-range dependency problem occurs as the compressed encoded representation tends to forget the earlier parts of the sequence and leads to the loss of the context [25, 5].

Attention Mechanism: In order to address the long-range dependency issue in the encoder-decoder based neural models [25, 24], the attention mechanism was proposed [5, 97]. The attention-based neural encoder-decoder models utilize a sequence of vectors called context vectors to contain the information of the source sentence [5, 97]. This sequence of vectors is constructed by generating a context vector in each time step via utilizing the previous hidden state of the decoder as well as all the encoder hidden states. This allows the decoder to attend to all the encoder hidden states along with utilizing its own previous hidden state to generate the appropriate target word for the current hidden state. In this way, the decoder is able to obtain the most relevant information from the context vector while generating the target word. In particular, the decoder utilizes the context vector to access the information from the source sequence which is relevant to generate the target word during the current time step. Thus, the utilization of the attention mechanism alleviates the need for the encoder layer to compress the whole source sentence into a single fixed-length vector [5, 97]. Experimental results show that utilizing the attention mechanism is very effective to process not only long text sequences, but also text sequences of any length [5, 97].

2.5.2 Transformer Architecture

The transformer model contains an encoder and a decoder [150]. It was proposed by Vaswani et al. [150] for the sequence to sequence modeling tasks. The transformer model obtained impressive results on tasks such as neural machine translation along with handling the long-term dependency problem more effectively than RNN and its variants such as LSTM, BLSTM, or GRU [150]. In Figure 2.2, the transformer architecture is shown[12]. In the following, we first give a brief description of the encoder and decoder of transformer.

Transformer Encoder: The encoder of transformer contains a stack of six identical layers. Each encoder layer is divided into two sub-layers: (i) The Multi-Head Self-Attention Layer, and (ii) The Position-wise Fully Connected Feed-Forward Network. Both sub-layers contain a *residual connection* [51] and utilize *layer normalization* [4].

In the multi-head self-attention layer, each token in the input text can give attention to all other tokens in the same text. The authors of transformer named this attention mechanism as Scaled Dot-Product Attention (see Figure 2.2c) [150]. For the attention calculation, the authors first created three matrices named query vector Q, key vector K, and value vector V. In the first encoder layer, for each token

[12]The figure is taken from the paper: Attention Is All You Need [150].

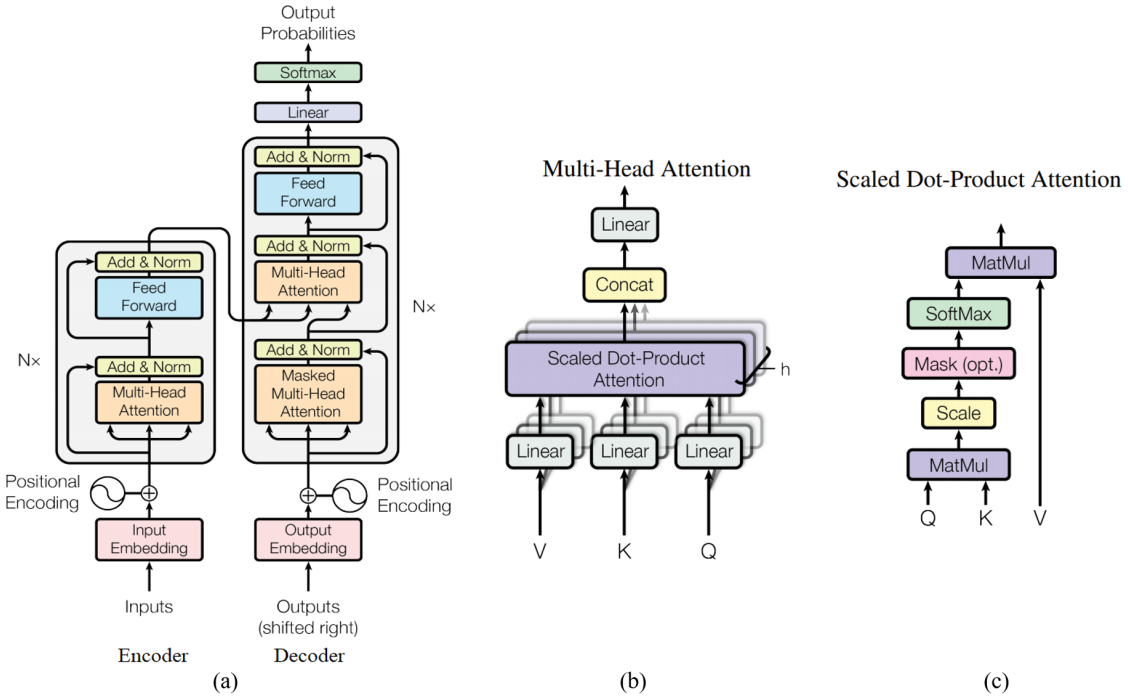


Figure 2.2: The Transformer Architecture[12]: (a) The Encoder and Decoder of Transformer. (b) The Multi-Head Attention Mechanism: head (h) = 8. (c) The Scaled Dot-Product Attention Mechanism. This picture is taken from the paper "Attention is All You Need" by Vaswani et al. [150]

in the input sequence, three matrices (Q, K, and V) are created based on the dot-product between the embedding vector X with the three weight matrices W^Q, W^K, and W^V respectively. Note that these weight matrices are also updated during the training process. Afterwards, the output Z for each token in the self-attention layer is calculated based on the following formula:

$$softmax\left(\frac{\mathrm{Q}\times\mathrm{K}^{\mathrm{T}}}{\sqrt{\mathrm{d}_k}}\right)\mathrm{V}=\mathrm{Z} \tag{2.1}$$

In equation (2.1), $\sqrt{\mathrm{d}_k}$ is the square root of the dimension of the key matrix K. The transformer encoder also uses the multi-head attention mechanism [150] by calculating the self-attention eight times with eight different Query/Key/Value weight matrices to obtain eight Z matrices (see Figure 2.2b). It then concatenates the eight Z matrices into a single matrix and multiplies that matrix with an additional weight matrix. Finally, the resulting matrix is sent to the fully connected feedforward layer [140]. This layer consists of two linear transformations [33] with RELU activation [45] which is stated in the following equation.

$$FFN(x)=max(0,xW_1+b_1)W_2+b_2 \tag{2.2}$$

All the key, query, and value vectors in the subsequent encoder layers come from the output of their respective previous layers.

Transformer Decoder: The decoder of the transformer also consists of a stack of six identical layers similar to the encoder. However, in addition to the two sub-layers (the multi-head self-attention layer and the fully connected feed-forward layer as used in each encoder layer), the decoder inserts another sub-layer, which performs multi-head attention over the output of the encoder stack. In the decoder, the query

Q comes from the previous decoder layer, but the key K and the value V comes from the output of the encoder. Similar to the sub-layers in the encoder, each sub-layer in the decoder also contains residual connections followed by layer normalization. To reserve the auto-regressive property of the decoder [5, 97], the self-attention sub-layer of decoder is modified to prevent it from attending to subsequent positions by allowing it to only attend to the previous positions [150].

In transformer, the dimension of input and output is $d_{model} = 512$. The dimension of the inner-layer is $d_{ff} = 2048$, To keep track of positional information of the tokens in a sequence, sinusoidal positional encodings are added to the input embeddings at the bottoms of both encoder and decoder stacks [150].

2.5.3 Language Models based on Transformer

Though the original transformer model [150] contains an encoder and a decoder to solve the sequence-to-sequence problems, various language models based on only the encoder or the decoder of the transformer were also proposed recently for different natural language understanding tasks [36, 16, 123]. For example, Cer et al. [16] utilized the transformer encoder to generate embeddings from sentences for transfer learning [53] to other tasks. Radford et al. [123] utilized the transformer decoder by pre-training it on large text corpora for the language modeling task and then fine-tuned the model for downstream tasks. One of the most remarkable trans-

former encoder based model is the BERT [36] model which showed state-of-the-art performance across several NLP tasks [155].

The BERT model was pre-trained for masked language modeling [147] and next sentence prediction task on the BooksCorpus (800M words) [180] dataset along with the English Wikipedia (2,500M words) [36]. For the masked language modeling task, 15% tokens in each text sequence are replaced with a [MASK] token. The model then attempts to predict the original value of the masked words based on the context provided by the non-masked words in the sequence. In the next sentence prediction task, the model receives a pair of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document. After being pre-trained on large datasets, the BERT model learns the contextualized representation of each word in a sentence. Moreover, with just fine-tuning the pre-trained BERT model in the task specific datasets, state-of-the-art results have been achieved in a wide range of natural language understanding tasks [36, 155]. In order to fine-tune the pre-trained BERT model in a task specific dataset, a special $[CLS]$ token is required to be added at the beginning of the input text [36]. If there are multiple sequences in the input text, then these sequences are combined together into a single sequence by separating them with a special $[SEP]$ token. Moreover, for BERT fine-tuning, new parameters are added for an additional classification layer which along with the parameters of the pre-trained BERT model

are jointly updated to maximize the log-probability of the correct label. Note that during the fine-tuning stage, only the output of the first token ($[CLS]$) is taken to predict the label of the input sequence. The output of this token is considered as the aggregate representation of the entire input sequence [36].

The outstanding performance of the BERT model [36] also led to the development of various new models [166, 96, 74, 134] that utilized the pre-trained language modeling objective. For example, Liu et al. proposed the MT-DNN model [91] that combined the BERT model with the multi-task neural network [90] to effectively learn multiple related tasks jointly along with outperforming the original BERT model. The newly proposed Transformer-XL [34] architecture was utilized by the XLNet model [166] which also outperformed BERT in several NLP tasks by effectively learning contextual representations. The RoBERTa model [96] proposed new design choices and training strategies for the traditional BERT model that resulted in significant performance gain in several natural language understanding tasks. Different parameter reduction techniques were utilized by the recently proposed ALBERT model that provided significant performance improvement over the original BERT model along with leveraging lower memory consumption and higher training speed [74]. Besides, the ALBERT model set new state-of-the-art results in various natural language understanding datasets [155]. More recently, a new method to pre-train a smaller general-purpose language model called DistilBERT

was proposed [134]. Via leveraging knowledge distillation during the pre-training stage to reduce the size of a BERT model by 40%, it was shown that DistilBERT could still provide comparable performance with its larger counterpart BERT by retaining 97% of the language understanding capabilities of BERT along with having 60% faster speed than BERT [134].

In addition, language models based on BERT are also proposed for many other languages [100, 81, 32] except English, as well as being proposed as a multilingual model [121] via pre-training it on multiple languages. Moreover, language models based on BERT are proposed for various other domains as well, such as the biomedical domain [47, 82], the clinical language processing domain [1], as well as for understanding the source codes of different programming languages [41, 64] and etc.

2.6 The Question Answering Task

In this section, we discuss recent work on the question answering task by dividing it into two categories. In the first category, we review the question answering task where the requirement is to select the answer from the source document(s). In the second category, we focus on the question answering task where the requirement is to generate the answer in natural language. Since in this thesis, we primarily study the *answer selection and ranking* task for the *answer sentence selection* problem

and the *query focused abstractive summarization* task for the *answer summary generation* problem, we mainly review these two tasks here.

2.6.1 Answer Sentence Selection

The answer sentence selection task is a fundamental problem of NLP and IR [169, 22, 23]. One common problem for such tasks is the *answer selection and ranking* task, where given a question with a list of candidate answers, the objective is to rank the candidate answers based on their relevance with the question [22, 23]. For this task, various sentence similarity modeling frameworks are utilized to measure the relevance between the query and the candidate answer [21, 22, 20, 126, 127]. Earlier, the sentence similarity models for such tasks relied on different feature engineering-based approaches [152, 169]. One example of the feature engineering-based sentence similarity model is the work of Yih et al. [169], where the WordNet[13] based semantic features were utilized to measure the similarity between the question and the candidate answer sentence. However, these feature engineering-based approaches have some critical limitations. For instance, the features which are used in one dataset may not perform well in another dataset [22]. Moreover, these approaches require lots of handcrafted rules and are often error-prone. Recently, several deep learning-based approaches for sentence similarity

[13]https://wordnet.princeton.edu/

modeling showed impressive performance without requiring any handcrafted features [23, 22, 126, 161, 21, 13, 99, 149, 139, 63, 127, 89]. In such deep neural models, some researchers focused on extracting the common features in a sentence pair to improve the performance of sentence similarity modeling [22, 23, 161]. For instance, a collaborative and adversarial network was proposed in which a generator and a discriminator had been utilized to extract the common features between the question and the candidate answer [22]. In addition, to improve the performance of recurrent neural network-based sentence similarity models, Chen et al. [23] utilized the contextual information from the question sentence while generating the hidden states of the candidate answer sentence. They first detected the aligned words in a question-answer pair and then utilized the context of these aligned words to generate the hidden state representation of the candidate answer sentence. Moreover, various recurrent neural models also utilized the attention mechanism [144, 136] to improve the sentence representation for sentence similarity modeling.

Though the recurrent neural network architecture was extensively used for the answer selection and ranking task, the transformer architecture has been rarely evaluated for such tasks [150]. Very recently, some models were proposed which utilized the pre-trained transformer encoders such as BERT or RoBERTa for the answer selection and ranking task [73, 43]. Some work also utilized transfer learning from such transformer-based models via first pre-training the model in a large

dataset for the answer selection and ranking task followed by fine-tuning it in the target dataset [73, 43]. Though these models provided state-of-the-art performance in different datasets such as TREC-QA [159] and WikiQA [165], the effectiveness of combining transfer learning with pre-trained transformer encoders are not deeply investigated yet: i) these models were not evaluated in other datasets such as community question answering datasets [107, 108, 109], ii) these models were not compared with some of the state-of-the-art pre-trained transformer encoder models such as RoBERTa [96] that did not require such transfer learning.

2.6.2 Answer Summary Generation

The answer summary generation task is a natural language generation problem where the requirement is to generate the required answer or summary from the given source document(s). However, there is a major difference between the generic text summarization task and the answer summary generation task. In the generic text summarization task, the goal is to just summarize the given source document(s); whereas in the answer summary generation task, a question or query is also included along with the input document(s) and the summary or answer is generated based on the given query. Thus, the *query focused text summarization* task is an example of the answer summary generation problem [114, 35].

For the query focused text summarization task, sentence representation of the

input text plays a vital role to generate the summary [113, 168]. Earlier, most deep learning-based models utilized various pre-trained word embedding techniques like GloVe [118] or Word2Vec [103] to obtain the initial representation of the input text. The fixed embedding representations extracted from such word embeddings are then fed into the neural models which are later utilized by the encoder of neural nets to learn the encoded representations [168]. For extractive summary generation, only the encoder of the neural net is needed to extract the relevant text spans from the input text. But when the requirement is to generate abstractive summaries, a decoder is also needed along with the encoder since the abstractive summaries may contain some words which are not appeared in the source document(s) [138].

For abstractive summarization, various encoder-decoder based neural models have been utilized in recent years [132, 110]. As in the summarization task, most salient features from the source document(s) are required for summary generation, various attention-based neural encoder-decoder architectures are also proposed [135, 19]. However, one of the major problems of these neural models is that they tend to repeat the same word multiple times which leads to generate non-cohesive summaries [138, 113]. To overcome this issue, See et al. [138] proposed the copy and coverage mechanism. Note that all work stated above used various models based on the recurrent neural network architecture. Very recently, the BERT for SUMmarization (BERTSUM) model was [73] proposed for abstractive summarization

where the BERT model [36] was used as the encoder and the transformer decoder [150] was used as the decoder. Despite being a promising approach to generate abstractive summaries, the transformer architecture [150, 92] is not utilized for the query focused abstractive text summarization task yet.

It is also worth mentioning that even though significant research has been done in different datasets for generic abstractive or extractive summarization, the amount of work for the query focused abstractive summarization task is very limited [113, 10]. Moreover, the amount of datasets available for the query focused summarization task is also very small [10]. Among the datasets used for such tasks, one of the most widely used datasets is the dataset from Document Understanding Conference[14] (DUC) [10]. This dataset is used for the query focused summarization task for both extractive [98, 40, 164] and abstractive [10] cases in multi-document scenarios. Though most of the previous work for the query focused summarization task was concentrated on multi-document scenarios [9, 98, 40, 10], some research in recent years also studied this problem in single document scenarios [113, 10]. For the query focused summarization task in the single-document scenario, Nema et al. [113] created a dataset from the Debatepedia[15] which is the only available dataset for such tasks in the single-document scenario.

[14]https://duc.nist.gov/

[15]http://www.debatepedia.org/

3 Answer Selection and Ranking Task

3.1 Background

In the *Answer Selection and Ranking Task*, one crucial step is to measure the similarity between the question and the candidate answer [169]. For this task, given a question along with a list of candidate answers, at first the similarity between the question with each candidate answer is measured. Then based on the similarity score, the candidate answers are ranked.

Since the neural models have been found to be more effective than the non-neural models for sentence similarity modeling, several models based on the recurrent neural network architecture or the convolutional neural network architecture are utilized in recent years for the answer selection and ranking task to measure the similarity between the input sentences (the question and the candidate answer) [50, 49, 21, 22, 20, 126, 127, 171, 89]. In such neural models, word embeddings like GloVe [118] or Word2Vec [103] are mostly utilized to obtain the initial representations of the input sentences. Then the input sentences based on their word

embedding representations are fed into the neural models to obtain their encoded representations. Note that the encoded representation is the sentence level representation of each input sentence obtained from the neural network. Afterwards, the similarity between the encoded representation of each sentence produced by the neural models is measured [23, 22]. However, since in this approach various word embeddings like GloVe or Word2Vec are used, one critical limitation of using such embeddings is that they could only provide the fixed representation of a word and fail to capture the context when the same word is used in different sentences.

Recently, contextualized word representation models such as ELMo and BERT have shown superior performance over traditional word embeddings such as GloVe or Word2Vec in a wide range of tasks [119, 36]. For the ELMo model [119], a deep bidirectional language model is pre-trained on a large text corpora to learn the contextualized representations of a word. For the BERT model [36], the encoder of transformer [150] is utilized to learn the contextualized representations. Since these contextual embeddings can capture better representation of a sentence via generating embedding of each word based on its surrounding context, in this thesis, we use such contextualized embeddings for the answer selection and ranking task. Moreover, due to the superiority of the transformer architecture over the other neural network models, we combine these embeddings with the encoder of the transformer model to investigate the effectiveness of using contextualized embeddings with the

transformer encoder for the answer selection and ranking task.

We organize this chapter as follows. First, we propose two new approaches for the answer selection and ranking task via utilizing different contextualized embeddings, namely, the ELMo, BERT, and RoBERTA [96] and integrate these embeddings with the transformer encoder. Then we present the details of our experimental setup that we used to evaluate our proposed approaches. Finally, we discuss the findings of our experiments to end this chapter.

3.2 Our Proposed Approach

Let us assume that we want to measure the similarity between the two sentences $X = x_1, x_2, ..., x_m$ and $Y = y_1, y_2, ..., y_n$ for the answer selection and ranking task. In this thesis, we accomplish this goal via combining the transformer encoder [150] with different contextualized embeddings [36, 119, 96]. For that purpose, we propose two approaches: i) Feature-based approach, and ii) Fine-tuning-based approach.

In the feature-based approach, our framework works in the following steps as demonstrated in Figure 3.1(a). For the given input sentences X and Y, our model first creates contextualized embedding representations $\mathbf{x_i}$ for each token $x_i \in X$ and $\mathbf{y_i}$ for each token $y_i \in Y$ using the pre-trained BERT/ELMo model. Then for each sentence, the token embeddings are combined with the positional encodings [150] in order to track the order of the token sequence. Then the contextualized embedding

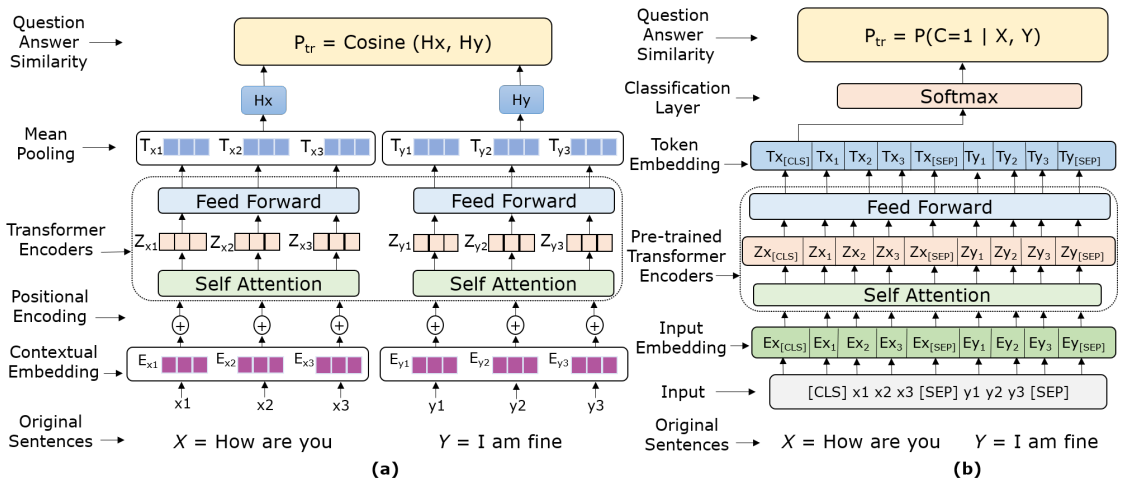


Figure 3.1: Our similarity modeling framework that applies contextualized embeddings: (a) Feature-based approach using randomly initialized transformer encoders, and (b) Fine-tuning-based approach using pre-trained transformer encoders.

representations of all tokens in each sentence are sent to two separate randomly initialized transformer encoders to obtain encoded representations. After the self-attention calculation in each transformer encoder, the resulting representations are passed through the feed-forward and pooling layers to obtain the condensed vectors H_X and H_Y for each sentence X and Y respectively. Finally, the cosine similarity is calculated between H_X and H_Y.

In the fine-tuning approach (see Figure 3.1(b)), we adopt a pre-trained transformer encoder model (e.g., BERT/RoBERTa) and concatenate the given question and a candidate answer to give as input to the model [36, 96]. Then, we fine-tune

the pre-trained transformer encoder model for the question-answer similarity task to identify the most relevant candidate answers. In the following, we describe these two approaches in detail.

3.2.1 Feature Based Approach

As shown in Figure 3.1(a), our feature-based approach adopts the encoder from the transformer model [150]. It is to be noted that the original transformer architecture includes both the encoder and the decoder to perform the machine translation task. In contrast, we only adopt the encoder of transformer and combine it with contextualized embeddings to measure the similarity between the question and the candidate answer for the answer selection and ranking task. For that purpose, we first extract the ELMo/BERT embeddings [119, 36] of each sentence and feed them to two separate randomly initialized transformer encoders. Each encoder uses a self-attention layer to represent each token based on other words in the input sentence. This self-attention calculation is done by utilizing the following three vectors for each token, a Query vector Q, a Key vector K, and a Value vector V. Note that these three vectors are initially created by multiplying the embedding vector with three weight matrices (W_Q, W_K, W_V) respectively. Then, the output Z for each word based on self-attention is computed as:

$$\mathrm{Z} = softmax\left(\frac{\mathrm{Q} \times \mathrm{K}^{\mathrm{T}}}{\sqrt{\mathrm{d}_k}}\right)\mathrm{V} \tag{3.1}$$

Note that the transformer encoder uses multi-head attention mechanism to give attention on different positions. This is done by calculating the self attention eight times with eight different Query/Key/Value weight matrices to obtain eight Z matrices. The eight weight matrices used for self-attention calculation are also updated during the training phase. Afterwards, the eight Z matrices are concatenated into a single matrix which is then multiplied by an additional weight matrix. The resulting matrix is then sent to a feed-forward layer. Then, we apply the mean pooling method [157] to obtain the sentence representation H. Finally, we utilize the Cosine distance similarity function and restrict it to a range of [0, 1] to calculate the similarity score between the question and the candidate answer.

$$\mathrm{S}\left(X, Y\right) = cosine(H_X, H_Y) \tag{3.2}$$

In order to obtain contextualized embeddings, we extract features from both ELMo and BERT models. In the following, we describe these embeddings.

ELMo Embeddings: Instead of using fixed embedding representation, ELMo provides contextual embedding of a word based on its context in the entire sentence [119]. Thus, it can capture multiple meanings of a word based on where it is used.

ELMo uses a bidirectional LSTM [55] to have a sense of both the next and the previous word. It is pre-trained on a vast amount of text data [18] and provides three layers of representations for each word: one layer provides a character based word representation and the other two layers are the LSTM hidden states. The ELMo layer is the weighted sum of these three layers. We use the output of this layer as contextualized word embeddings.

BERT Embeddings: The BERT model [36] can also provide contextualized embeddings like ELMo. The model was originally pre-trained for masked language modeling and next sentence prediction task on the BooksCorpus (800M words) [180] dataset along with the English Wikipedia (2,500M words). We use the token embeddings generated from the BERT model and feed them into the transformer encoder.

3.2.2 Fine Tuning Based Approach

Instead of using contextualized embeddings as input to a randomly initialized transformer encoder, we also utilize various pre-trained transformer encoder models such as BERT and RoBERTa for the answer selection and ranking task as demonstrated in Figure 3.1(b). Below, we describe different pre-trained transformer encoder models that we fine-tune for the answer selection and ranking task.

BERT Fine-tuning for Answer Selection: In the BERT model, sentence pairs are combined together into a single sequence, separated by a special token $[SEP]$. The output of BERT is taken only for the first token ($[CLS]$), which is considered as the aggregate representation of the sequence. During fine-tuning, parameters are added for an additional classification layer W. All the parameters of the pre-trained BERT model along with W are fine-tuned jointly to maximize the log-probability of the correct label. The label probabilities $P \in R^K$ (where K is the total number of classifier labels) are calculated as follows:

$$P = softmax(CW^T) \tag{3.3}$$

In the answer selection task, there are two classifier labels (similar = 1, dissimilar = 0). In the original BERT model [36], sentence pair classification task was done by predicting the correct label (1 or 0). But in our work, we modify the final layer by only considering the predicted score P_{BERT} for the similarity label to rank the answers based on the question-answer similarity score.

$$P_{BERT} = P(C = 1|X, Y) \tag{3.4}$$

RoBERTa Fine-tuning for Answer Selection: Since the BERT model was significantly under-trained, the RoBERTa model was proposed by modifying different hyperparameters in BERT along with new design choices [96]. More specifically, RoBERTa used a much larger mini batches and learning rates compared

to the BERT. Also, the next sentence prediction task was removed from the pre-training stage in RoBERTa. In addition, while the BERT model was pre-trained on only two datasets, the RoBERTa model was pre-trained on five different datasets. These new parameter settings and objectives showed significant improvements over the original BERT model in different NLP tasks [96]. To utilize it for the answer selection and ranking task, we followed the similar approach of BERT fine-tuning by modifying the final layer to obtain the similarity score $P_{RoBERTa}$.

$$P_{RoBERTa} = P(C = 1|X, Y) \tag{3.5}$$

3.3 Experimental Setup

In this section, we present the datasets, evaluation metrics, the training procedure, and the parameter settings that we used in our experiments.

3.3.1 Datasets

To evaluate the effectiveness of our approach, we ran experiments on six different datasets for the answer selection and ranking task as shown on Table 3.1. Specifically, we used two widely used question answering (QA) datasets, namely, the TREC-QA and WikiQA as well as four community question answering (CQA) datasets, namely, the YahooCQA, SemEval-2015CQA, SemEval-2016CQA, and

Table 3.1: Dataset Overview ('#' denotes 'Number of' and 'RAW' indicates the 'Original' version).

Dataset		# Questions			# Candidate Answers		
		Train	**Dev**	**Test**	**Train**	**Dev**	**Test**
TREC-QA	**RAW**	1229	82	100	53417	1148	1517
	Cleaned	1229	65	68	53417	1117	1442
WikiQA[16]	**RAW**	2118	296	633	20360	2733	6165
	Cleaned	873	126	243	8672	1130	2351
YahooCQA		50112	6289	6283	253440	31680	31680
SemEval-2015CQA		2600	300	329	16541	1645	1976
SemEval-2016CQA		4879	244	327	36198	2440	3270
SemEval-2017CQA		4879	244	293	36198	2440	2930

SemEval-2017CQA. These datasets are described below.

TREC-QA: The TREC-QA dataset is created from the QA track (8-13) of Text REtrieval Conference [159]. It has two versions: RAW and Cleaned. The difference between the two versions is that the RAW version has some questions for which there is no answer or there are only positive/negative answers, whereas the Cleaned version removes those instances from the development and test sets. As a result, the RAW version contains 1148 question-answer pairs in the development set

and 1517 question-answer pairs in the test set, while the Cleaned version contains 1117 question-answer pairs in the development set and 1442 question-answer pairs in the test set.

WikiQA: This is an open domain QA dataset [165] in which the answers were collected from the Wikipedia. In this dataset, there are many questions that do not contain any answers. Only 873, 126, and 243 questions out of 2118, 296, 633 questions in the training, development, and test sets contain any answers, respectively[16].

YahooCQA: This dataset was prepared for answer selection task by Tay et al. [145] from the *Yahoo! Answers Manner Question*[17] dataset. It is a community-based question answering dataset and comparatively larger than TREC-QA or WikiQA. Each question in the YahooCQA dataset is associated with at most one correct answer. The negative answers were generated by sampling 4 samples from the top 1000 hits obtained via Lucene[18] search. There are 253440, 31680, and 31680 question-answer pairs in the training, development, and test sets respectively.

SemEval-2015CQA: This CQA dataset is created from the *Qatar Living Forums*[19]. We focus on subtask A, the question-comment similarity task. Each com-

[16]For WikiQA, we used the original training data. But evaluation was done only on the cleaned test data where questions having no correct answers were removed.

[17]https://webscope.sandbox.yahoo.com

[18]https://lucene.apache.org/

[19]https://www.qatarliving.com/forum

ment is tagged with "Good", "Bad" or "Potentially Useful". We consider "Good" as the positive example and other tags as the negative examples by following the work of Sha et al. [139].

SemEval-2016CQA: This is another CQA dataset created from *Qatar Living Forums*. Though the task is similar to SemEval-2015CQA, the dataset used in SemEval-2016CQA is different.

SemEval-2017CQA: This one has the same training and development sets as SemEval-2016CQA. Only the test set is different in the SemEval-2017CQA which contains 293 questions whereas the SemEval-2016CQA contains 327 questions.

3.3.2 Evaluation Metrics

Similar to the recent work on the answer selection and ranking task [127, 43, 73], we used Mean Average Precision and Mean Reciprocal Rank as evaluation metrics to measure the performance of our models.

Mean Average Precision (MAP): The MAP of a model for a set of questions Q is the mean of the average precision score of each question obtain based on the retrieval performance. Note that in the domain of information retrieval, the precision score for a given query is the ratio of the retrieved documents that are relevant to the user's query to the total number of documents that are retrieved. In the answer selection and ranking task, the average precision is the mean of the

precision score of each relevant answer that is retrieved. If the total number of retrieved answers is n, then the Average Precision (AP) is calculated as follows [175]:

$$AP = \frac{\sum_{i=1}^{n} P@i}{R} \tag{3.6}$$

Here, R is the total number of relevant answers given in a set of candidate answers, and $P@i$ is the precision of the top-i retrieved answers ($P@i$ is 0 if the retrieved answer i is not a relevant answer). After calculating the AP for each question q, the MAP for the set of questions Q is calculated as follows [11]:

$$MAP = \frac{1}{Q} \sum_{q=1}^{|Q|} AP_q \tag{3.7}$$

Mean Reciprocal Rank (MRR): The reciprocal rank of a model for a given question is the reciprocal of the rank of the first correct answer [31]. The MRR is the mean of the reciprocal ranks of the results that are obtained for a set of questions Q. If the rank of the first correct answer for each question i is $rank_i$, then the MRR is calculated as follows:

$$MRR = \frac{1}{Q} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{3.8}$$

3.3.3 Training and Parameter Settings

Here, we discuss the training parameters of our models that we used in our experiments.

In our feature-based approach, the dimensions of the input layer and the output layer as well as the inner feed forward layer were same. Specifically, when we used ELMo, the dimensions of the hidden layers $\mathbf{d_{model}}$ and the feed forward layers $\mathbf{d_{ff}}$ of the transformer encoder were both set to 1024. When we used the $\text{BERT}_{\text{Base}}$ model, the size of $\mathbf{d_{model}}$ and $\mathbf{d_{ff}}$ were set to 768. Similar to the original transformer architecture [150], we set the number of attention heads $\mathbf{A}$ to 8. However, based on the performance in the development set, we used one encoder layer ($\mathbf{L} = 1$) instead of six identical layers used in the original transformer model. For the parameter update, we used the Adam optimization algorithm [68] and for the loss function, we used the mean squared error. Moreover, we set the learning rate to 5×10^{-5} and the dropout value to 2×10^{-1}.

For fine-tuning, we experimented with both the Base and the Large versions of BERT and RoBERTa for the pairwise sentence classification task [36]. For training, we used cross entropy loss function to calculate the loss. The parameters of the $\text{BERT}_{\text{Base}}$ and $\text{RoBERTa}_{\text{Base}}$ models were: $\mathbf{d_{model}} = 768$, $\mathbf{d_{ff}} = 3072$, $\mathbf{A} = 12$, $\mathbf{L} = 12$. For the $\text{BERT}_{\text{Large}}$ and $\text{RoBERTa}_{\text{Large}}$ models, the parameters were: $\mathbf{d_{model}} = 1024$, $\mathbf{d_{ff}} = 4096$, $\mathbf{A} = 16$, $\mathbf{L} = 24$. For all models, the Adam was used as the optimizer, the batch size $\in \{8, 16, 24\}$, and the learning rates $\in \{1e-5, 2e-5\}$.

3.3.4 Implementation

We implemented our models using Pytorch[20]. For fine-tuning, we used the HuggingFace Transformer[21] model [163] and ran our experiments in multi GPU [44, 105] settings with 4 Nvidia V100 GPUs. For the feature-based approach, we used a Pytorch-based implementation of the Transformer Encoder[22] and ran our experiments using a single Nvidia 1080 GPU. The BERT contextualized embeddings were generated using the MXNet[23] library and the ELMo contextualized embeddings were generated using the AllenNLP[24] library.

3.4 Results and Discussions

We performed extensive experiments to compare our contextualized embeddings based transformer encoder (CETE) with the recent progress. To understand the effectiveness of our approaches, we also compared with several baselines.

For the feature-based approach, we used a baseline that had the transformer encoder but utilized the GloVe word embeddings [118] which could not consider the contextual information. As the dimensions in the hidden layers and the feed

[20]https://pytorch.org/

[21]https://github.com/huggingface/transformers

[22]https://github.com/AnubhavGupta3377/Text-Classification-Models-Pytorch

[23]https://mxnet.apache.org/

[24]https://allennlp.org/

forward layers in our feature-based approach were same as the dimension of the contextualized embeddings, we also used the dimensions in those layers for this baseline same as the dimension of GloVe: $\mathbf{d_{model}} = 300$, $\mathbf{d_{ff}} = 300$, $\mathbf{A} = 6$, $\mathbf{L} = 1$.

For the fine-tuning approach, we compared our models with the fine-tuned XL-Net model [166]. Note that the XLNet model did not use the original transformer architecture [150]. Rather, it utilized the Transformer-XL [34] architecture by using the segments recurrence mechanism as well as the relative encoding scheme [166]. It also proposed the permutation-based language modeling to capture bidirectional context [166]. Similar to our fine-tuning-based models, we use both the Base and the Large versions of XLNet and keep the hyperparameters such as optimizer, batch size, and learning rates identical to our fine-tuning-based models.

First, we show the results our experiments in the TREC-QA and the WikiQA datasets in Table 3.2 and Table 3.3 respectively. Next, the performance of our models in the four CQA datasets, namely, the YahooCQA, the SemEval-2015CQA, the SemEval-2016CQA, and the SemEval-2017CQA are shown in Table 3.4, 3.5, 3.6, 3.7 respectively. Below, we first discuss the performance of our feature-based approach in these datasets, followed by discussing the performance of our fine-tuning-based approach in these datasets.

Table 3.2: Performance comparisons on the TREC-QA dataset.

Model	TREC-QA			
	RAW		Cleaned	
	MAP	MRR	MAP	MRR
Rao et al. [126]	0.780	0.834	0.801	0.877
Chen et al. [21]	-	-	0.781	0.851
Chen et al. [22]	-	-	0.841	0.917
Chen et al. [23]	-	-	0.823	0.889
Tay et al. [146]	0.770	0.825	0.784	0.865
Madabushi et al. [99]	0.836	0.863	0.865	0.904
Tymoshenko et al. [149]	0.777	0.869	-	-
Kamath et al. [63]	0.852	0.891	-	-
Rao et al. [127]	0.774	0.843	-	-
Yoon et al. [171]	-	-	0.875	0.940
Lai et al. [73]	-	-	0.914	0.957
Garg et al. [43]	-	-	**0.943**	0.974
Transformer Encoder + GloVe	0.708	0.764	0.728	0.812
CETE (ELMo Embeddings)	0.798	0.869	0.791	0.858
CETE ($\text{BERT}_{\text{Base}}$ Embeddings)	0.799	0.855	0.791	0.857
CETE ($\text{BERT}_{\text{Large}}$ Embeddings)	0.806	0.897	0.789	0.887
$\text{XLNet}_{\text{Base}}$ Fine Tuning	0.903	0.939	0.900	0.938
$\text{XLNet}_{\text{Large}}$ Fine Tuning	0.939	0.979	0.920	0.973
CETE ($\text{BERT}_{\text{Base}}$ Fine Tuning)	0.891	0.925	0.888	0.953
CETE ($\text{BERT}_{\text{Large}}$ Fine Tuning)	0.917	0.947	0.905	0.967
CETE ($\text{RoBERTa}_{\text{Base}}$ Fine Tuning)	0.927	0.962	0.905	0.950
CETE ($\text{RoBERTa}_{\text{Large}}$ Fine Tuning)	**0.950**	**0.980**	0.936	**0.978**

Table 3.3: Performance comparisons on the WikiQA dataset.

Model	WikiQA	
	MAP	**MRR**
Rao et al. [49]	0.709	0.723
Bian et al. [13]	0.754	0.764
Chen et al. [21]	0.721	0.731
Chen et al. [22]	0.730	0.743
Chen et al. [23]	0.736	0.745
Sha et al. [139]	0.746	0.758
Tay et al. [146]	0.712	0.727
Tymoshenko et al. [149]	0.762	0.776
Zhang et al. [177]	0.766	0.780
Liu et al. [89]	0.735	0.751
Kamath et al. [63]	0.700	0.716
Yoon et al. [171]	0.834	0.848
Lai et al. [73]	0.857	0.872
Transformer Encoder + GloVe	0.671	0.686
CETE (ELMo Embeddings)	0.762	0.774
CETE ($\text{BERT}_{\text{Base}}$ Embeddings)	0.727	0.741
CETE ($\text{BERT}_{\text{Large}}$ Embeddings)	0.714	0.731
$\text{XLNet}_{\text{Base}}$ Fine Tuning	0.808	0.820
$\text{XLNet}_{\text{Large}}$ Fine Tuning	0.836	0.847
CETE ($\text{BERT}_{\text{Base}}$ Fine Tuning)	0.829	0.843
CETE ($\text{BERT}_{\text{Large}}$ Fine Tuning)	0.843	0.857
CETE ($\text{RoBERTa}_{\text{Base}}$ Fine Tuning)	0.847	0.860
CETE ($\text{RoBERTa}_{\text{Large}}$ Fine Tuning)	**0.900**	**0.915**

Table 3.4: Performance comparisons on the YahooCQA dataset.

Model	YahooCQA	
	MAP	**MRR**
Tay et al. [146]	-	0.801
Transformer Encoder + GloVe	0.667	0.667
CETE (ELMo Embeddings)	0.762	0.762
CETE ($\text{BERT}_{\text{Base}}$ Embeddings)	0.776	0.776
CETE ($\text{BERT}_{\text{Large}}$ Embeddings)	0.778	0.778
$\text{XLNet}_{\text{Base}}$ Fine Tuning	0.939	0.939
$\text{XLNet}_{\text{Large}}$ Fine Tuning	0.945	0.945
CETE ($\text{BERT}_{\text{Base}}$ Fine Tuning)	0.948	0.948
CETE ($\text{BERT}_{\text{Large}}$ Fine Tuning)	0.951	0.951
CETE ($\text{RoBERTa}_{\text{Base}}$ Fine Tuning)	0.951	0.951
CETE ($\text{RoBERTa}_{\text{Large}}$ Fine Tuning)	**0.955**	**0.955**

Table 3.5: Performance comparisons on the SemEval-2015 dataset.

Model	SemEval-2015	
	MAP	**MRR**
Transformer Encoder + GloVe	0.843	0.864
CETE (ELMo Embeddings)	0.875	0.909
CETE ($\text{BERT}_{\text{Base}}$ Embeddings)	0.890	0.924
CETE ($\text{BERT}_{\text{Large}}$ Embeddings)	0.883	0.923
$\text{XLNet}_{\text{Base}}$ Fine Tuning	0.929	0.960
$\text{XLNet}_{\text{Large}}$ Fine Tuning	0.945	0.969
CETE ($\text{BERT}_{\text{Base}}$ Fine Tuning)	0.923	0.949
CETE ($\text{BERT}_{\text{Large}}$ Fine Tuning)	0.935	0.961
CETE ($\text{RoBERTa}_{\text{Base}}$ Fine Tuning)	0.933	0.956
CETE ($\text{RoBERTa}_{\text{Large}}$ Fine Tuning)	**0.947**	**0.970**

Table 3.6: Performance comparisons on the SemEval-2016 dataset.

Model	SemEval-2016	
	MAP	**MRR**
Sha et al. [139]	0.801	0.872
Transformer Encoder + GloVe	0.741	0.810
CETE (ELMo Embeddings)	0.767	0.824
CETE ($\text{BERT}_{\text{Base}}$ Embeddings)	0.773	0.835
CETE ($\text{BERT}_{\text{Large}}$ Embeddings)	0.765	0.831
$\text{XLNet}_{\text{Base}}$ Fine Tuning	0.849	0.912
$\text{XLNet}_{\text{Large}}$ Fine Tuning	0.860	0.912
CETE ($\text{BERT}_{\text{Base}}$ Fine Tuning)	0.843	0.906
CETE ($\text{BERT}_{\text{Large}}$ Fine Tuning)	0.866	0.927
CETE ($\text{RoBERTa}_{\text{Base}}$ Fine Tuning)	0.851	0.900
CETE ($\text{RoBERTa}_{\text{Large}}$ Fine Tuning)	**0.888**	**0.938**

3.4.1 Effectiveness of Feature-based Approach

Performance on TREC-QA and WikiQA: For the TREC-QA and the WikiQA datasets, we can see from both Table 3.2 and Table 3.3 that integrating the transformer encoder with ELMo or BERT have outperformed the baseline where only the GloVe embedding was used with the transformer encoder. Specifically, in terms of MAP, our CETE model with ELMo achieves 13.56% improvement over the transformer encoder with GloVe in the WikiQA dataset. For the TREC-QA dataset, the improvement is 12.71% in the RAW version and 8.65% in the Cleaned version over the baseline. Though our best performing CETE

Table 3.7: Performance comparisons on the SemEval-2017 dataset.

Model	SemEval-2017	
	MAP	**MRR**
Nakov et al. [109]	0.884	0.928
Transformer Encoder + GloVe	0.824	0.881
CETE (ELMo Embeddings)	0.860	0.914
CETE ($\text{BERT}_{\text{Base}}$ Embeddings)	0.875	0.922
CETE ($\text{BERT}_{\text{Large}}$ Embeddings)	0.867	0.922
$\text{XLNet}_{\text{Base}}$ Fine Tuning	0.902	0.934
$\text{XLNet}_{\text{Large}}$ Fine Tuning	0.930	0.962
CETE ($\text{BERT}_{\text{Base}}$ Fine Tuning)	0.904	0.942
CETE ($\text{BERT}_{\text{Large}}$ Fine Tuning)	0.921	0.963
CETE ($\text{RoBERTa}_{\text{Base}}$ Fine Tuning)	0.909	0.944
CETE ($\text{RoBERTa}_{\text{Large}}$ Fine Tuning)	**0.943**	**0.974**

model with ELMo could not outperform the state-of-the-art [73] in the WikiQA dataset, it outperformed or provided comparable performance with many recent work [22, 23, 139, 149, 63]. For both versions of TREC-QA, the feature-based CETE models do not outperform the state-of-the-art [63, 43]. However, their performances are still comparable or better than many recent work [146, 149, 23, 127].

Performance on CQA datasets: We notice from Table 3.4, 3.5, 3.6, 3.7 that our proposed approach of integrating transformer encoder with ELMo or BERT have again outperformed the baseline in all the CQA datasets. Specifically, in terms of MAP, our best performing feature-based approach $\text{BERT}_{\text{Large}}$ achieves 16.64%

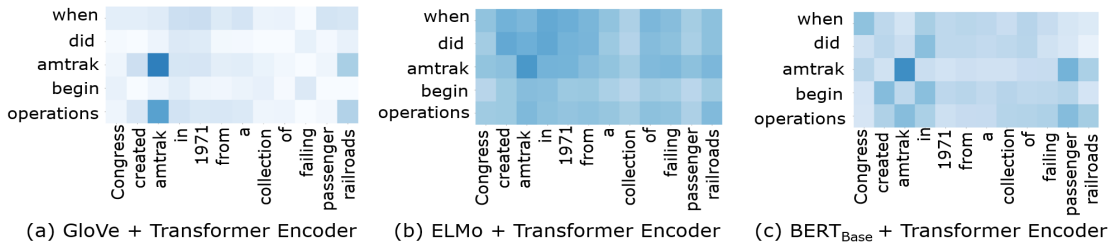
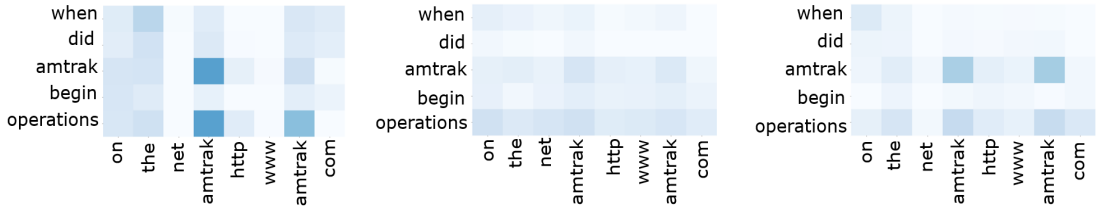


Figure 3.2: Similarity between the words of a question and a relevant candidate answer. Here, darker color indicates more similarity.

improvement over the transformer encoder with GloVe in the YahooCQA dataset. In the SemEvalCQA datasets, The CETE model with BERT$_{\text{Base}}$ performs the best with an improvement of 5.58%, 4.32%, and 6.19% over the baseline in terms of MAP in the SemEval-2015CQA, SemEval-2016CQA, and SemEval-2017CQA datasets respectively. Though none of our feature-based approaches outperform the current state-of-the-art models[25] [109, 139], they show comparable performance in each dataset.

Case study: To get deeper insights about why our feature-based CETE models with ELMo or BERT are more effective than the transformer encoder with GloVe, we randomly selected some question and candidate answer pairs and analyze the word-by-word similarity between them. Figure 3.2 shows the word-by-word similarity heatmap for a relevant question and candidate answer pair. We observe that

[25]We did not report any recent progress for SemEval2015-CQA as we found that prior work used different evaluation metrics.



(a) GloVe + Transformer Encoder (b) ELMo + Transformer Encoder (c) $\text{BERT}_{\text{Base}}$ + Transformer Encoder

Figure 3.3: Similarity between the words of a question and an irrelevant candidate answer. Here, lighter color indicates less similarity.

for the similar sentence pair, words between two sentences are more similar in the CETE model with ELMo/BERT embeddings than the transformer encoder with GloVe model. It indicates that for the similar sentence pair, the transformer encoder with contextualized embeddings effectively captures the overall context of the sentence, resulting in better performance for answer selection. For the dissimilar sentence pair, we observe from Figure 3.3 that the words between the sentence pair were more dissimilar in the CETE model than the transformer encoder with GloVe model, suggesting the effectiveness of integrating contextualized embeddings with the transformer encoder.

3.4.2 Effectiveness of Fine-tuning-based Approach

Performance on TREC-QA and WikiQA: We fine-tune the pre-trained BERT, RoBERTa, and XLNet models for the answer selection and ranking task. Among these three models, BERT and RoBERTa are transformer encoder based, whereas

the XLNet model is based on the Transformer-XL model. From the Table 3.2 and 3.3, we find that the Large versions of BERT, RoBERTa, and XLNet always outperform their respective Base versions in the TREC-QA and WikiQA datasets respectively. While comparing XLNet with BERT, we find that XLNet outperforms BERT in both versions of the TREC-QA. However, in the WikiQA dataset, the BERT model achieves superior performance. In all datasets, the RoBERTa model outperforms both XLNet and BERT.

In comparison to the prior work, we observe new state-of-the-art results in the RAW version of TREC-QA by fine-tuning with both Base and Large versions of BERT. The fine-tuned $\text{BERT}_{\text{Large}}$ and $\text{BERT}_{\text{Base}}$ models have an improvement of 7.63% and 4.58% respectively in terms of MAP over the previous state of the art [63]. However, these models do not achieve the state-of-the-art results in the Cleaned version of TREC-QA [43] as well as in the WikiQA [73]. With the fine-tuned RoBERTa models, we observe even more improvements in the WikiQA and the RAW version of TREC-QA compared to the fine-tuned BERT models. In the RAW TREC-QA, both the Base and Large versions of RoBERTa outperform the previous state-of-the-art [63], with $\text{RoBERTa}_{\text{Large}}$ performing the best with MAP 0.950 and MRR 0.980. In the WikiQA dataset, our fine-tuned $\text{RoBERTa}_{\text{Large}}$ model sets a new state-of-the-art result with an improvement of 5.02% in terms of MAP and 4.93% in terms of MRR over the previous best performing model [73].

Though our approach of fine-tuning RoBERTa$_{\text{Large}}$ provides new state-of-the-results in TREC-QA (RAW) and WikiQA datasets, in terms of MAP it could not outperform the RoBERTa$_{\text{Large}}$ with Transfer Learning approach: the RoBERTa-TANDA model [43] in the Cleaned version of TREC-QA. It is to be noted that in the two-step fine-tuning-based RoBERTa-TANDA model, the first step of fine-tuning was done in a large dataset created from the Wikipedia [72] which contains 57242 questions, along with more than 20 Millions candidate answers. Then the second step of fine-tuning was done in the target domain. In comparison to them, we only do a one-step fine-tuning in the target domain which contains 1229 questions with 53417 candidate answers in the training set (our training data size is only about 2% of total questions and 0.3% of total candidate answers used to train the RoBERTa-TANDA model). Without the leverage of large datasets, our fine-tuning approach provides almost similar result in terms of MAP with only 0.75% less than the RoBERTa-TANDA model. In terms of the MRR, we observe a new state-of-the-art result with an improvement of 0.41% compared to the RoBERTa-TANDA model. We could not conduct any significance tests to determine whether the performance difference between our fine-tuned RoBERTa$_{\text{Large}}$ model and the RoBERTa-TANDA model [43] is statistically significant or not because the RoBERTa-TANDA model was only evaluated[26] on the cleaned version of the TREC-QA dataset among the

[26]We did not report the result of RoBERTa-TANDA for the WikiQA dataset as the number of questions and the candidate answers used in their test data were different than ours.

datasets that we used in our experiments. Thus, we did not have enough results for the RoBERTa-TANDA model to conduct the significance test.

Performance on CQA datasets: In all CQA datasets, we find from Table 3.4, 3.5, 3.6, 3.7 that the RoBERTa model again outperforms both BERT and XLNet. Among BERT and XLNet, we find that $\text{BERT}_{\text{Large}}$ outperforms XLNet in the YahooCQA and SemEval-2016 datasets, whereas in SemEval-2015 and SemEval-2017, $\text{XLNet}_{\text{Large}}$ outperforms BERT.

We observe new state-of-the-art results in all CQA datasets by fine-tuning both BERT and RoBERTa models. Though both Base and Large versions of BERT and RoBERTa provide state-of-the-art results across all CQA datasets, we find that the Large versions outperform their respective Base versions in all of them. For the SemEval datasets, our best performing $\text{RoBERTa}_{\text{Large}}$ model has an improvement of 10.86% in SemEval-2016CQA and 6.67% in SemEval-2017CQA in terms of MAP than the state-of-the-art models [139, 109], respectively. For the YahooCQA dataset, the $\text{RoBERTa}_{\text{Large}}$ model again performs the best with an improvement of 19.23% in terms of MRR over the previous state-of-the-art result [146].

Comparing Fine-tuning with Feature Extraction: To analyze the performance difference between the fine-tuning approach and the feature-based approach

in more detail, we conduct significance tests. We notice based on the paired t-test ($p \leq .05$) that all the fine-tuning-based approaches perform **significantly** better than the feature-based approaches. It is worth noting that in our feature-based approach, features are first extracted from the pre-trained model and then they are fed to the transformer encoder which is required to be trained from scratch. In contrast, in the fine-tuning approach, the pre-trained model is fine-tuned for a specific task by adding some additional randomly initialized parameters. As Peters et al. [120] suggest, the performance of the fine-tuning-based approach and the feature-based approach depends on the similarity between the pre-training and the target task. They also observed that fine-tuning the BERT model significantly outperformed the feature-based approaches for the textual similarity task. This may explain why fine-tuning approach performs better than the feature-based approach for the answer sentence selection task.

3.4.3 Ablation Studies

In order to further investigate the effectiveness of our proposed approaches, we conduct several ablation studies. Below, we first discuss the ablation test result on the feature-based approach, followed by discussing the result on the fine-tuning-based approach.

Effects on feature-based approach: In order to investigate the effectiveness of integrating transformer encoder with contextual embeddings in the feature-based approach, the following models are included in our ablation test:

- ELMo Embeddings: Only the feature-based ELMo Embeddings were used without any Transformer Encoder.
- $\text{BERT}_{\text{Base}}$ Embeddings: Only the feature-based $\text{BERT}_{\text{Base}}$ Embeddings were used without any Transformer Encoder.
- $\text{BERT}_{\text{Large}}$ Embeddings: Only the feature-based $\text{BERT}_{\text{Large}}$ Embeddings were used without any Transformer Encoder.

The above models simply measure the similarity between the question and the candidate answers based on the fixed contextualized embeddings generated from ELMo and BERT without sending them to the transformer encoder. We compare these models with our feature-based models that do include the transformer encoder. The results of our ablation study based on the average MAP and MRR scores across all datasets are given in Figure 3.4. From the ablation test, we find that integrating transformer encoder with contextual embeddings improves the performance by 43.73%, 25.23%, and 26.34% in terms of MAP and 41.32%, 21.27%, and 24.08% in terms of MRR in ELMo, $\text{BERT}_{\text{Base}}$, and $\text{BERT}_{\text{Large}}$ respectively. These improvements are **statistically significant** based on paired t-test

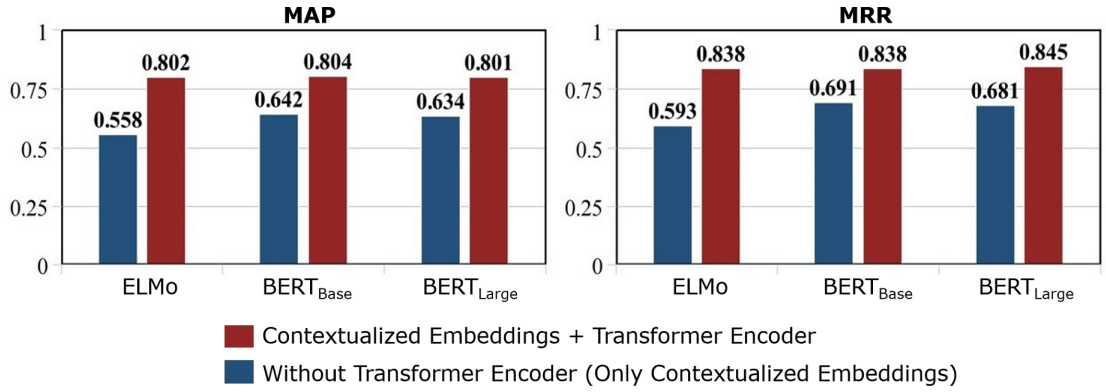


Figure 3.4: Performance comparisons based on the ablation test. Only the fixed contextual embeddings of each model (ELMo, $\text{BERT}_{\text{Base}}$, and $\text{BERT}_{\text{Large}}$) are compared with their integration with the Transformer Encoder. MAP and MRR of each model are based on the average across all datasets.

($p \leq .05$). This shows the effectiveness of our proposed approach of integrating transformer encoder with the features extracted from the BERT or ELMo models.

Effects on fine-tuning-based approach: To investigate the effectiveness of our approach of fine-tuning pre-trained transformer models, we excluded fine-tuning and ran experiments by utilizing only the feature-based embeddings generated from the pre-trained model. For that purpose, we selected the $\text{BERT}_{\text{Large}}$ model and studied the effect of removing fine-tuning from this model (see Table 3.8) in following four datasets for performance comparisons: TREC-QA (RAW), WikiQA, YahooCQA, and SemEval-2016CQA.

Table 3.8: Performance comparisons based on the Ablation Test. Here, 'FT' denotes 'Fine Tuning' and 'L' denotes 'Large'.

Model	QA datasets				CQA datasets			
	TREC-QA		WikiQA		YahooCQA		SemEval'16	
	MAP	MRR	MAP	MRR	MAP	MRR	MAP	MRR
BERT_{L} FT	0.92	0.95	0.84	0.86	0.95	0.95	0.87	0.93
***without* FT**	0.41	0.48	0.57	0.57	0.44	0.44	0.60	0.67

From Table 3.8, we find that removing fine-tuning from BERT decreases the performance by 55.4%, 32.1%, 53.7%, and 31.0% in terms of MAP in the TREC-QA, WikiQA, YahooCQA, and SemEval-2016CQA datasets respectively. Note that the deterioration here without fine-tuning is **statistically significant** based on paired t-test ($p \leq .05$).

3.4.4 Summary

In this chapter, we present two approaches to utilize contextualized embeddings with the transformer encoder for the answer selection and ranking task. Our experiments on six datasets demonstrate that the performance of our feature-based approach is comparable with most of the prior work. Moreover, we find that our approach of fine-tuning the pre-trained transformer encoder models for answer

sentence selection is more effective compared to the feature-based approach. We also show that our fine-tuning-based models are effective for the answer selection and ranking task even without leveraging transfer learning from large question-answering corpora. More importantly, we observe new state-of-the-art results on all six datasets in terms of the MRR metric using our proposed fine-tuning-based RoBERTa model along with setting new state-of-the-art results with the same model on five datasets in terms of the MAP metric.

4 Query Focused Text Summarization Task

In the *Query Focused Text Summarization Task*, given a query along with a set of document(s), the goal is to generate a summary from the source document(s) that is relevant to the given query. This task can be done in two scenarios:

1. Query Focused Single-Document Summarization: where a query along with a document are given, and the objective is to generate a query focused summary from the source document.

2. Query Focused Multi-Document Summarization: where a query along with a set of documents are given, and the objective is to generate a query focused summary from all the documents which are given in the document set.

In this thesis, we generate abstractive summaries for the query focused summarization task in both single-document and multi-document scenarios. Here, we divide the chapter into two sections, where we first discuss the Single-Document Query Focused Abstractive Summarization (SD-QFAS) task followed by the Multi-Document Query Focused Abstractive Summarization (MD-QFAS) task.

4.1 Single-Document Query Focused Abstractive Summarization

4.1.1 Background

In the abstractive text summarization task, the generated summaries may contain words or phrases that did not appear in the source document(s) [168]. Recently, various neural encoder-decoder models have provided state-of-the-art performance in a wide range of natural language generation tasks [172, 168]. The impressive success of using neural models for sequence to sequence modeling in such tasks have also inspired researchers to utilize the neural encoder-decoder architecture for abstractive summary generation in recent years [110, 132]. However, one major problem in the neural models for abstractive summarization is that while generating the summaries they tend to repeat the same word multiple times that lead to the generation of non-cohesive summaries [138]. To address this issue, See et al. [138] proposed the Pointer Generation Network (PGN) that utilized a novel copy and coverage mechanism to discourage the repetition of the same words. More recently, the BERTSUM [92] model was proposed which used the BERT model [36] as the encoder and the decoder of transformer [150] as the decoder. It is to be noted that the BERTSUM model showed impressive performance for the abstractive summarization task and set new state-of-the-art results in several datasets.

While significant progress has been made on the single-document generic abstractive summarization task, applying neural models for the query focused abstractive summarization task in the single-document scenario has been rare [10]. One notable exception on utilizing neural models for such tasks is the Diversity Driven Attention (DDA) model [113]. This model can effectively generate query focused abstractive summaries via focusing on different portions of a document based on the given query at different times. However, similar to the less amount of work for the SD-QFAS task, the number of datasets available for this task is also very small. To the best of our knowledge, the only available dataset for such tasks is the Debatepedia dataset[27]. Nonetheless, the size of this dataset is very small compared to the datasets used for generic abstractive summarization [10, 92, 138]. Thus, the lack of large training data for the SD-QFAS task in the available dataset makes this task a few-shot learning problem. To address this issue, the Relevance Sensitive Attention (RSA) for Query Focused Summarization[10] utilized transfer learning by first pre-training the PGN model [138] on a large generic abstractive summarization dataset and then incorporated query relevance into the pre-trained model to generate the query focused abstractive summaries in the Debatepedia dataset. However, they did not fine-tune their model on QFAS datasets and obtained a very low Precision score [10].

[27]http://www.debatepedia.org/

To tackle the above issues, we introduce a novel transfer learning approach for the SD-QFAS task by first pre-training our summarization model on a large generic abstractive summarization dataset followed by fine-tuning it for the SD-QFAS task via incorporating query relevance. In contrast to the prior work [113, 10, 2, 61] that are based on the recurrent neural network architecture, we utilize the transformer architecture to leverage the effectiveness of pre-training and fine-tuning since the former and the latter with the transformer-based models have been found more effective on various natural language processing tasks [36, 83, 43].

4.1.2 Our Proposed Approach

Let us assume that we have a query $Q = q_1, q_2, ..., q_k$ containing k words and a source document $D = d_1, d_2, ...d_n$ containing n words. Our task is to generate an abstractive summary $S = s_1, s_2, ...s_m$ containing m words from the source document D based on the given query Q.

To achieve this goal, our proposed method adopts the BERTSUM model [92] that utilizes the transformer architecture for abstractive summarization via utilizing the BERT model [36] on its encoder and the decoder of transformer [150] on its decoder. However, the BERTSUM model was designed for the generic summarization task without considering any query relevance [92]. Therefore, we incorporate query relevance (QR) in the BERTSUM model along with leveraging transfer learn-

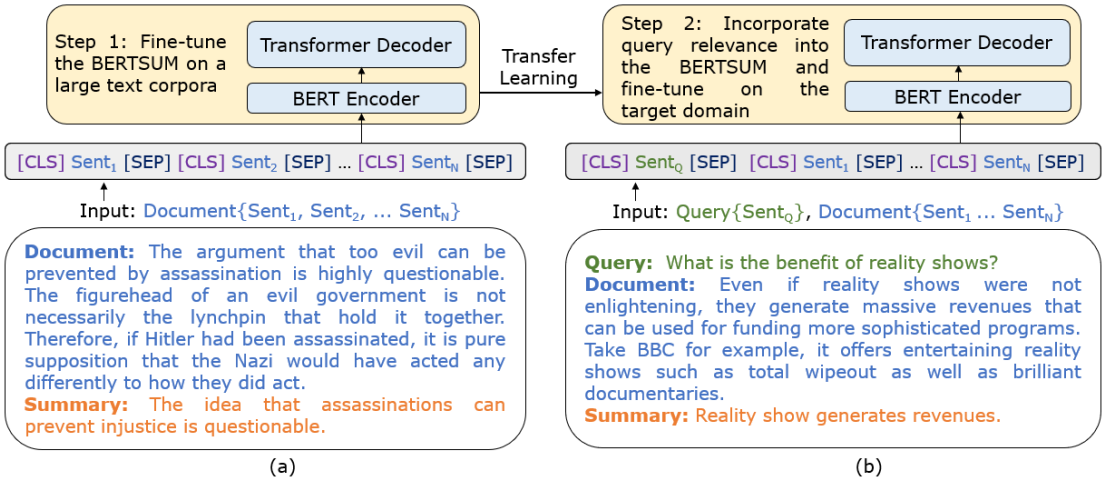


Figure 4.1: Our proposed approach works in two steps: (a) Pre-train the BERTSUM model on a generic abstractive summarization corpus (e.g., XSUM) and (b) Fine-tune the pre-trained model for the SD-QFAS task on the target domain (e.g., Debatepedia).

ing (TL) to utilize this model for few-shot learning in the SD-QFAS task. More specifically, our model (denoted as QR-BERTSUM-TL) performs the SD-QFAS task in two steps as shown in Figure 4.1. In the first step, we pre-train the BERTSUM model on a large training corpus of generic abstractive summarization. Then, we fine-tune the pre-trained model for the SD-QFAS task by utilizing the query relevance. In the following, we describe these two steps in detail.

(i) Pre-training the BERTSUM Model: In this step, we pre-train the BERTSUM model on a large generic abstractive summarization dataset. Among

the datasets used for BERTSUM [92], the XSUM[28] [111] dataset was the most abstractive one containing highest number of novel bi-gram. Therefore, we pre-train the BERTSUM model on this dataset. During the training process, the model utilizes the pre-trained BERT model [36] as the encoder and the randomly initialized Transformer decoder [150] as the decoder. However, the original BERT model inserted the special token [CLS] at the beginning of only the first sentence. In contrast, the BERTSUM model inserts the [CLS] token at the beginning of each sentence. Moreover, each sentence-pair in BERTSUM is also separeted by the [SEP] token.

(ii) Incorporating Query Relevance and Fine-tuning BERTSUM: In this step, we fine-tune the BERTSUM model on the Debatepedia dataset which was pre-trained on the XSUM dataset in the previous step. During fine-tuning, we incorporate the query relevance via concatenating the query with the document as the input of the encoder (see Figure 4.1b). We do this because we find that a similar approach of concatenating the question with the document works well for different question-answering tasks [83]. Furthermore, we use different types of attention mechanisms to utilize the incorporation of query relevance in BERTSUM. These attention mechanisms are described in the following.

[28]https://github.com/EdinburghNLP/XSum/tree/master/XSum-Dataset

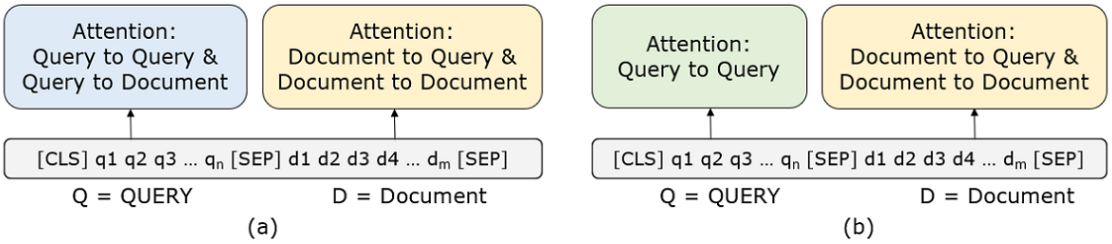


Figure 4.2: An overview of various attention models. (a) The Bidirectional Self-Attention Mechanism. (b) The Query-Document Attention Mechanism.

Attention Mechanisms in QR-BERTSUM-TL: We utilize two types of attention mechanisms to utilize the query relevance in our proposed QR-BERTSUM-TL model. These are: (i) the Bidirectional Self-Attention mechanism, and (ii) the Query-Document attention mechanism. These two attention mechanisms are shown in Figure 4.2. Below, we describe these attentions.

(i) The Bidirectional Self-Attention Mechanism: In the QR-BERTSUM-TL architecture, we concatenate the query with the document and then the concatenated text is given as input to the encoder of the model. This is how we incorporate the query relevance in the original BERTSUM model [92]. Note that the original BERTSUM architecture uses the BERT model as its encoder [36] that utilizes the bidirectional self-attention mechanism [150] to generate the encoded representation. When we utilize the bidirectional self-attention mechanism [36] (Figure 4.2a) in the QR-BERTSUM-TL model, both the query and the document gives attention to

each other to provide the encoded representation of the concatenated input.

(ii) The Query-Document Attention Mechanism: Dong et al. [37] proposed the sequence-to-sequence language modeling objective for text sequences that are consisted of two segments. In such text sequences, each token in the first segment can only attend to the tokens in both directions within the same segment but cannot attend to any tokens in the second segment, while the tokens in the second segment can attend to the leftward tokens in their own segment as well as to all tokens in the first segment. Based on the this objective, we propose the Query-Document (QD) attention mechanism. In our QD attention, each token in the query can only attend to the tokens which are within the query. While the tokens in the document can attend to all tokens in both query and document bidirectionally. Our intuition behind this approach is that, since in the original QR-BERTSUM-TL model the bidirectional self-attention allowed the query to also attend to the document, the final encoded representation of the concatenated input might lose some query related information. Because the bidirectional self-attention let the query segment to also be influenced by the document segment to produce the encoded representation of the query segment. Thus, we hypothesize that during the decoding process, the decoder may produce summaries that may not be fully relevant to the query. To avoid such scenarios, we allow the query segment to only attend to itself whereas the document segment is allowed to provide a query

focused representation by attending to both the query and to itself. Given query, key, and value vectors $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ respectively, with $\mathbf{d}_k$ as the square root of the dimension of $\mathbf{K}$, we calculate the encoded representation $\mathbf{Z}$ using QD attention via adding the mask matrix M in the self-attention formula of the transformer encoder [150]:

$$\mathrm{Z} = softmax\left(\frac{\mathrm{Q} \times \mathrm{K}^{\mathrm{T}}}{\sqrt{\mathrm{d}_k}} + M\right)\mathrm{V} \tag{4.1}$$

In equation (4.1), $M_{ij} = 0$ allows attention from token i to token j, whereas $M_{ij} = -\infty$ prevents attention from token i to token j.

4.1.3 Experimental Setup

In this section, we describe the datasets that we used to evaluate the effectiveness of our proposed approach, followed by the evaluation metrics, the training parameters that had been used in our experiments, and the details of our implementation.

4.1.3.1 Datasets

In our experiments, we primarily used the Debatepedia [113] dataset to evaluate our proposed approach for the SD-QFAS task. Moreover, due to the lack of datasets available for the SD-QFAS task, we also use the QA-NLG dataset from MS-MARCO [6] and utilize it for the SD-QFAS task to further investigate the generalized effectiveness of our proposed approach across different datasets.

Debatepedia Dataset: Debatepedia is an encyclopedia of pro and con arguments and quotes on debate topics. Nema et al. [113] utilized Debatepedia to create a dataset for the SD-QFAS task [113]. The average number of words per document, summary, and query in the Debatepedia dataset is 66.4, 11.16, and 9.97 respectively. They used 10-fold cross validation in their experiments with the DDA model on this dataset. The average number of instances in each fold is 10,859 for training, 1,357 for testing, and 1,357 for validation respectively. It is to be noted that, we find in the source code[29] of the DDA [113] model that the dataset was augmented to create new training instances while it was evaluated using the DDA model [113]. However, the data augmentation approach was not mentioned in the original paper of the DDA model where this dataset was first introduced for the SD-QFAS task [113]. Based on our analysis of the source code of the DDA model, we find that in the augmented dataset, the test data and the validation data were same as the original, but the average training instances in each fold were 95,843. For data augmentation, a pre-defined vocabulary of 24,822 words was used where each word had been associated with a synonym. Then for each training instance, N ($10 \leq N \leq 17$) words in each document and M ($1 \leq M \leq 3$) words in each query were randomly selected (except stop words and numerical values) and then replaced with their synonyms found in the vocabulary. If a selected word was not

[29]https://git.io/JeBZX

found in the vocabulary, it was added there with the most similar word found based on cosine similarity in the GloVe [118] vocabulary. For each training instance, this process is repeated 8 times to create 8 new document and query instances. But the same summary of the original instance was used in the newly generated instances. Note that we did not leverage any data augmentation for our proposed model. Instead, we used the original Debatepedia dataset for evaluation and only pre-processed it by removing the start token <s> and the end token <eos>.

MS-MARCO: Due to the lack of datasets available for the SD-QFAS task, we utilize the QA-NLG dataset from MS-MARCO [6] for such tasks. However, in this dataset, a set of passages along with a query are given and the abstractive answer is required to be generated from the most relevant passage among them. To utilize this dataset for QFAS, we follow the work of Nishida et al. [114], where they only utilized the gold passages in the training set as well as in the development set to evaluate their model in one of their experiments. We use this dataset similarly for the SD-QFAS task by only utilizing the gold passage as the source document. In the MS-MARCO dataset, the training set contains 153725 queries which is much higher than the Debatepedia dataset. Thus, we did not use this dataset to study the effectiveness of our proposed model for few-shot learning. Rather, we use this dataset to investigate the generalized effectiveness of our proposed model for the SD-

QFAS task. Moreover, similar to the prior work [114], we use the development set of this dataset that contains 12467 queries as the evaluation data. During experiments, we use 10% data from the training set for validation and select the model for evaluation in the development set which performs the best in the validation set.

4.1.3.2 Evaluation Metrics

To evaluate the performance of our proposed model in the Debatepedia dataset as well as in the MS-MARCO dataset, we follow the prior works that utilized these datasets to select the evaluation metrics [113, 10, 114].

For the Debatepedia dataset, we report the results based on the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metric in terms of the ROUGE-1, ROUGE-2, and ROUGE-L scores[30]. Though the prior works on the Debatepdia dataset only addressed the ROUGE scores in terms of the Recall metric [113, 10], in this work, we also included the Precision and F1 metrics in addition to the Recall metric to address the ROUGE scores. Similar to the prior work, we calculated the result based on the average across 10-folds. When the ROUGE score is calculated based on Recall, it focuses to determine how much of the reference summary is covered by the model's predicted summary. If the number of overlapping n-grams between the reference summary and the model's predicted summary is ON and the

[30]We used the following package for calculation: https://pypi.org/project/pyrouge/

total number of n-grams in the reference summary is TN_R, then the ROUGE-N (Recall) for n-grams overlap calculated as follows:

$$\text{ROUGE-n (Recall)} = \frac{ON}{TN_R} \tag{4.2}$$

When the ROUGE score is calculated based on Precision, it focuses to determine how much of the model's predicted summary is relevant. If the number of overlapping n-grams between the reference summary and the model's predicted summary is ON and the total number of n-grams in the model's predicted summary is TN_M, then the ROUGE-N (Precision) for n-grams is calculated as follows:

$$\text{ROUGE-n (Precision)} = \frac{ON}{TN_M} \tag{4.3}$$

For the MS-MARCO dataset, it should be noted that the prior work on this dataset used the Bilingual Evaluation Understudy (BLEU) metric based on unigrams along with utilizing the ROUGE-L metric for performance evaluation [114]. Thus, similar to the prior work [114], we also use these two metrics in terms of the F1 score to evaluate our proposed models in the MS-MARCO dataset. Below, we briefly discuss the Recall, Precision, and F1 metrics. Then, we discuss various ROUGE scores followed by the BLEU-1 metric that we use for the SD-QFAS task in different datasets.

Recall: It is is used to measure the performance of a model in terms of identifying what percentage of the positive examples in a dataset that the model has

successfully labeled as positive. If we denote the total number of examples that the model has correctly predicted as positive as TP and the total number of examples that the model has incorrectly predicted as negative as FN, then the recall will be calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{4.4}$$

Precision: It is is used to measure the performance of a model in terms of identifying the percentage of examples that the model has labeled as positive are actually positive. If we denote the total number of examples that the model has correctly predicted as positive as TP and the total number of examples that the model has incorrectly predicted as positive as FP, then the precision will be calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{4.5}$$

F1: It is the harmonic mean of precision and recall which is calculated as follows:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.6}$$

ROUGE-1: It uses the overlap of the unigrams between the summary produced by the model and the gold reference summary.

ROUGE-2: It uses the overlap of the bigrams between the summary produced by the model and the gold reference summary.

ROUGE-L: It utilizes the overlap of the n-grams of the Longest Common Subsequence (LCS) between the summary produced by the model and the gold reference summary. It is to be noted that instead of using any pre-defined length of the n-grams, the ROUGE-L automatically calculates the longest common n-grams to compute the score.

BLEU-1: BLEU [115] utilizes a modified form of the precision metric to evaluate the performance of the candidate summary generated by a model. Given a set of gold reference summaries for each candidate summary, the modified precision mechanism works as follows:

1. At first, for each distinct word in the generated candidate summary, the maximum number of times the word appears in any gold reference summaries is identified.

2. Afterwards, the total number of times each distinct word appears in the generated candidate summary is counted which is then clipped by the maximum value counted for that word in the previous step. For each word w, if the total count for w in the candidate summary is t_w and the maximum value found in the gold reference summaries for w is m_w, then the clipped value c_w

is counted as follows:

$$c_w = min(m_w, t_w) \tag{4.7}$$

3. Finally, the BLEU-1 score is calculated via dividing the sum of the clipped values C_W by the total number of words generated in the candidate summary T_W.

$$\text{BLEU-1} = \frac{C_W}{T_W} \tag{4.8}$$

4.1.3.3 Training and Parameter Settings

Here, we discuss the training parameters that we used in our experiments for the SD-QFAS task. To pre-train the BERTSUM model on the XSUM dataset, we kept the parameters similar to the original work [92]: dropout = 0.1, label smoothing with smoothing factor = 0.1, the hidden units in the transformer decoder = 768 and the hidden size for all feed-forward layers = 2048, the warmup_steps for the encoder = 20000 and for the decoder = 10000, the learning_rate for the encoder = 0.002 and for the decoder = 0.1, the batch size = 140 with total training steps = 30000. To fine-tune the QR-BERTSUM-TL model on the target dataset, we set new values to the following parameters[31]: batch size = 500, warmup_steps_encoder = 6000, warmup_steps_decoder = 2000, and total training steps = 60000. Moreover, we truncated each input document to 100 tokens and each generated summary to

[31] We also use these values for the baseline QR-BERTSUM$_{\text{Vanilla}}$ model.

consider at most 25 tokens. As used in the original BERTSUM model [92], we also utilized the beam search decoding mechanism with size = 5.

4.1.3.4 Implementation

For implementation, we utilize the Transformer Library of HuggingFace [163] along with using the official source code of the BERTSUM[32] model [92]. All of our experiments were run using NVIDIA V100 with 4 GPUs.

4.1.4 Results and Discussions

To evaluate the effectiveness of our approach, we consider the following models as baselines:

QR-BERTSUM$_{\text{Vanilla}}$: This model adopted the BERTSUM architecture [92] and incorporated Query Relevance (QR) by concatenating the query with the document. We trained it end-to-end only on the target SD-QFAS dataset.

BERTSUM$_{\text{XSUM}}$: This baseline used the BERTSUM model pre-trained on the XSUM dataset and did not do any fine-tuning on the target dataset.

Below, we first discuss the performance of our model in the Debatepedia dataset. Then we discuss the performance of our model in the MS-MARCO dataset.

[32]https://github.com/nlpyang/PreSumm

Table 4.1: Performance of different models for the SD-QFAS task on the Debatepedia dataset. Here, '*' denotes 'results based on our experiments with DDA'. 'R', 'P', and 'F' denote 'Recall', 'Precision', and 'F1' respectively. The 'Original' and 'Augmented' versions of DDA are denoted by 'ORG' and 'AUG' respectively, while '†' denotes 'QD attention' and '§' denotes 'Bidirectional Self-Attention'.

MODEL	**ROUGE-1**			**ROUGE-2**			**ROUGE-L**		
	R	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**
QR-BERTSUM$_{\text{Vanilla}}$	22.3	35.7	26.4	9.9	16.7	11.9	21.2	33.9	25.1
BERTSUM$_{\text{XSUM}}$	17.4	11.5	13.3	3.0	2.5	2.8	15.0	10.0	11.5
DDA*(ORG)	7.5	7.7	7.4	2.8	2.9	2.8	7.1	7.5	7.2
DDA*(AUG)	37.8	47.4	40.5	27.6	33.7	29.4	37.3	46.7	39.9
DDA [113]	41.3	-	-	18.8	-	-	40.4	-	-
Selection Driven [2]	43.2	-	-	27.4	-	-	42.7	-	-
Overlap-Wind [61]	44.4	-	-	30.5	-	-	44.2	-	-
RSA [10]	53.1	-	-	16.1	-	-	46.2	-	-
QR-BERTSUM-TL †	**58.0**	60.3	**58.7**	**45.2**	**46.1**	**45.5**	**57.1**	59.2	**57.7**
QR-BERTSUM-TL §	**58.0**	**60.4**	58.5	**45.2**	**46.1**	**45.5**	**57.1**	**59.3**	**57.7**

Performance on Debatepedia: For the Debatepedia dataset, in addition to the baselines, we also compare the performance of our proposed model with some

of the previous models proposed for this dataset: the first model proposed for this dataset, the DDA model [113]; the current state-of-the-art in terms of ROUGE-1 and ROUGE-L, the RSA model [10]; the current state-of-the-art in terms of ROUGE-2, the Overlap-Wind model [61]; and the recently proposed Selection Driven model [2]. Moreover, we also ran our own experiments with the DDA model on both the original and augmented versions of the Debatepedia dataset. The experimental results of our proposed approach and other models are shown in Table 4.1. To be noted that, The QR-BERTSUM$_{\text{Vanilla}}$ model shown in Table 4.1 was trained end-to-end on the original version of the Debatepedia Dataset.

We find that among the baseline models, both the QR-BERTSUM$_{\text{Vanilla}}$ and the BERTSUM$_{\text{XSUM}}$ models outperform the DDA*(ORG) model for few-shot learning. Moreover, since the QR-BERTSUM$_{\text{Vanilla}}$ and the BERTSUM$_{\text{XSUM}}$ are based on the transformer architecture, these models outperforming the RNN based DDA*(ORG) model suggests the effectiveness of using transformer instead of RNN for the SD-QFAS task. We also find that data augmentation **significantly** (based on paired t-test with $p \leq .05$) improves the performance of DDA, with the DDA*(AUG) model outperforming all baselines. As our result with DDA*(AUG) could not fully reproduce the result in [113], we assume that different pre-processing settings for the input document as well as the generated summary could be the possible reason behind this since Nema et al. [113] did not mention their pre-processing techniques.

When we compare our proposed QR-BERTSUM-TL model with the baselines, we find that the QR-BERTSUM-TL model with both attentions significantly improved the performance over the QR-BERTSUM$_{\text{Vanilla}}$ model (which did not leverage transfer learning) as well as the BERTSUM$_{\text{XSUM}}$ model (which did not utilize fine-tuning). These improvements suggest the effectiveness of utilizing both transfer learning and fine-tuning in the QR-BERTSUM-TL model.

While we compare the performance between different attentions in the QR-BERTSUM-TL model, we observe that both attentions provide the exact same result in most ROUGE scores with only a few exceptions. Based on the result, we find that the QR-BERTSUM-TL model with the bidirectional self-attention outperforms its QD attention counterpart in two cases in terms of the Precision metric, with an improvement of 0.17% for both ROUGE-1 and ROUGE-L scores. The only case when The QR-BERTSUM-TL model with the QD attention outperforms the QR-BERTSUM-TL with the bidirectional self-attention is based on the F1 metric in terms of the ROUGE-1 score, with an improvement of 0.34%. The overall result in the Debatepedia dataset suggests that the QD attention is not more effective than the bidirectional self-attention for the SD-QFAS task.

In comparison to the prior work, we observe that the proposed QR-BERTSUM-TL model sets a new state-of-the-art result in all three ROUGE scores for both attentions. More specifically, in terms of recall, we find that the QR-BERTSUM-

TL model (for both attentions) has an improvement of 9.23%, and 23.59% in terms of ROUGE-1, and ROUGE-L respectively, over the previous state-of-the-art RSA model [10]. As mentioned in [10], the RSA model provided very low ROUGE precision score (the authors did not state the exact score) by generating very large summaries which are 10 times longer than the required length. In contrast, our proposed model shows high precision score by effectively generating summaries according to the required length. We also observe a huge gain in comparison to the previous models based on the ROUGE-2 score, with an improvement of 140.43%, 180.75%, 64.96%, 48.20% over the DDA [113], RSA [10], Selection Driven [2], and Overlap-Wind [61] models respectively in terms of the recall metric.

Performance on MS-MARCO: For the MS-MARCO dataset, in addition to the baselines[33], we compared our proposed model with the current state-of-the-art in this dataset: the MASQUE model [114]. We show the result in Table 4.2.

From Table 4.2, we observe that our proposed QR-BERTSUM-TL model (for both attentions) again outperforms all the baseline models. More specifically, we find that our best performing QR-BERTSUM-TL using the bidirectional self-attention outperforms the baseline QR-BERTSUM$_{\text{Vanilla}}$ with an improvement of 9.50% in terms of ROUGE-L and 14.53% in terms of BLEU-1. Further to note

[33]The QR-BERTSUM$_{\text{Vanilla}}$ model in Table 4.2 was trained end-to-end on the MS-MARCO dataset.

Table 4.2: Performance of different models for the SD-QFAS task on the MS-MARCO dataset in terms of ROUGE-L and BLEU-1 based on the F1 metric. Here, '†' denotes 'QD attention' and '§' denotes 'Bidirectional Self-Attention'.

MODEL	ROUGE-L	BLEU-1
QR-BERTSUM$_{\text{Vanilla}}$	71.6	70.2
BERTSUM$_{\text{XSUM}}$	20.1	21.5
MASQUE [113]	**78.7**	78.1
QR-BERTSUM-TL †	72.3	72.1
QR-BERTSUM-TL §	78.4	**80.4**

that the improvement with this model is much higher while comparing with the other baseline: the BERTSUM$_{\text{XSUM}}$ model, where the performance improvement is 290.05% in terms of ROUGE-L and 273.95% in terms of BLEU-1. These improvements demonstrate the effectiveness of utilizing both pre-training and fine-tuning in our proposed model.

Though previously we found in the Debatepedia dataset that the performance of both the QD attention and the bidirectional self-attention was almost similar, we find in the MS-MARCO dataset that the proposed QD attention is much less effective than the bidirectional self-attention mechanism. More specifically, we find that the performance is deteriorated by 7.78% in terms of ROUGE-L and 10.32% in

terms of BLEU-1 when the QD attention is used instead of using the bidirectional self-attention.

In comparison to the prior work, we find that our proposed models (for both attentions) could not outperform the current state-of-the-art MASQUE [114] model in terms of the ROUGE-L score. Though the QR-BERTSUM-TL model with the QD attention fails to outperform the MASQUE model in terms of both ROUGE-L and BLEU-1 metrics, it outperforms the MASQUE [114] model with an improvement of 2.94% in terms of the BLEU-1 score when it utilizes the bidirectional self-attention.

Note that we find from Table 4.1 and Table 4.2 that when transfer learning is not utilized, the performance of QR-BERTSUM$_{\text{Vanilla}}$ model in the MS-MARCO dataset is much better than its performance in the Debatepedia dataset. This could be due to the fact that the total training instances (153725 examples) in the MS-MARCO dataset is almost 15 times higher than the total training instances (10859 examples) in the Debatepedia dataset. Nonetheless, the performance improvement via utilizing query incorporation and transfer learning to fine-tune the BERTSUM model in both datasets shows that our proposed QR-BERTSUM-TL model is very effective for the SD-QFAS task.

Since we observe inconsistent effects in different datasets when we utilize various types of attentions to incorporate the query relevance in the proposed QR-BERTSUM-TL model, we conduct ablation studies to further investigate the in-

Table 4.3: Ablation test results in terms of Recall on Debatepedia and F1 on MS-MARCO. Here, '†' denotes 'QD attention' and '§' denotes 'Bidirectional Self-Attention', while 'w/o' denotes 'without'. Moreover, we denote 'ROUGE' as 'R' and 'BLEU' as 'B'.

MODEL	**Datasets**				
	Debatepedia			**MS-MARCO**	
	R-1	**R-2**	**R-L**	**B-1**	**R-L**
QR-BERTSUM-TL '§'	57.96	45.20	57.05	80.39	78.39
QR-BERTSUM-TL '†'	57.97	45.21	57.06	72.10	72.25
w/o Query Relevance	56.82	44.66	56.07	66.21	61.50

corporation of query relevance in our proposed model. In the following, we discuss our findings from the ablation study.

Ablation Studies: In our ablation test, we remove the query incorporation from our proposed QR-BERTSUM-TL model. We show the result of our ablation test in Table 4.3 which suggests that the removal of query relevance leads to huge performance deterioration in the MS-MARCO dataset, which is **statistically significant** based on paired t-test ($p \leq .05$). However, we surprisingly find that the performance deterioration in the Debatepedia dataset is very small (less than 1%). For each at-

Table 4.4: An example from the Debatepedia dataset.

Query: Is self-defense a good reason for guns ownership?
Document: The supreme court case warren vs district of Columbia maintained that there is no way to police protection and there is no contracts between the individuals and local police. In short, the court has ruled that each person is responsible for his/her own protection.
Gold Summary: Individuals are responsible to defending themselves as cops are not.

tention, the performance deterioration in the Debatepedia dataset after the removal of query incorporation is **not statistically significant** based on paired t-test ($p \leq .05$). The discrepancy in performance deterioration between MS-MARCO and Debatepedia gives a strong indication that the queries in the Debatepedia dataset are not effective for summarization. In the following, we analyze the Debatepedia dataset to investigate the possible reasons behind the discrepancy in performance deterioration.

Analyzing the Debatepedia dataset: Due to the surprising performance in the Debatepedia dataset that we observe after removing the query relevance, we manually analyze the dataset to find out the possible reasons. Based on our analysis,

we find that many queries in this dataset are not relevant to the document as well as to the summary. Table 4.4 shows such an example from this dataset where the gold summary is more of a generic summary where the query has no relevance with both the document as well as the gold summary. We also find many examples where words in the query do not appear in the document, as well as the requirements for some queries are just yes/no type answers. Besides, we observe some examples where excluding queries that are relevant to the documents do not have any negative effects to generate the most appropriate summaries. These findings suggest that the queries in the Debatepedia dataset are not relevant to the generated summaries and this dataset is more of a generic summarization dataset.

4.1.5 Summary

In this section, we presented a transfer learning technique with the transformer-based BERTSUM model and utilized it for the SD-QFAS task via incorporating query relevance. Our approach shows state-of-the-art result in the Debatepedia dataset without the leverage of any data augmentation. This suggests that our model can overcome the lack of availability of large training data for the SD-QFAS task by effectively generating summaries with few-shot learning. Moreover, our experimental results in this dataset also suggest the effectiveness of using the transformer model instead of RNN for such tasks. Furthermore, we propose a novel QD

attention mechanism to incorporate query relevance in the BERT encoder for the SD-QFAS task and observe that our proposed attention performs on par with the original bidirectional self-attention used in BERT encoder.

In the larger sized MS-MARCO dataset, we again find that our proposed approach is effective to improve the performance. However, in this dataset we observe that utilizing the QD attention in our proposed QR-BERTSUM-TL model is much less effective than using the bidirectional self-attention in the same model. Due to observing different effects with the QD attention in the Debatepedia and the MS-MARCO datasets, we conduct an ablation test to investigate the incorporation of query relevance in the QR-BERTSUM-TL model. Based on the ablation test, we find that the removal of query relevance from the QR-BERTSUM-TL model significantly degrades the performance in the MS-MARCO dataset for both attentions. However, we surprisingly find that the removal of query relevance from this model could still provide identical result in the Debatepedia dataset when the query relevance is incorporated into the model. Based on further investigation, we find that the queries in the Debatepedia dataset have little to no effect in summary generation and the gold summaries in this dataset are quite generic.

4.2 Multi-Document Query Focused Abstractive Summarization

4.2.1 Background

Since the rise of the internet, accessing the ever-increasing amount of unstructured text has become a major problem for many users [168, 40]. Often, users require a readable summary generated from multiple sources to fulfill there information needs [40, 164, 71]. To this end, the query focused summarization task in the multi-document scenario focuses to summarize a set of documents while answering a given query. With the increasing popularity of virtual assistants in recent years, there is a growing interest to incorporate the abstractive summarization capability for response generation in such systems [114]. Thus, the importance of studying the Query Focused Multi-Document Abstractive Summarization (MD-QFAS) task to tackle these issues has been on the rise.

For the multi-document abstractive summarization task, most of the early works were focused on generic summarization [112, 42]. Whereas the amount of work for the MD-QFAS task had been very limited [168]. Note that the currently available multi-document summarization datasets (e.g., DUC 2005, 2006, 2007) do not contain any labeled training data. Thus, it is required to generate summaries from the source documents without any in-domain knowledge. To tackle

the lack of training data for the MD-QFAS task, most previous works were based on various unsupervised approaches that could only generate extractive summaries [158, 48, 153, 167, 179, 154, 98, 40]. To generate the abstractive summaries in such tasks, Baumel et al. [10] proposed a transfer learning technique that addressed the issue of no dedicated training data for the datasets available for such tasks. They adopted the Pointer Generation Network (PGN) [138] pre-trained for the generic abstractive summarization task in a large dataset to predict the query focused summaries in the target dataset via modifying the attention mechanism of the PGN model. However, their model failed to outperform the extractive approaches in terms of various ROUGE scores. It should be pointed out that utilizing the state-of-the-art neural summarization models [92] that leveraged supervised training is not applicable in these datasets due to the unavailability of the training data. Moreover, while using datasets similar to the target dataset as the training data, we find that these datasets only contain multi-document gold reference summaries. Thus, the state-of-the-art neural summarization models cannot be trained in such datasets since these models cannot consider long text sequences (i.e., multiple documents) as input at once due to the computational complexities [174, 12].

In the MD-QFAS task, another key challenge to address is the identification of the sentences from multiple documents which are relevant to the query [160]. Because there could be several irrelevant candidate sentences from different doc-

uments which are semantically similar with the relevant ones as well as with the query [10, 40]. To identify the sentences which are relevant to the query, various approaches such as similar word count [10] or Cross-Entropy Method [40] were utilized. Though neural models based on supervised training have significantly outperformed various non-neural models for the answer sentence selection task in recent years [43, 73], due to the absence of labeled data for the relevant sentences in the MD-QFAS datasets, neural models have not been effectively used for such tasks yet. Recently, Garg et al. [43] showed that neural models such as BERT or RoBERTa pre-trained in a large question answering dataset could effectively select answers in other similar datasets without any supervised training. More recently, such pre-trained answer sentence selection models were used by Xu and Lapata [164] for the MD-QFAS task. In their work, they utilized distant supervision from various question answering datasets using the fine-tuned BERT [36] model to filter out the irrelevant sentences from the documents. However, Baumel et al. [10] showed that filtering sentences as an early step could lead to performance deterioration for the MD-QFAS task.

To address the above issues, we propose a novel weakly supervised learning approach via utilizing distant supervision using pre-trained transformer-based models to generate weak reference summary of each single-document from the multi-document gold reference summaries. Moreover, we also propose a novel iterative

approach to address the computational issue to train neural models for the abstractive summarization task in multi-document scenarios [174, 12]. In addition, instead of directly applying pre-trained answer selection models to filter out sentences from the source document as an early step as used in [164], we applied it in the final stage to select the most relevant sentences from the generated query focused abstractive summary via utilizing our best performing model for answer sentence selection: the fine-tuned RoBERTa [96] model that we proposed in Chapter 3. In the following, our proposed approach is described in details.

4.2.2 Our Proposed Approach

Suppose, we have a query $Q = q_1, q_2, ..., q_k$ containing k words and a set of N documents $D = d_1, d_2, ..., d_N$. For the MD-QFAS task, the goal is to generate a summary $S = s_1, s_2, ...s_n$ for the query Q from D containing n words.

Figure 4.3 shows an overview of our proposed approach. Since the available datasets for the MD-QFAS task do not include any training data, we provide supervised training to our target dataset via utilizing other MD-QFAS datasets as our training data [10, 98, 40]. However, the available MD-QFAS datasets only contain the gold reference summaries generated by human experts from multiple documents and don't contain the gold reference summary of each individual document [10, 98, 40]. Due to the limitations of using neural models in long documents

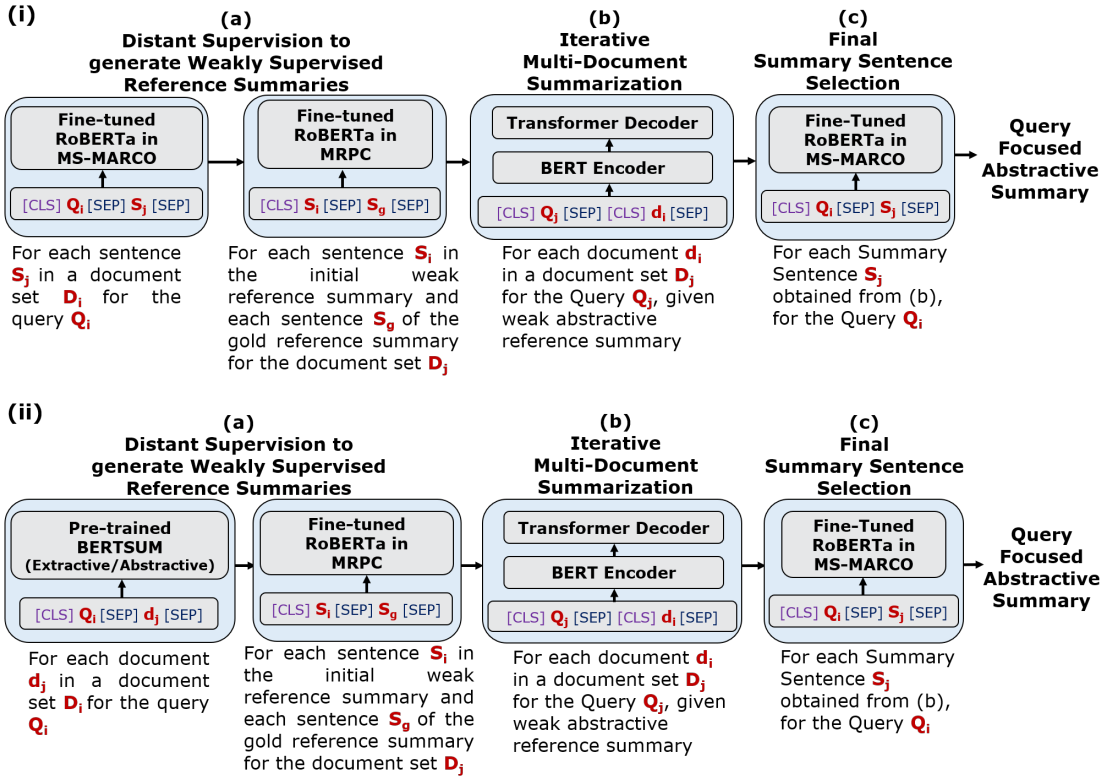


Figure 4.3: An overview of our model that uses the fine-tuned $\textbf{RoBERTa}_{\textbf{MS-MARCO}}$ in (i) and the pre-trained **BERTSUM** in (ii) to (a) generate the initial weak extractive reference summaries followed by utilizing the $\textbf{RoBERTa}_{\textbf{MRPC}}$ model for distant supervision to generate the weak abstractive reference summaries. Then, (b) the pre-trained **QR-BERTSUM-TL** model is fine-tuned to iteratively generate the query focused abstractive summaries which are then (c) ranked by the $\textbf{RoBERTa}_{\textbf{MS-MARCO}}$ model.

[92, 174], we propose an iterative approach with weakly supervised learning to train our model on each document in a document set. For that purpose, the weak reference summary of each document in a document set is at first generated via leveraging distant supervision from the multi-document gold reference summaries. Afterwards, the query focused abstractive summary of each document in a document set is generated using our iterative summarization approach. Finally, we rank the generated query focused summaries in a document set via utilizing a pre-trained answer selection model.

In the following, we first describe how we utilize weakly supervised learning with distant supervision. Then we discuss our proposed iterative approach to generate the query focused abstractive summary in multi-document scenarios using transformer-based models followed by describing how we select the most relevant sentences from the generated query focused summaries as our final summary.

4.2.2.1 Weakly Supervised Learning with Distant Supervision

We generate the weakly supervised reference summary of each document in a document set in two steps (see Figure 4.3a). In the first step, we utilize various pre-trained transformer-based models to generate the initial weak reference summary of each document. In the second step, we replace each sentence in the generated weak reference summary with each sentence in the multi-document gold reference

summaries via utilizing a RoBERTa-based fine-tuned sentence similarity model. For that purpose, we measure the similarity between each sentence in the multi-document gold reference summaries with each sentence in the generated weak reference summary. Then, based on the similarity score, we select the most relevant sentences from the gold reference summaries as the final weak reference summary for each document. Below, we describe these two steps in details:

(i) Initial Weak Reference Summary Generator: To generate the initial weak reference summary of each document in a document set, we utilize one of the following transformer-based models:

1. **Using Pre-trained BERTSUM Model:** Due to the impressive performance of utilizing the pre-trained BERTSUM model [92] in the SD-QFAS task described in the previous section, we also utilize this model in this step. First, we adopt the BERTSUM model (extractive or abstractive) pre-trained for the generic summarization task in CNN/DailyMail dataset [92, 52]. Then, we concatenate the query with each document in our training dataset and give as input to the pre-trained BERTSUM model (similar to our approach of concatenating the query with the document for the QR-BERTSUM-TL model). Afterwards, the pre-trained BERTSUM model generates the initial weak reference summary of each input document d_k in a document set.

2. **Using Fine-tuned RoBERTa Model:** We also utilize the RoBERTa model to generate the initial weak summary of each document. For this purpose, we adopt the fine-tuning-based RoBERTa model proposed for the answer sentence selection task in Chapter 3 for its impressive performance in several datasets for such tasks. To generate the weak reference summary of each document d_k, at first we fine-tune the RoBERTa model in the QA-ALL dataset of MS-MARCO [6] for the passage ranking (i.e., answer sentence selection) task. Then, the fine-tuned RoBERTa model measures the similarity score C between the given query Q_i and each sentence S_j in d_k. Based on the similarity score, we select the top 3 most relevant sentences as the weak reference summary since extracting only 3 sentences was found effective in different extractive summarizers such as the LEAD-3 baseline as well as the $\text{BERTSUM}_{\text{EXT}}$ model [92].

(ii) Final Weak Reference Summary Generator: We further provide distant supervision to manipulate the weak reference summary generated in the previous step by replacing each sentence in the weak reference summary with the most similar sentence found in the multi-document gold reference summaries. For this purpose, at first we adopt the RoBERTa model fine-tuned for the sentence similarity modeling task in the MRPC dataset [96]. Then for each document d_k in a document

set D_i, we utilize the fine-tuned RoBERTa$_{\text{MRPC}}$ model to measure the similarity between each sentence S_j in the weak reference summary and each sentence S_g in the gold reference summaries. Based on the similarity score, each sentence in the weak reference summary of a document is replaced with the most relevant sentence found in the multi-document gold reference summaries. Note that for a document d_k when a sentence S_g from the gold reference summaries is already used to replace a sentence S_j in the weak reference summary, then for the same document d_k we don't consider the sentence S_g again for replacement. Instead, we use the next most relevant sentence from the multi-document gold reference summaries for replacement. It should be pointed out that since the multi-document gold reference summaries are written by human annotators, the weak reference summaries generated in this step can be considered as weak abstractive reference summaries. In this way, we utilize distant supervision to generate the weak abstractive reference summary to train our model for the MD-QFAS task in the following step.

4.2.2.2 Iterative Fine-Tuning for Multi-Document Summarization

For the MD-QFAS task, we adopt the transformer-based [150] BERTSUM model pre-trained for generic summarization [92] to leverage the advantages of fine-tuning it for the query focused abstractive summarization task (as demonstrated in section 4.1 for the SD-QFAS task). However, the BERTSUM model was trained for the

single-document summarization task by considering at most 512 tokens [92]. Since the total number of tokens in a document set in multi-document scenarios could be much larger than 512 tokens [10, 40], and due to the computational complexity of training transformer-based models in long sequences [69, 12, 174, 26], we take an iterative approach (see Figure 4.3b). In our proposed iterative approach, we adopt the QR-BERTSUM-TL architecture proposed for the SD-QFAS task in section 4.1. To be noted that, the QR-BERTSUM-TL model used for the SD-QFAS task was at first pre-trained on the XSUM dataset for the generic summarization task. Since the generated summaries in the target dataset (e.g., Debatepedia) for the SD-QFAS task were smaller in size, the XSUM dataset was used as the summaries in this dataset were also short [92, 113]. But in the multi-document scenario, since the generated summaries are longer in size [10, 40], we did not use the XSUM dataset. Rather, we pre-train the model for generic summarization in the CNN/DailyMail dataset as this dataset requires longer summaries. Then, for each document in a document set, we fine-tune the pre-trained model using the weak abstractive reference summary to generate the query focused abstractive summary. Afterwards, we filter out some sentences from the generated summaries to select the most relevant sentences as the final summary in the next step.

4.2.2.3 Fine-Tuned RoBERTa for Summary Sentence Selection

In this stage, for each document set, all the sentences in the query focused abstractive summaries generated in the previous step are ranked using a fine-tuned RoBERTa model. For this purpose, we adopt the RoBERTa model fine-tuned for the answer selection task in the MS-MARCO dataset, which we also utilized for initial weak reference summary generation. The fine-tuned $\text{RoBERTa}_{\text{MS-MARCO}}$ model is then utilized to measure the relevance between each sentence S_i in the generated summary and the query Q_j for the document set D_j to select the sentences for the final summary that are most relevant to the query (see Figure 4.3c). It should also be pointed out that while selecting the most relevant sentences as the final query focused summary, we use the Trigram Blocking to reduce redundancy [116].

4.2.3 Experimental Setup

In this section, we first describe the datasets used for the MD-QFAS task. Then we discuss the evaluation metrics that we used to evaluate our model followed by the training parameters used in our experiments. Finally, we describe the details of our implementation.

4.2.3.1 Datasets

We use the DUC 2005, 2006, and 2007 datasets for the MD-QFAS task. The number of document sets were 50, 50, and 45 while the number of documents in each document set were 32, 25, and 25 in DUC 2005, 2006 and 2007 datasets respectively [40]. Each document set is associated with a topic statement (regarded as the query) and the objective is to generate a summary containing at most 250 words from the document set based on that query. Given the absence of the training data, to evaluate our model in each year's dataset we use the datasets from other two years for training. From each year's training data, we randomly selected 20% of the document sets for validation while the rest were used for training via utilizing weakly supervised learning.

4.2.3.2 Evaluation Metrics

Similar to the prior work [10, 40], we reported the results based on both recall and F1 metrics in terms of ROUGE-1, ROUGE-2, and ROUGE-SU4 scores [88] using the standard parameter setting[34]. Since we already describe the ROUGE-1 and ROUGE-2 scores alongside the recall and F1 metrics for the SD-QFAS task, below we only define the ROUGE-SU score.

ROUGE-SU: In ROUGE-SU, **S** stands for skip-bigram whereas **U** stands for

[34]ROUGE-1.5.5.pl -a -c 95 -m -n 2 -2 4 -u -p 0.5 -l 250

unigram. The skip-bigram is any pair of words in a sentence that maintains the sentence order but allows arbitrary gaps in between the two words. Thus, the ROUGE-SU score considers the overlaps of both the skip-bigrams as well as the unigrams between the summary produced by the model and the gold reference summary. Moreover, maximum skip distance between two words can also be set. As used in prior work [98, 10], we select the value of skip distance = 4 and evaluate the result based on the ROUGE-SU4 score.

4.2.3.3 Training and Parameter Settings:

To fine-tune the QR-BERTSUM-TL model for the MD-QFAS task, we kept most parameters similar to what we used for the SD-QFAS task in Section 4.1 and ran 50 steps for fine-tuning with batch size equal to 250. For RoBERTa, we fine-tune its pre-trained model for sentence similarity modeling using the same parameters that we utilized for the answer sentence selection task in chapter 3.

4.2.3.4 Implementation

For the RoBERTa model, we use its Large version [96] when we generate the initial weak reference summaries using this model as well as when the generated query focused abstractive summaries are ranked in the final step. For the implementation of RoBERTa, we use the Transformer library of HuggingFace [163]. When

the initial weak reference summaries are generated using BERTSUM, we experimented with both of its extractive and abstractive models. For weak extractive reference summary generation, we adopt the $\text{BERTSUM}_{\text{EXT}}$ model pre-trained on the CNN/DailyMail dataset for the generic extractive summarization task [92, 52]. For the weak abstractive reference summary generation using BERTSUM as well as for fine-tuning the QR-BERTSUM-TL model, we utilize the $\text{BERTSUM}_{\text{EXT-ABS}}$ model pre-trained on the CNN/DailyMail dataset for the generic abstractive summarization task [92, 52]. For the BERTSUM-based models, we use the same source code for implementation that we used for the SD-QFAS task. All of our experiments for the MD-QFAS task were run using NVIDIA V100 with 4 GPUs.

4.2.4 Results and Discussions

We now analyze the effectiveness of our approach by comparing with other models and also perform ablation test to investigate the performance of various methods used in our model. We denote our approach of using the **P**re-trained models (RoBERTa and BERTSUM) for **Q**uery focused **SUM**mary generation as PQSUM. As mentioned earlier, for the proposed $\text{PQSUM}_{\text{WSL-DS}}$ model, we experiment with various initial weak reference summary generation models:

- $\text{BERTSUM}_{\text{EXT}}$: We adopt the pre-trained $\text{BERTSUM}_{\text{EXT}}$ model from [92]. This model was trained for the generic extractive summarization task in the

CNN/DailyMail dataset [52].

- $\text{BERTSUM}_{\text{ABS-EXT}}$: We adopt the pre-trained $\text{BERTSUM}_{\text{ABS-EXT}}$ model from [92] which was trained for the generic abstractive summarization task (after being initially trained for extractive summarization) in the CNN/DailyMail dataset [52].

- $\text{RoBERTa}_{\text{MS-MARCO}}$: We adopt the pre-trained RoBERTa model [96] and fine-tuned it for the question-answer similarity task in the QA-NLG dataset of MS-MARCO [6].

Moreover, to analyze the effectiveness of using Weakly Supervised Learning with Distant Supervision (WSL-DS) in our proposed model $\mathbf{PQSUM_{WSL\text{-}DS}}$, we use two baselines that utilize the pre-trained BERTSUM model for zero-shot transfer learning [8] without utilizing weak supervision and fine-tuning. For each document, one baseline generates extractive summary: $\mathbf{PQSUM_{UNS\text{-}EXT}}$, while the other generates abstractive summary: $\mathbf{PQSUM_{UNS\text{-}ABS}}$. Similar to our proposed model, the generated summaries in both baselines are ranked using the RoBERTa model. Moreover, we compare our model with four recent works: i) CES-50 [40], ii) RSA [10], iii) Dual-CES [128], and iv) QUERYSUM [164]. Below, we give a brief description of these models.

- CES-50 [40]: The CES-50 is an extractive summarizer that utilizes the cross-

entropy method [130] to select a subset of sentences from the document(s).

- RSA [10]: It is an abstractive summarization model based on the pointer generation network [138]. In the RSA model [10], at first, the pointer generation network is pre-trained on a large generic summarization dataset for abstractive summary generation. Then to predict the abstractive summaries in the target dataset, all the documents in a document set are sorted based on their relevance with the query. Afterwards, for each document set the query relevance is incorporated in the pre-trained model via modifying the attention mechanism. Then the pre-trained model predicts the query focused abstractive summary of each document in a document set. This approach is continued until the model generates an abstractive summary that contains 250 tokens for each document set.

- Dual-CES [128]: This model is built on top of the CES model [40] and proposes a novel two-step dual-cascade optimization approach for extractive summarization.

- QUERYSUM [164]: This model proposes a coarse-to-fine modeling framework for extractive summarization. In this model, document clusters are given as input to the model and then three separate modules are utilized to estimate which segments are relevant to the query and likely to contain an answer

Table 4.5: Performance comparisons in terms of (a) **F1** and (b) **Recall** on the DUC 2005 dataset. Here, '*' denotes extractive summarization model, while '†', '∓', and '§' indicate that the initial weak reference summaries are generated by $\text{BERTSUM}_{\text{EXT}}$, $\text{BERTSUM}_{\text{ABS-EXT}}$, and $\text{RoBERTa}_{\text{MS-MARCO}}$ respectively. Moreover, we denote 'ROUGE' as 'R'.

Model	**DUC 2005**					
	F1			**Recall**		
	R-1	**R-2**	**R-SU4**	**R-1**	**R-2**	**R-SU4**
CES-50 [40] *	37.78	7.45	13.02	40.35	7.94	13.91
RSA [10]	-	-	-	39.82	6.98	**15.73**
Dual-CES [128] *	38.08	7.54	13.17	**40.82**	8.07	14.13
$\textbf{PQSUM}_{\textbf{EXT}}$ *	37.52	7.84	13.29	37.55	7.84	13.31
$\textbf{PQSUM}_{\textbf{ABS}}$	38.35	7.94	13.44	38.36	7.92	13.43
$\textbf{PQSUM}_{\textbf{WSL-DS}}$ †	40.13	8.94	14.53	40.16	8.94	14.54
$\textbf{PQSUM}_{\textbf{WSL-DS}}$ ∓	40.04	8.65	14.48	40.09	8.66	14.50
$\textbf{PQSUM}_{\textbf{WSL-DS}}$ §	**40.32**	**9.17**	**14.73**	40.36	**9.17**	14.74

to gradually filter out relevant sentences from the input document for final selection in the summary. For relevance measurement, this model also utilized the BERT model [36] pre-trained on question answering datasets.

Table 4.6: Performance comparisons in terms of (a) **F1** and (b) **Recall** on the DUC 2006 dataset. Here, '*' denotes extractive summarization model, while '†', '∓', and '§' indicate that the initial weak reference summaries are generated by $\text{BERTSUM}_{\text{EXT}}$, $\text{BERTSUM}_{\text{ABS-EXT}}$, and $\text{RoBERTa}_{\text{MS-MARCO}}$ respectively. Moreover, we denote 'ROUGE' as 'R'.

Model	DUC 2006					
	F1			Recall		
	R-1	R-2	R-SU4	R-1	R-2	R-SU4
CES-50 [40] *	40.47	9.13	14.73	43.01	9.69	15.65
RSA [10]	-	-	-	42.89	8.73	**17.75**
Dual-CES [128] *	41.23	9.47	14.97	**43.94**	10.09	15.96
QUERYSUM [164] *	41.6	9.5	15.3	-	-	-
$\text{PQSUM}_{\text{EXT}}$ *	40.68	9.29	14.66	40.41	9.22	14.56
$\text{PQSUM}_{\text{ABS}}$	40.87	9.43	14.83	40.59	9.39	14.73
$\text{PQSUM}_{\text{WSL-DS}}$ †	43.44	**10.94**	**16.46**	43.11	**10.85**	16.34
$\text{PQSUM}_{\text{WSL-DS}}$ ∓	42.48	10.55	16.02	41.96	10.39	15.79
$\text{PQSUM}_{\text{WSL-DS}}$ §	**43.49**	10.78	16.45	43.22	10.70	16.35

Performance Comparisons: The results of our experiments in the DUC 2005, DUC 2006, and DUC 2007 datasets are shown in Table 4.5, Table 4.6, and Table 4.7 respectively. From these tables, we find that in all datasets, the proposed

Table 4.7: Performance comparisons in terms of (a) **F1** and (b) **Recall** on the DUC 2007 dataset. Here, '*' denotes extractive summarization model, while '†', '∓', and '§' indicate that the initial weak reference summaries are generated by $\text{BERTSUM}_{\text{EXT}}$, $\text{BERTSUM}_{\text{ABS-EXT}}$, and $\text{RoBERTa}_{\text{MS-MARCO}}$ respectively. Moreover, we denote 'ROUGE' as 'R'.

Model	DUC 2007					
	F1			Recall		
	R-1	R-2	R-SU4	R-1	R-2	R-SU4
CES-50 [40] *	42.86	11.34	16.53	45.45	12.02	17.54
RSA [10]	-	-	-	43.92	10.13	**18.54**
Dual-CES [128] *	43.24	11.78	16.83	**46.02**	**12.53**	17.91
QUERYSUM [164] *	43.3	11.6	16.8	-	-	-
$\text{PQSUM}_{\text{EXT}}$ *	42.57	11.20	15.98	42.41	11.08	15.92
$\text{PQSUM}_{\text{ABS}}$	42.17	10.82	15.98	42.05	10.79	15.91
$\text{PQSUM}_{\text{WSL-DS}}$ †	44.29	11.89	17.24	44.11	11.84	17.16
$\text{PQSUM}_{\text{WSL-DS}}$ ∓	44.20	11.80	17.12	43.72	11.53	16.92
$\text{PQSUM}_{\text{WSL-DS}}$ §	**44.72**	**12.44**	**17.72**	44.61	12.40	17.66

$\text{PQSUM}_{\text{WSL-DS}}$ model outperform the baselines as well as the prior work in terms of the F1 metric for all ROUGE scores. To be noted that, these improvements in performance are observed with all initial weak reference summary generators for

the **PQSUM$_{\textbf{WSL-DS}}$** model.

More specifically, in the DUC 2005 dataset, our best performing PQSUM$_{\text{WSL-DS}}$ model with RoBERTa$_{\text{MS-MARCO}}$ as the initial weak reference summary generator outperforms the previous state-of-the-art Dual-CES [128] with an improvement of 5.88%, 21.62%, and 11.85% in terms of ROUGE-1, ROUGE-2, and ROUGE-SU4 scores respectively based on the F1 metric. However, in terms of the Recall metric, none of our models could outperform the prior state-of-the-art models based on the ROUGE-1 [128] and ROUGE-SU4 [10] scores. Nevertheless, all of our models outperform the prior work in ROUGE-2 (Recall) where the RoBERTa$_{\text{MS-MARCO}}$ model as the initial weak reference summary generator sets a new state-of-the-art with an improvement of 13.63% from the previous state-of-the-art Dual-CES [128].

In the DUC 2006 dataset, we again observe that our proposed models outperform all the prior work in terms of the F1 metric for all ROUGE scores, as well as in terms of the ROUGE-2 score based on the Recall metric. However, for both Recall and F1, we find in this dataset that in terms of the ROUGE-2 score, our proposed model with BERTSUM$_{\text{EXT}}$ as the initial weak reference summary generator is the most effective model; whereas for the ROUGE-1 score, our model with RoBERTa$_{\text{MS-MARCO}}$ as the initial weak reference summary generator is the most effective one. Based on the ROUGE-SU4 score, we find that our model is most effective in terms of the F1 metric when the BERTSUM$_{\text{EXT}}$ model is used as the

initial weak reference summary generator; whereas in terms of the Recall metric, our model with $\text{RoBERTa}_{\text{MS-MARCO}}$ performs the best. In terms of the F1 metric, the proposed $\text{PQSUM}_{\text{WSL-DS}}$ model with $\text{RoBERTa}_{\text{MS-MARCO}}$ sets a new state-of-the-art based on the ROUGE-1 score with an improvement of 4.54% from the previous state-of-the-art QUERYSUM model [164]; while this model with $\text{BERTSUM}_{\text{EXT}}$ outperforms QUERYSUM with an improvement of 15.16% and 7.58% based on ROUGE-2 and ROUGE-SU4 respectively [164]. In terms of the Recall metric, we observe that the $\text{PQSUM}_{\text{WSL-DS}}$ with $\text{BERTSUM}_{\text{EXT}}$ outperforms the previous state-of-the-art Dual-CES with an improvement of 7.53% based on ROUGE-2.

We find in the DUC 2007 dataset that for the proposed $\text{PQSUM}_{\text{WSL-DS}}$ model, the best performance is obtained in terms both Recall and F1 when the model utilizes $\text{RoBERTa}_{\text{MS-MARCO}}$ for the initial weak reference summary generation (similar to its performance in the DUC 2005 dataset). More notably, this model outperforms all the prior work in terms of the F1 metric for all ROUGE scores, with an improvement of 3.28% [164], 5.60% [128], and 5.29% [128] from the previous state-of-the-art models based on ROUGE-1, ROUGE-2, and ROUGE-SU4 respectively. Further to note that when our proposed model utilizes the BERTSUM model [92] as the initial weak reference generator, it also outperforms all the prior work in terms of F1 for all ROUGE scores. However, none of our models could outperform the previous state-of-the-art in terms of the Recall metric.

While comparing between the baselines, we find that in both DUC 2006 and DUC 2007 datasets, our abstractive baseline outperforms its extractive counterpart. However, in the DUC 2007 dataset, we find that the extractive baseline performs better than the abstractive one. This may indicate that the gold reference summaries in the DUC 2007 dataset are more extractive in nature. Moreover, while comparing the baselines with our proposed model, we find that for all initial weak reference summary generators, the proposed model outperforms the baselines in all datasets. This suggests the effectiveness of utilizing weakly supervised learning with the transformer-based summarization models for the MD-QFAS task. Furthermore, the improvements in our proposed models from the baselines are **statistically significant** based on paired t-test ($p \leq .05$).

It should also be pointed out that even though our abstractive baseline outperforms the extractive baseline, we find in all datasets that for initial weak reference summary generation, using extractive models ($\text{BERTSUM}_{\text{EXT}}$, $\text{RoBERTa}_{\text{MS-MARCO}}$) are more effective than the abstractive model ($\text{BERTSUM}_{\text{EXT-ABS}}$). This finding may suggest that while utilizing distant supervision to replace each summary sentence in the initial weak reference summary with the most similar sentence found in the multi-document gold reference summaries, the usage of extractive models for initial weak reference summary generation is more effective for sentence similarity modeling.

Table 4.8: Ablation Test result in terms of F1 based on the average across all three datasets. Here, '§' denotes that the initial weak reference summaries in the $\text{PQSUM}_{\text{WSL-DS}}$ model are generated using the $\text{RoBERTa}_{\text{MS-MARCO}}$ model, while 'without' is denoted by 'w/o' and 'ROUGE' is denoted by 'R'.

Model	R-1	R-2	R-SU4
$\text{PQSUM}_{\text{WSL-DS}}$ §	**42.84**	**10.80**	**16.30**
w/o Weakly Supervised Learning	40.12	9.43	14.65
w/o Distant Supervision	41.88	10.16	15.55
w/o Trigram Blocking	41.01	10.53	15.87

Ablation Studies: We also conduct ablation studies to further study the effects of different methods used in our proposed models. For the ablation test, we select the $\text{PQSUM}_{\text{WSL-DS}}$ that utilize the $\text{RoBERTa}_{\text{MS-MARCO}}$ model as the initial weak reference summary generator. The result of our ablation study based on the average ROUGE scores in terms of the F1 metric across all datasets is shown in Table 4.8.

We find from the Table 4.8 that by ranking the sentences in the source documents using the $\text{RoBERTa}_{\text{MS-MARCO}}$ model without leveraging *Weakly Supervised Learning* to fine-tune the BERTSUM model, the performance is **significantly** degraded (based on paired t-test with $p \leq .05$) by 6.35%, 12.69%, and 10.12% in terms

of ROUGE-1, ROUGE-2, and ROUGE-SU4 respectively. The performance deterioration also occurs if we exclude *Distant Supervision* by removing the $\text{RoBERTa}_{\text{MRPC}}$ model as well as if the *Trigram Blocking* is not utilized. However, in these two cases, the performance deterioration is **not statistically significant** based on paired t-test ($p \leq .05$).

4.2.5 Summary

In this section of Chapter 4, we propose a novel weakly supervised learning approach for the query focused multi-document abstractive summarization task that alleviates the computational complexity issue to train transformer-based models in long text sequences. We show that our proposed model could effectively leverage the advantages of fine-tuning pre-trained summarization models by tackling the issue of no labeled training data of each individual document in datasets available for such tasks. Extensive experiments show that our proposed approach sets a new state-of-the-art result in terms of various evaluation metrics for the MD-QFAS task in three benchmark datasets.

5 Conclusions and Future Work

We conclude the thesis in this chapter by first addressing our concluding remarks followed by discussing our plans for the future work.

5.1 Conclusions

In this thesis, we utilize the transformer architecture on two types of question answering tasks: (i) answer sentence selection and (ii) answer sentence generation.

For the answer sentence selection problem, we study the answer selection and ranking task. For this task, we present two new approaches via utilizing contextualized embeddings with the transformer encoder. In one approach, we extract feature-based contextualized embeddings from BERT/ELMo and integrate them with a randomly initialized transformer encoder. Our experimental results on six benchmark datasets demonstrate that the performance of our feature-based approach is comparable with most of the prior work. For the other approach, we utilize various pre-trained transformer encoder-based contextualized language mod-

els such as BERT and RoBERTa and fine-tune them for the answer selection and ranking task. Based on extensive experiments, we find that our fine-tuning-based approach is more effective than the feature-based approach and obtains new state-of-the-art results on all six datasets in terms of the MRR metric. In contrary to the recent state-of-the-art answer selection models that leveraged transfer learning from large question answering datasets, our model did not require such transfer learning to obtain the state-of-the-art performance.

For the other question answering task, we study the query focused abstractive text summarization task for answer summary generation. We present several novel approaches for this task to address both single-document and multi-document scenarios. For the single-document scenario, we handle the few-shot learning issue in the Debatpedia dataset via utilizing transfer learning from a transformer-based generic summarization model. We show that our proposed approach significantly outperforms the prior work for the query focused summarization task in this dataset. In addition, we address several issues in the Debatepedia dataset and show that this dataset is more of a generic summarization dataset. For the query focused summarization task in multi-document scenarios, we handle the issue of no dedicated labeled training data in the datasets available for such tasks via proposing a weakly supervised learning model. We show that our proposed model that utilizes distant supervision to generate the weak reference summary of each individual document

in a document set can effectively leverage the advantages of fine-tuning pre-trained transformer-based generic summarization models for the query focused summarization task. Moreover, our proposed model tackles the computational complexity issue to train transformer-based models in long documents. Experimental results in three benchmark datasets for the query focused multi-document summarization task show that our proposed model is very effective to generate the abstractive summaries and outperforms many prior works (both extractive and abstractive) to set new state-of-the-art results in terms of various evaluation metrics.

5.2 Future Work

In the future, we will utilize various new transformer-based models [101, 141, 142, 65, 176, 129, 69, 122, 26, 39, 86, 67] to investigate their effectiveness in the question answering task. In addition, we will utilize transformer-based models for question answering on other domains, such as the biomedical domain [3] or the multilingual domain [29]. We will also utilize the transformer architecture [150] on more tasks, such as information retrieval applications [58, 59, 170, 60, 102], sentiment analysis [94, 173, 95], learning from unlabeled or imbalanced datasets [7, 8, 93], and automatic chart question answering [66]. Finally, for the future reproducibility of our experiments, we have made the source codes used in this thesis publicly available here: *https://github.com/tahmedge/Tahmid-MSc-Thesis-YorkU*.

Bibliography

[1] Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*, 2019.

[2] Chudamani Aryal and Yllias Chali. Selection driven query focused abstractive document summarization. In *Canadian Conference on Artificial Intelligence*, pages 118–124. Springer, 2020.

[3] Sofia J Athenikos and Hyoil Han. Biomedical question answering: A survey. *Computer Methods and Programs in Biomedicine*, 99(1):1–24, 2010.

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[6] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[7] M Saiful Bari, Shafiq Joty, and Prathyusha Jwalapuram. Zero-resource cross-lingual named entity recognition. *arXiv preprint arXiv:1911.09812*, 2019.

[8] M Saiful Bari, Muhammad Tasnim Mohiuddin, and Shafiq Joty. Multimix: A robust data augmentation strategy for cross-lingual nlp. *arXiv preprint arXiv:2004.13240*, 2020.

[9] Tal Baumel, Raphael Cohen, and Michael Elhadad. Topic concentration in query focused summarization datasets. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2573–2579, 2016.

[10] Tal Baumel, Matan Eyal, and Michael Elhadad. Query focused abstractive summarization: Incorporating query relevance, multi-document coverage, and summary length constraints into seq2seq models. *arXiv preprint arXiv:1801.07704*, 2018.

[11] Steven M. Beitzel, Eric C. Jensen, and Ophir Frieder. *MAP*, pages 1691–1692. Springer US, Boston, MA, 2009. URL `https://doi.org/10.1007/978-0-387-39940-9_492`.

[12] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[13] Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1987–1990, 2017.

[14] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. Massive exploration of neural machine translation architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1451, 2017.

[15] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[16] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[17] Yves Chauvin and David E Rumelhart. *Backpropagation: theory, architectures, and applications*. Psychology Press, 1995.

[18] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

[19] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. Distraction-based neural networks for modeling documents. In *Proceedings of the Twenty-*

Fifth International Joint Conference on Artificial Intelligence, pages 2754–2760, 2016.

[20] Qian Chen, Zhu Zhuo, and Wen Wang. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*, 2019.

[21] Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, Liang He, and Weijie An. Enhancing recurrent neural networks with positional attention for question answering. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 993–996, 2017.

[22] Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, and Liang He. CAN: Enhancing sentence similarity modeling with collaborative and adversarial network. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 815–824, 2018.

[23] Qin Chen, Qinmin Hu, Jimmy Xiangji Huang, and Liang He. CA-RNN: Using context-aligned recurrent neural networks for modeling sentence similarity. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[24] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: encoder–decoder approaches. *Syntax, Semantics and Structure in Statistical Translation*, page 103, 2014.

[25] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[26] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[27] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[28] Philipp Cimiano, Christina Unger, and John McCrae. Ontology-based interpretation of natural language. *Synthesis Lectures on Human Language Technologies*, 7(2):1–178, 2014.

[29] Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *arXiv preprint arXiv:2003.05002*, 2020.

[30] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

[31] Nick Craswell. *Mean Reciprocal Rank*, pages 1703–1703. Springer US, Boston, MA, 2009. URL `https://doi.org/10.1007/978-0-387-39940-9_488`.

[32] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. Pre-training with whole word masking for chinese bert. *arXiv preprint arXiv:1906.08101*, 2019.

[33] Charles G Cullen. *Matrices and linear transformations*. Courier Corporation, 2012.

[34] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, 2019.

[35] Yang Deng, Wai Lam, Yuexiang Xie, Daoyuan Chen, Yaliang Li, Min Yang, and Ying Shen. Joint learning of answer selection and answer summary generation in community question answering. *arXiv preprint arXiv:1911.09801*, 2019.

[36] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, 2019.

[37] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, pages 13063–13075, 2019.

[38] Xinya Du, Junru Shao, and Claire Cardie. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, 2017.

[39] Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. Beyond english-centric multilingual machine translation. *arXiv preprint arXiv:2010.11125*, 2020.

[40] Guy Feigenblat, Haggai Roitman, Odellia Boni, and David Konopnicki. Unsupervised query-focused multi-document summarization using the cross entropy method. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 961–964, 2017.

[41] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. Codebert: A pre-trained model for programming and natural languages. *arXiv preprint arXiv:2002.08155*, 2020.

[42] Tanvir Ahmed Fuad, Mir Tafseer Nayeem, Asif Mahmud, and Yllias Chali. Neural sentence fusion for diversity driven abstractive multi-document summarization. *Computer Speech and Language*, 58:216–230, 2019.

[43] Siddhant Garg, Thuy Vu, and Alessandro Moschitti. Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. *arXiv preprint arXiv:1911.04118*, 2019.

[44] Michael Garland, Scott Le Grand, John Nickolls, Joshua Anderson, Jim Hardwick, Scott Morton, Everett Phillips, Yao Zhang, and Vasily Volkov. Parallel computing experiences with CUDA. *IEEE Micro*, 28(4):13–27, 2008.

[45] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

[46] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[47] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific

language model pretraining for biomedical natural language processing. *arXiv preprint arXiv:2007.15779*, 2020.

[48] Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, 2009.

[49] Hua He and Jimmy Lin. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 937–948, 2016.

[50] Hua He, Kevin Gimpel, and Jimmy Lin. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586, 2015.

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[52] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

[53] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[54] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[55] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[56] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, and Corso Elvezia. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. *A Field Guide to Dynamical Recurrent Neural Networks*, pages 237–244, 2001.

[57] Md Tahmid Hossain, Shyh Wei Teng, Dengsheng Zhang, Suryani Lim, and Guojun Lu. Distortion robust image classification using deep convolutional neural network with discrete cosine transform. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 659–663, 2019.

[58] Xiangji Huang and Qinmin Hu. A bayesian learning approach to promoting diversity in ranking for biomedical information retrieval. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 307–314, 2009.

[59] Xiangji Huang, Fuchun Peng, Dale Schuurmans, Nick Cercone, and Stephen E. Robertson. Applying machine learning to text segmentation for information retrieval. *Information Retrieval*, 6(3-4):333–362, 2003.

[60] Xiangji Huang, Ming Zhong, and Luo Si. York University at TREC 2005: Genomics track. In *Proceedings of the Fourteenth Text REtrieval Conference, TREC*, 2005.

[61] Tatsuya Ishigaki, Hen-Hsen Huang, Hiroya Takamura, Hsin-Hsi Chen, and Manabu Okumura. Neural query-biased abstractive summarization using copying mechanism. In *European Conference on Information Retrieval*, pages 174–181. Springer, 2020.

[62] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, 2017.

[63] Sanjay Kamath, Brigitte Grau, and Yue Ma. Predicting and integrating expected answer types into a simple recurrent neural network model for answer sentence selection. *Computación y Sistemas*, 23(3), 2019.

[64] Aditya Kanade, Petros Maniatis, Gogul Balakrishnan, and Kensen Shi. Pre-trained contextual embedding of source code. *arXiv preprint arXiv:2001.00059*, 2019.

[65] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[66] Dae Hyun Kim, Enamul Hoque, and Maneesh Agrawala. Answering questions about charts and generating visual explanations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[67] Young Jin Kim and Hany Hassan Awadalla. Fastformers: Highly efficient transformer models for natural language understanding. *arXiv preprint arXiv:2010.13382*, 2020.

[68] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[69] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations*, 2019.

[70] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[71] Sayali Kulkarni, Sheide Chammas, Wan Zhu, Fei Sha, and Eugene Ie. Aquamuse: Automatically generating datasets for query-based multi-document summarization. *arXiv preprint arXiv:2010.12694*, 2020.

[72] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466, 2019.

[73] Tuan Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. A gated self-attention memory network for answer selection. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5955–5961, 2019.

[74] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[75] Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Huang. Utilizing bidirectional encoder representations from transformers for answer selection task. In *Proceedings of the V AMMCS International Conference: Extended Abstract*, page 221, 2019.

[76] Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Huang. Query focused abstractive summarization via incorporating query relevance and transfer learning with transformer models. In *Canadian Conference on Artificial Intelligence*, pages 342–348. Springer, 2020.

[77] Md Tahmid Rahman Laskar, Enamul Hoque, and Jimmy Xiangji Huang. Utilizing bidirectional encoder representations from transformers for answer selection. *arXiv preprint arXiv:2011.07208*, 2020.

[78] Md Tahmid Rahman Laskar, Enamul Hoque, and Xiangji Huang. WSL-DS: Weakly supervised learning with distant supervision for query focused multi-document abstractive summarization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5647–5654, 2020.

[79] Md Tahmid Rahman Laskar, Xiangji Huang, and Enamul Hoque. Contextualized embeddings based transformer encoder for sentence similarity modeling in answer selection task. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5505–5514, 2020.

[80] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113, 1997.

[81] Hang Le, Loïc Vial, Jibril Frej, Vincent Segonne, Maximin Coavoux, Benjamin Lecouteux, Alexandre Allauzen, Benoît Crabbé, Laurent Besacier, and Didier Schwab. FlauBERT: Unsupervised language model pre-training for french. *arXiv preprint arXiv:1912.05372*, 2019.

[82] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[83] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[84] Hang Li and Zhengdong Lu. Deep learning for information retrieval. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1203–1206, 2016.

[85] Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, 2016.

[86] Xian Li, Asa Cooper Stickland, Yuqing Tang, and Xiang Kong. Deep transformers with latent depth. *arXiv preprint arXiv:2009.13102*, 2020.

[87] Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, et al. Xglue: A new benchmark dataset for cross-lingual pre-training, understanding and generation. *arXiv preprint arXiv:2004.01401*, 2020.

[88] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[89] Linqing Liu, Wei Yang, Jinfeng Rao, Raphael Tang, and Jimmy Lin. Incorporating contextual and syntactic structures improves semantic similarity modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1204–1209, 2019.

[90] Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 912–921, 2015.

[91] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, 2019.

[92] Yang Liu and Mirella Lapata. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3721–3731, 2019.

[93] Yang Liu, Aijun An, and Xiangji Huang. Boosting prediction accuracy on imbalanced datasets with SVM ensembles. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining, PAKDD*, pages 107–118, 2006.

[94] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. ARSA: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 607–614, 2007.

[95] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. Modeling and predicting the helpfulness of online reviews. In *Proceedings of the 8th IEEE International Conference on Data Mining*, pages 443–452, 2008.

[96] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[97] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[98] Shulei Ma, Zhi-Hong Deng, and Yunlun Yang. An unsupervised multi-document summarization framework based on neural document model. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1514–1523, 2016.

[99] Harish Tayyar Madabushi, Mark Lee, and John Barnden. Integrating question classification and deep learning for improved answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3283–3294, 2018.

[100] Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. Camembert: a tasty french language model. *arXiv preprint arXiv:1911.03894*, 2019.

[101] Yoshitomo Matsubara, Thuy Vu, and Alessandro Moschitti. Reranking for efficient transformer-based answer selection. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1577–1580, 2020.

[102] Jun Miao, Jimmy Xiangji Huang, and Zheng Ye. Proximity-based rocchio's model for pseudo relevance. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 535–544, 2012.

[103] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[104] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in Bioinformatics*, 18(5):851–869, 2017.

[105] Ahnaf Munir, Md Tahmid Rahman Laskar, Md Sakhawat Hossen, and Salimur Choudhury. A localized fault tolerant load balancing algorithm for rfid systems. *Journal of Ambient Intelligence and Humanized Computing*, 10(11): 4305–4317, 2019.

[106] Kevin P Murphy. *Machine learning: A Probabilistic Perspective*. MIT press, 2012.

[107] Preslav Nakov, Lluís Màrquez, Walid Magdy, Alessandro Moschitti, Jim Glass, and Bilal Randeree. SemEval-2015 task 3: Answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, pages 269–281, 2015.

[108] Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, Abed Alhakim Freihat, Jim Glass, and Bilal Randeree. SemEval-2016 task 3: Community question answering. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, pages 525–545, 2016.

[109] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. Semeval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, pages 27–48, 2017.

[110] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, 2016.

[111] Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, 2018.

[112] Mir Tafseer Nayeem, Tanvir Ahmed Fuad, and Yllias Chali. Abstractive unsupervised multi-document summarization using paraphrastic sentence fusion. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1191–1204, 2018.

[113] Preksha Nema, Mitesh M Khapra, Anirban Laha, and Balaraman Ravindran. Diversity driven attention model for query-based abstractive summarization.

In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1063–1072, 2017.

[114] Kyosuke Nishida, Itsumi Saito, Kosuke Nishida, Kazutoshi Shinoda, Atsushi Otsuka, Hisako Asano, and Junji Tomita. Multi-style generative reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2273–2284, 2019.

[115] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[116] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, 2018.

[117] Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, 2019.

[118] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[119] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, 2018.

[120] Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP*, pages 7–14, 2019.

[121] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*, 2019.

[122] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *arXiv preprint arXiv:2003.08271*, 2020.

[123] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[124] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

[125] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[126] Jinfeng Rao, Hua He, and Jimmy Lin. Noise-contrastive estimation for answer selection with deep neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1913–1916, 2016.

[127] Jinfeng Rao, Linqing Liu, Yi Tay, Wei Yang, Peng Shi, and Jimmy Lin. Bridging the gap between relevance matching and semantic matching for short text similarity modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5373–5384, 2019.

[128] Haggai Roitman, Guy Feigenblat, Doron Cohen, Odellia Boni, and David Konopnicki. Unsupervised dual-cascade learning with pseudo-feedback distillation for query-focused extractive summarization. In *Proceedings of The Web Conference 2020*, pages 2577–2584, 2020.

[129] Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280, 2020.

[130] Reuven Y Rubinstein and Dirk P Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer Science & Business Media, 2004.

[131] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[132] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, 2015.

[133] Stuart J Russell and Peter Norvig. *Artificial Intelligence: A modern approach*. Prentice-Hall, 1995.

[134] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[135] Baskaran Sankaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah. Temporal attention model for neural machine translation. *arXiv preprint arXiv:1608.02927*, 2016.

[136] Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*, 2016.

[137] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[138] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, 2017.

[139] Lei Sha, Xiaodong Zhang, Feng Qian, Baobao Chang, and Zhifang Sui. A multi-view fusion neural network for answer selection. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[140] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[141] Luca Soldaini and Alessandro Moschitti. The cascade transformer: an application for efficient answer sentence selection. *arXiv preprint arXiv:2005.02534*, 2020.

[142] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mass: Masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*, pages 5926–5936, 2019.

[143] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[144] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. LSTM-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.

[145] Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. Learning to rank question answer pairs with holographic dual lstm architecture. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 695–704, 2017.

[146] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 583–591, 2018.

[147] Wilson L Taylor. "Cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433, 1953.

[148] Trias Thireou and Martin Reczko. Bidirectional long short-term memory networks for predicting the subcellular localization of eukaryotic proteins. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3 (4):441–446, 2007.

[149] Kateryna Tymoshenko and Alessandro Moschitti. Cross-pair text representations for answer sentence selection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2162–2173, 2018.

[150] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[151] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018:13 pages, 2018.

[152] Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. Using dependency-based features to take the'para-farce'out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop 2006*, pages 131–138, 2006.

[153] Xiaojun Wan and Jianguo Xiao. Graph-based multi-modality learning for topic-focused multi-document summarization. In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[154] Xiaojun Wan and Jianmin Zhang. Ctsum: extracting more certain summaries for news articles. In *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 787–796, 2014.

[155] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, 2018.

[156] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3266–3280, 2019.

[157] Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 707–712, 2015.

[158] Dingding Wang, Shenghuo Zhu, Tao Li, Yun Chi, and Yihong Gong. Integrating clustering and multi-document summarization to improve document understanding. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 1435–1436, 2008.

[159] Mengqiu Wang, Noah A Smith, and Teruko Mitamura. What is the jeopardy model? a quasi-synchronous grammar for qa. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2007.

[160] Yizhong Wang, Kai Liu, Jing Liu, Wei He, Yajuan Lyu, Hua Wu, Sujian Li, and Haifeng Wang. Multi-passage machine reading comprehension with cross-passage answer verification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1918–1927, 2018.

[161] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. *arXiv preprint arXiv:1602.07019*, 2016.

[162] Gail Weiss, Yoav Goldberg, and Eran Yahav. On the practical computational power of finite precision rnns for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 740–745, 2018.

[163] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. HuggingFace's Transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

[164] Yumo Xu and Mirella Lapata. Query focused multi-document summarization with distant supervision. *arXiv preprint arXiv:2004.03027*, 2020.

[165] Yi Yang, Wen-tau Yih, and Christopher Meek. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, 2015.

[166] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

[167] Jin-ge Yao, Xiaojun Wan, and Jianguo Xiao. Compressive document summarization via sparse optimization. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[168] Jin-Ge Yao, Xiaojun Wan, and Jianguo Xiao. Recent advances in document summarization. *Knowledge and Information Systems*, 53(2):297–336, 2017.

[169] Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. Question answering using enhanced lexical semantic models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1744–1753, 2013.

[170] Xiaoshi Yin, Jimmy Xiangji Huang, Zhoujun Li, and Xiaofeng Zhou. A survival modeling approach to biomedical search result diversification using wikipedia. *IEEE Transactions on Knowledge and Data Engineering*, 25(6): 1201–1212, 2013.

[171] Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. A compare-aggregate model with latent clustering for answer selection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2093–2096, 2019.

[172] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *arXiv preprint arXiv:1708.02709*, 2017.

[173] Xiaohui Yu, Yang Liu, Xiangji Huang, and Aijun An. Mining online reviews for predicting sales performance: A case study in the movie domain. *IEEE Transactions on Knowledge and Data Engineering*, 24(4):720–734, 2012.

[174] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*, 2020.

[175] Ethan Zhang and Yi Zhang. *Average Precision*, pages 192–193. Springer US, Boston, MA, 2009. URL `https://doi.org/10.1007/978-0-387-39940-9_482`.

[176] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*, 2019.

[177] Yinan Zhang, Raphael Tang, and Jimmy Lin. Explicit pairwise word interaction modeling improves pretrained transformers for english semantic similarity tasks. *arXiv preprint arXiv:1911.02847*, 2019.

[178] Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. DialoGPT: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*, 2019.

[179] Sheng-hua Zhong, Yan Liu, Bin Li, and Jing Long. Query-oriented unsupervised multi-document summarization via deep learning model. *Expert Systems with Applications*, 42(21):8146–8155, 2015.

[180] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 19–27, 2015.