Improving Retriever Performance in Handling Ambiguous Queries

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Sai Lakshmi Poojitha Nandigam
201402182
poojitha.nandigam@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
March 2024

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Improving Retriever Performance in Handling Ambiguous Queries" by Sai Lakshmi Poojitha Nandigam, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. Manish Shrivastava

To Amma and Nannagaru

Acknowledgments

I'm thankful for many things in my life, but the most important is the encouragement I got from friends and family to complete my research and write my thesis.

I am grateful to Amma and Nannagaru for their unwavering love, and to my friends for being the pillars when it was difficult to make progress in my research. The amount of blood, sweat, and tears I've put into finishing this is unreal.

I am very thankful to my advisor, Prof. Manish Shrivastava, who has been incredibly patient and supportive in my research path and career. I would like to thank Nikhil for valuable discussions throughout these years. Srija for pushing me to do research at the most crucial time, Nishanth and Parth for giving me solid advice whenever I need, Annayya, Anshika, Ananya and Aquib for being supportive and Mahtab for valuable comments during my paper submission, Winni Akka for valuable advice regarding paper writing, Abhinav for helping with choosing conferences, Gautam and Rochelle who have supported me to write my paper, my Valorant friends Veidic and Morpheus for supporting me while working on my paper, Quillbot and OpenAI for inventing ChatGPT. I would also like to thank Vital sir for helping me with the thesis submission.

Abstract

In the realm of open-domain question answering, a common challenge arises from the inherent ambiguity of user queries. Conventional question-answering systems that offer a single answer often falter when confronted with ambiguity, as questions may be subject to multiple interpretations and yield various distinct answers. This paper addresses this challenge through the lens of multi-answer retrieval, a task focused on retrieving passages capable of capturing the diverse array of answers to a single question.

Our approach introduces a re-ranking methodology that leverages Determinantal point processes, employing BERT embeddings as kernels. This technique takes a holistic view by jointly considering both query-passage relevance and passage-passage correlation. The goal is to retrieve passages that not only align with the user's query but also encompass a diversity of information. Empirical results underscore the effectiveness of our re-ranking approach, demonstrating its superiority over state-of-the-art methods, particularly when evaluated on the AmbigQA dataset.

In parallel, the field of question-answering, powered by reader models or answer generation, has seen significant advancements. Neural sequence-to-sequence models, especially those incorporating attention mechanisms, have played a vital role in this progress. However, these models encounter a limitation as they predominantly rely on pointer generators that exclusively target words from the source passage, even though their goal is to provide answers that involve both the question and the source text. To address this constraint, we introduce an innovative query pointer module within a comprehensive multi-pointer generator framework. This module facilitates the generation of answers that combine question and passage information in diverse ways, resulting in contextually relevant responses that effectively address the nuances of user queries. Our model demonstrates a significant improvement of 3.5 points in Rouge-L scores when compared to the state-of-the-art model on existing datasets. These advancements collectively signify substantial progress in both multi-answer retrieval and question-answering research, offering the promise of enhanced capabilities in information retrieval, natural language understanding, and content summarization.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The need to handle vast amounts of data generated every second has led to the emergence of new subdomains within Computer Science. These subdomains are designed to extract value from, and in some cases profit from, the enormous data resources available. However, current algorithms are limited in their ability to accurately understand natural language data from large resources. Open-domain Question answering is a promising technique that, if successful, could provide a much-needed solution to this long-standing issue.

The advent of chatbots has revolutionized the way in which we interact with technology, and has led to an increasing demand for natural language processing that is based on user queries. For instance, in the field of education, a student might pose a question such as "Why is Global warming increasing year to year?", and an Open-domain question answering system can retrieve relevant information from various sources available on the internet (such as a Wikipedia page on Global warming or a research paper or a blog) and generate a well-formed answer that maintains relevancy with the student's query.

In the legal domain, paralegals often face the challenging task of manually extracting pertinent information from extensive legal documents for specific cases. Here, the integration of a question-answering system plays a crucial role by expediting and optimizing the complex process of information retrieval and contextual summarization within a vast legal document corpus. This technology serves as an invaluable assistant, significantly streamlining information management for paralegal professionals, ultimately enhancing their efficiency and effectiveness in their legal roles.

The usefulness of an Open-domain question answering system extends beyond education and law. It can be used by search engines, where users seek short and condensed answers, as well as in community question answering, where users require brief summaries of answers. Overall, this technique has the ability to improve the intelligence and efficacy of meeting user requirements, as well as the quality and

relevance of the responses provided to users.

Ambiguous questions, those that lack clarity or have multiple possible interpretations, are a ubiquitous challenge in language understanding and question-answering systems. They are prevalent not only in everyday conversations but also in formal settings, including information retrieval, legal contexts, and scientific research. Ambiguity in questions arises due to diverse factors such as vague language, multiple meanings of terms, varying contexts, and differing interpretations. In many cases, traditional question-answering systems struggle to provide precise and meaningful responses to such questions.

Existing question-answering systems often rely on predefined databases, structured data, and language models to generate responses. However, when confronted with ambiguous inquiries, these systems encounter limitations. They tend to provide single, deterministic answers, which may not align with the various interpretations that different users may have in mind. Consequently, answering ambiguous questions with a single response can lead to inaccuracies, misinterpretations, or incomplete information.

Addressing the challenge of ambiguous questions requires innovative approaches that embrace ambiguity rather than attempt to eliminate it. These approaches may involve providing multiple distinct answers, offering additional context, or utilizing alternative retrieval techniques. The goal is to ensure that question-answering systems can capture the breadth of possible interpretations and empower users to select the response that best aligns with their intended inquiry. This evolving landscape of handling ambiguous questions underscores the need for dynamic and adaptable retrieval techniques that can accommodate diverse user perspectives and enhance the precision and relevance of answers.

In this thesis, we introduce a comprehensive open-domain question answering system comprising two integral components: a passage retriever and an answer generator. In the subsequent sections, we delve into a detailed exposition of these tasks, specifically focusing on Passage Retrieval and Answer Generation. These components are pivotal in ensuring the system's ability to effectively retrieve relevant information and generate precise responses to user queries.

1.2 Passage Retrieval

The objective of an open domain question-answering system is to answer queries based on a large corpus of text. To answer an open domain question, the first step is to retrieve the passages. The Passage retrieval and the Passage re-ranking methods are both defined below.

Passage Retrieval: Passage retrieval is the process of retrieving relevant passages from a large collection of passages in response to a user's query or information need. Let D be a collection of

documents, and Q be a query. The task of passage retrieval aims to retrieve a set of N passages $P = p_1, p_2, ..., p_N$ from D that are most relevant to the query Q.

Passage Re-ranking: Passage re-ranking involves re-ranking a set of passages retrieved by the Passage retriever. Let $P = p_1, p_2, ..., p_N$ be the set of passages returned by the Passage retriever, the passage re-ranker chooses a subset of k passages from P, and ranks these passages based on their estimated relevance to a given query Q.

The relevant passages are retrieved by the passage retriever and forwarded to the answer generation step. To further enhance passage retrieval, a passage re-ranker can be employed to re-rank the passages retrieved by a passage retriever.

The re-ranker assigns a score to each passage that reflect how pertinent each passage is to the query. The scores can be calculated utilizing various techniques such as TF-IDF [50], cosine similarity, BM25, [49], or neural network based models. The passage re-ranking method involves sorting the passages in descending order based on the scores, so that the most relevant passages are at the top of this list. This re-ranked list can then be provided as an input to the Answer generation step.

Re-ranking techniques have been used in the past to vastly improve the accuracy of question-answering ([55];[42]; [35]; [10]). [35] proposes a re-ranker based on an auto-regressive framework in which each passage chosen is dependent on the passages picked at a prior time step in order to address diverse multi-answer retrieval.

1.3 Answer Extraction and Generation

In a Open-domain QA system, the step following passage retrieval is answer generation or answer extraction. It is a major component of an Open domain QA system that determines the answer to the question given the context.

Answer generation is a fundamental component of question-answering systems, which aim to provide accurate and contextually relevant responses to user queries. Answer generation involves the process of identifying and generating appropriate answers to a wide range of questions. While it plays a pivotal role in enhancing the utility and effectiveness of question-answering systems, it is also a complex and multifaceted task that comes with several challenges.

Answer extraction involves predicting an answer span from the retrieved passages, whereas an answer generator requires deep semantic understanding of the passages and generate answer based on the whole text rather than extracting a span. With the advent of neural models and the availability of large

datasets, Answer generation has seen significant improvements in recent years. Generally, neural models are employed to perform answer generation. We concentrate on Generative Readers in our thesis.

Answer generation serves as the crux of contemporary question answering systems. These systems harness advanced natural language processing and machine learning methodologies to extract answers from extensive datasets and unstructured textual content. This capability finds application in a diverse array of contexts, including customer support chatbots, virtual assistants, search engines, and information retrieval

Ambiguity in questions is a common issue, and addressing it requires sophisticated answering systems. These systems are designed to provide multiple distinct answers, disambiguate based on context, and allow users to select the answer that best aligns with their intended query. By offering multiple responses, these systems accommodate diverse interpretations and enhance user satisfaction.

Machine Reading Comprehension(MRC) focuses on training machines to comprehend text passages and respond to queries about them in a way that is analogous to how humans would do so. Machine Reading Comprehension's primary objective is to enable computers to read and comprehend written text so that they can answer queries or provide information based on the text's content.

When compared to the Machine Reading Comprehension (MRC), the answer generation module in an Open domain question answering system is significantly more difficult. This is due to the fact that Machine Reading Comprehension is intended to provide an answer to a question based on a particular context. On the other hand, questions posed to Open domain question-answering systems may originate from a variety of subjects and may be answered from a variety of sources, as a result, the generated answer may include information from a number of different passages.

The level of sophistication of the answer generated may range from simple keyword-based answers to more complex natural language generation (NLG) approaches, which create responses that resemble those produced by humans and are contextually appropriate. The quality of the generated answers is a crucial aspect of Natural Language Processing systems, and it is frequently evaluated using metrics such as fluency, coherence, informativeness, and relevance to the query asked.

1.4 Main Contribution

Our work in this thesis is mostly concentrated on developing a novel passage retriever that is capable of retrieving diverse yet relevant passages as well as an answer generator.

In the first part of this thesis, we provide a diverse passage retrieval technique for questions that can have multiple answers. We present a novel re-ranking method utilizing Determinantal Point processes. We ran our method on AmbigQA dataset [36] and compared it to the state-of-the art method. Our method outperforms it by 3% (top 5), 8% (top 10) for single-answer questions, and 18% (top5) and 21% (top10) for multi-answer retrieval on AmbigQA dataset

In the second part of the thesis, we propose a multi-pointer generator and different ways to combine the query pointer module with the passage pointer module to generate the relevant context of the summary. Also to tackle the lack of data on the query based summarisation, we propose a new query-based abstractive summarisation dataset by adapting a Reading Comprehension dataset to fit the query based summarisation. Our model has improved results on the state of the art model on existing dataset by 3.5 points on Rouge.

1.5 Thesis Organization

The organization of the thesis is as follows:

1. In the first chapter, we discuss the research motivation, main contribution of the thesis, and the organization of the thesis.

2. In the second chapter, we discuss the background work related to Open domain QA systems and ambiguous questions, particularly the retriever-reader model.

3. In the third chapter, we present a novel re-ranking technique based on Dense Passage Retriever (DPR) and Determinantal point processes (DPP) as our retriever model.

4. In the fourth chapter, we present a reader model for generating answers from the passages retrieved utilizing pointer generator networks, sequence to sequence models with attention and Bidirectional Long Short term Memory Network(Bi-LSTM) models. .

5. In the fifth chapter, we conclude the thesis with important observations, conclusions and future work in respective research areas and suggestions for future work.

Chapter 2

Background and Related work

2.1 Open domain QA

The goal of open-domain question answering is to answer questions by using a substantial amount of information from the Internet, such as Wikipedia and other sources. Most users may not ask clear questions to an open-domain question answering system, and questions in general can be interpreted in a variety of ways depending on the context in which they are asked. Ambiguous questions make up more than 50% of the questions in a popular open-domain QA dataset (Natural Questions [28]). Ambiguous questions need a wide range of answers since they may be interpreted in a variety of ways. In this chapter, we concentrate on questions that have multiple distinct answers.

Open domain question-answering systems are designed to generate answers from several data sources. Since similar information can be present across multiple data sources, it introduces a significant amount of redundancy. Traditional open-domain QA ([3]) systems comprise of a Retriever, which retrieves passages relevant to the question. A passage retriever is primarily concerned with retrieving passages that are relevant to the query, and it does not address redundancy in the passages during retrieval. To be able to produce diverse answers to the question, the passages retrieved must be both relevant to the question and distinct from one another. After the retrieval stage, we introduce a novel re-ranking approach to handle redundant passages. As a result, the re-ranked passages would capture most of the diverse answers to the question. In this thesis, we address the multi-answer retrieval task, which entails retrieving passages that can cover the distinct answers.

2.2 Retriever

The Retriever component of the Open domain QA systems aims to retrieve relevant documents or passages that contain the answer given a question. Not only does it retrieve, it also ranks them in the order and awards the most relevant documents with a greater score than the ones with less relevance.

$$R(P, q) = \arg, \max_{p \in P} s(p, q) \tag{2.1}$$

where $s(p, q)$ is a scoring function that calculates the relevance score of a passage p with respect to a query q. The re-ranking function R takes the set of passages P and the query q as inputs and returns the passage p that has the highest relevance score with respect to the query q.

There are two kinds of Retrievers - Sparse Retrievers and Dense Retrievers. The techniques utilized by sparse and dense retrievers are described in the following sections.

2.2.1 Sparse Retriever

Some systems adopt classical Information Retrieval(IR) methods such as TF-IDF [50], [13] and BM25 [49], [58], [48] to retrieve relevant documents. TF-IDF stands for "Term Frequency-Inverse Document Frequency." It is a common technique used in information retrieval and text mining to evaluate the relevance of a document to a query.

In passage retrieval, TF-IDF is used to rank the relevance of passages within a document to a given query. TF-IDF assigns a weight to each term in the query based on its frequency in the document (term frequency) and the inverse frequency of the term across all passages in the corpus (inverse document frequency).

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \tag{2.2}$$

- tf(t, d) is the term frequency of term "t" in passage "d". It measures the number of times the term "t" appears in the passage "d". It can be calculated as:

$$\text{TF}(t, d) = \frac{\text{number of times term } t \text{ appears in passage } d}{\text{total number of terms in passage } d} \tag{2.3}$$

- IDF(t, D) is the inverse document frequency of term "t" across the entire collection of passages "D". It measures the informativeness of the term "t" in the collection of passages. It can be calculated as:

$$\text{IDF}(t, D) = \log \left(\frac{\text{total number of passages in the collection } D}{\text{number of passages containing the term } t} \right) \tag{2.4}$$

BM25 (Best Matching 25) is a ranking function used by search engines to score and rank documents based on their relevance to a given query. It is an improved version of the BM11 algorithm developed by Robertson and Sparck Jones in the 1990s [49]. BM25 is a bag-of-words retrieval model that measures the similarity between a query and a document using a combination of term frequency (TF) and inverse document frequency (IDF) measures, as well as a set of tuning parameters. The score of a document for a given query is computed as follows:

$$\text{BM25}(q,d) = \sum_{i=1}^{|q|} \text{idf}(q_i) \cdot \frac{f(q_i,d) \cdot (k_1+1)}{f(q_i,d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{\text{avgdl}})} \tag{2.5}$$

where:

- q is the query, d is a document, $|q|$ is the length of the query (in terms of words), q_i is the i-th word of the query, $f(q_i, d)$ is the frequency of word q_i in document d
- $\text{idf}(q_i)$ is the inverse document frequency of word q_i in the collection of documents, k_1 and b are tuning parameters that control the impact of term frequency and document length normalization on the score, respectively, $|d|$ is the length of document d (in terms of words), avgdl is the average document length in the collection of documents.

BM25 has been shown to be effective in many information retrieval tasks, and it is widely used in modern search engines, including Elasticsearch and Solr. DrQA([3]) uses Wikipedia as knowledge source and employs a sparse retrieval method using TF-IDF and a recurrent neural network to identify the answer spans. While [57] adopts Anserini retriever([56]) using BM25 as the ranking function and BERT model ([12]) as the reader.

Sparse retrieval based methods, such as TF-IDF and BM25, face challenges when retrieving relevant passages that do not match the question's exact terms. Dense retrieval-based approaches on the other hand, overcome this problem by mapping each word into a vector space in which words with similar meanings tend to be closer together.

2.2.2 Dense Retriever

Unlike sparse retrieval based methods, Dense retrievers utilize dense vector representations of words to compute semantic similarity between the query and passages. A dense retriever would easily be able to retrieve synonyms or paraphrases containing different tokens since synonyms are mapped closer in the vector space.

Dense Passage Retriver(DPR) [25] consists of a passage encoder E_p and a query encoder E_q. It first maps every passage to a d-dimensional vector and builds an index for all the M passages. During run-time, the query encoder E_q encodes the question. It then computes a similarity score utilizing the

pre-computed index of passage vectors and the encoded query vector by performing a dot product of their vectors.

For a given query q and a set of passages P, the objective is to find the passage p $\in$ P that maximizes the similarity score, where the similarity function Sim(q, p) measures the relevance of the passage to the query:

$$p^* = \arg\max_{p \in P} \text{Sim}(E_q, E_p) \tag{2.6}$$

The function Sim(q, p) is usually implemented using neural network-based models, such as Siamese networks or cross-attention mechanisms. These models learn dense embeddings for both queries and passages, allowing for efficient retrieval of relevant passages for a given query.

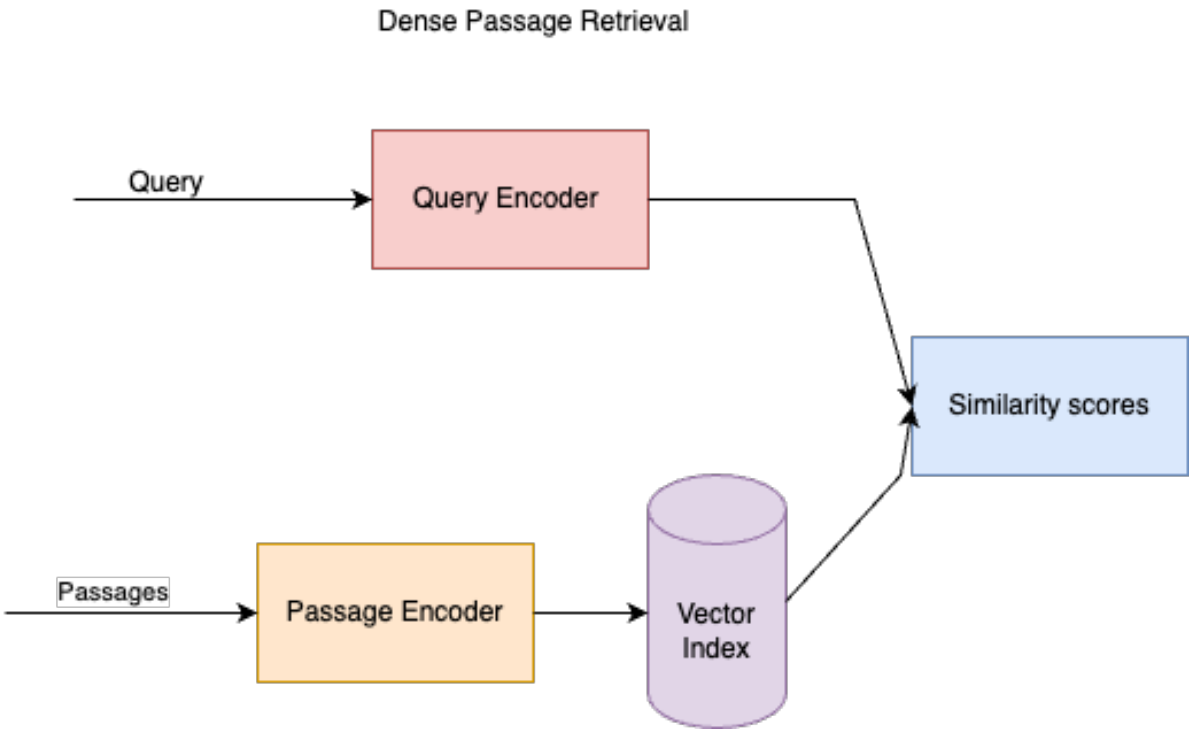

Figure 2.1 An overview of the proposed re-ranking method using Dense Passage Retrieval Framework. A similarity score between the question and passage is computed. This score is utilised to retrieve top-k passages. FAISS index is used to store the pre-computed passage encodings.

Both passage encoder and the query encoder use BERT ([12]) networks, takes the representation at the [CLS] token as the output, d = 768. FAISS ([23]) is utilized during inference to index the passages.

The main advantage of FAISS is its speed and scalability, which enables the efficient search and clustering of large-scale datasets containing millions or billions of high-dimensional vectors.

In the context of Dense Passage Retrieval (DPR), FAISS plays a crucial role in indexing and searching through dense embeddings of passages and queries. It empowers DPR models to quickly identify passages that are most relevant to a given query by efficiently searching through large collections of text data.

2.3 Reader

Reader aims at generating the answer from the retrieved passages. The reader is used in conjunction with a retriever in Open domain Question Answering systems. The retriever picks a collection of relevant passages, and then the reader is used to extract the answer from these passages. Together, the retriever and the reader constitute a robust framework for answering questions in a variety of domains.

There are two categories of Readers: Extractive and Generative. Extractive Reader involves predicting an answer span from the retrieved passages, whereas a Generative Reader generates answers and requires a deeper semantic understanding of the passages. In the past, most of the work has focused on extractive approach, but with the advent of neural models and the availablity of large datasets, Generative Readers have seen significant improvements in recent years. We concentrate on Generative Reader in our thesis.

Extractive readers assume that the answer to a given question exists in the context, and focuses on learning to predict the start and end position of an answer span from the retrieved documents. DPR employs BERT reader to compute the probabilities of passage containing the answer and predicting whether a token is a start or ending position of the answer span. Some open domain QA systems also adopt graph-based readers to learn to predict an answer span from a graph. Graph Reader takes input a graph and learns the passage encoding using Graph convolution networks, and extracts answer spans.

Some systems like DrQA [4] takes the question and passages as input extracting various features such as Part of Speech, Named Entity and Term Frequency from the passage, and adopts a multi-layer Bi-LSTM as the reader that takes the question and paragraphs as input, and predicts the answer span.

Generative reader aims to generate answers instead of extracting answer spans. Many open domain question answering systems like BART [29] and T5 [46] adopt pre-trained sequence to sequence language models as readers. RAG [24], [45] retrieves the documents using DPR and generates answers by incorporating both the query and the relevant documents, and employs a pretrained BART model as

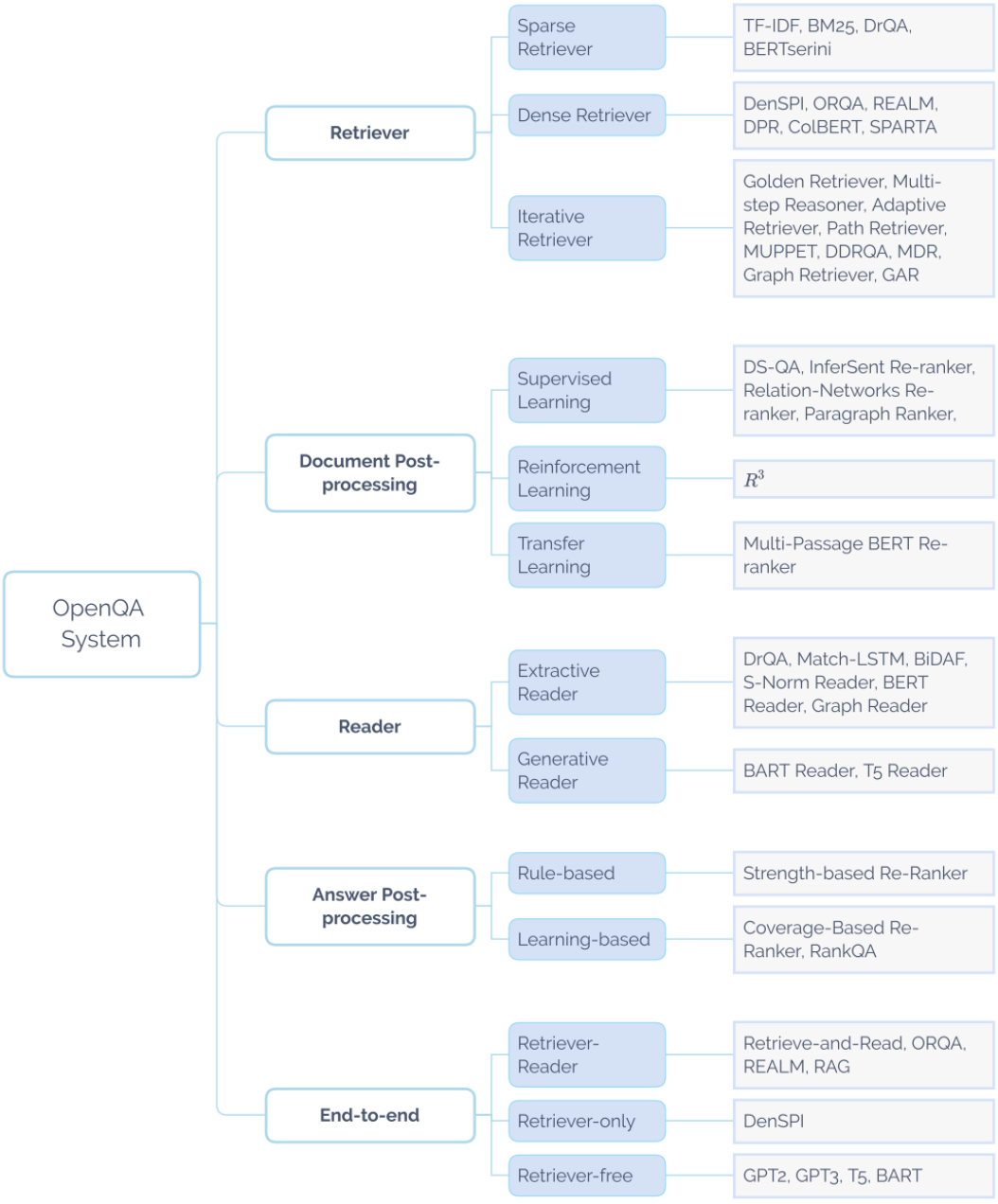

Figure 2.2 [61]: A categorization of open-question answering systems based on retriever-reader architecture

the reader to generate answers. FID [22], [20], [21] first encodes each retrieved document using T5 or BART encoder and performs attention over all the output representations using the decoder to geenerate the final answer.

As shown in 2.2, various architectural approaches have been developed to tackle the challenge of providing precise and contextually relevant answers. These approaches can be categorized using specific model architectures. Large language models (LLMs) like GPT-2 [2], GPT-3 [2], and BART [29] are often regarded as reader-only systems, while Dense Passage Retrieval (DPR) is a representative of retriever-only systems. Models such as REALM [16], ORQA [25], and RAG [24] on the other hand, are classified as retrieve-read systems.

Reader-only systems, exemplified by LLMs such as GPT-2 [2], GPT-3 [2], and BART [29] are designed for comprehensive understanding and answer extraction. These models employ a pre-trained, autoregressive architecture to generate detailed and contextually relevant answers. Given a question or query, they analyze the provided context and generate coherent responses. However, they do not engage in the initial retrieval of documents or passages, relying on externally provided contexts.

Retriever-only systems, like Dense Passage Retrieval (DPR), specialize in swift and efficient document selection. DPR employs a bi-encoder architecture, consisting of separate encoders for questions and passages. It indexes and rapidly scans large corpora to identify potentially relevant passages by comparing the encoded question with encoded passages using a retrieval algorithm. While it excels at document retrieval, it does not perform detailed comprehension or answer extraction.

In contrast, retrieve-read systems, including REALM [16], ORQA [25], and RAG [24], combine the strengths of both retrieval and comprehension. These models often employ a two-step process. Firstly, they use a retriever component that scans and selects documents or passages most likely to contain the answer to a given question. Subsequently, a reader component, often employing LLMs, is used to analyze and extract answers from the retrieved content. REALM, for instance, utilizes a dual-encoder architecture for retrieval and understanding. This combination allows for both efficient document selection and in-depth comprehension, making retrieve-read systems well-suited for open-domain QA tasks.

Each of these QA system architectures, driven by specific model designs, presents distinct advantages and trade-offs. Reader-only systems excel in comprehensive understanding, retriever-only systems prioritize efficient document selection, and retrieve-read systems balance both aspects to provide contextually relevant answers. The choice among these architectures, often leveraged with pre-trained models and retrieval components, plays a pivotal role in designing QA systems to meet diverse user needs and application scenarios.

2.4 Determinantal Point processes

Determinantal Point Processes (DPP) ([27]) are probabilistic models that are successful in recognizing various subsets of elements from a collection while maintaining quality. In natural language processing applications where diversity is required, DPP techniques have shown to be successful. [9], [30], and [8], [52] have employed DPPs to perform summarization by selecting relevant but also diversified items to be included in the summaries. In this study, we offer an unsupervised re-ranking method for retrieving multiple answers adopting Determinantal point processes, where the kernels are modelled by BERT.

Determinantal Point Processes (DPPs) assign a probability to every subset of a ground set, Y. This means that a sample from the process could be empty or it could be all of Y, depending on the probabilities assigned. DPPs are used in various applications where diversity is a key factor. For instance, they can be used to model the positions of basketball players on a court, under the assumption that a team tends to spread out for better coverage [27].

A point process P on a ground set Y is a probability measure over all the 2^Y subsets of Y. P is called a determinantal point process if, when Y is randomly drawn according to P, For every subset $A \subseteq \mathcal{Y}$

$$P(A \subseteq \mathbf{Y}) = det(K_A) \tag{2.7}$$

where K is a real-symmetric $N * N$ positive semi-definite matrix indexed by elements of $\mathcal{Y}$. We refer to K as the marginal kernel. If $A = \{i\}$ is a singleton set, we have,

$$P(i \in Y) = K_{ii} \tag{2.8}$$

If $A = \{i, j\}$ is a two element set, then

$$P(i, j \in \mathbf{Y}) = \begin{vmatrix} K_{ii} & K_{ij} \\ K_{ji} & K_{jj} \end{vmatrix} \tag{2.9}$$

$$= K_{ii}K_{jj} - K_{ij}Kji \tag{2.10}$$

$$= P(i \in \mathbf{Y})P(j \in \mathbf{Y}) - K_{ij}^2 \tag{2.11}$$

Negative correlations between pairs of components are therefore determined by the diagonal elements: high K_{ij} values indicate that elements i and j seldom occur together. DPPs are "diversifying" because they make highly similar elements unlikely to appear together. If we think of the entries of the marginal kernel as measurements of similarity between pairs of elements in Y, then highly similar

elements are unlikely to appear together. This is demonstrated in Equation 2.4. If $K_{ij} = \sqrt{K_{ii}K_{jj}}$, then i and j are "perfectly similar" and do not appear together almost surely. Conversely, when K is diagonal there are no correlations and the elements appear independently

The geometry of DPPs can be explained in terms of determinants, which relate to the volume spanned by the feature vectors of the selected items. This volume is determined by the dot products between the feature vectors, which are measured by the kernel matrix L. The determinant of L is proportional to the probability assigned by the DPP to a set Y, and diverse sets with more orthogonal feature vectors are more probable. Items with parallel feature vectors are selected together with probability zero, and larger feature vectors are more likely to be selected. As the similarity between two items increases, the probabilities of sets containing both of them decrease.

[38] focuses on efficient sampling methods for DPPs and their application to clustering tasks. It introduces a fast sampling algorithm and discusses the advantages of DPPs for capturing diversity in clustering. [59] explores the use of DPPs for sparse regression problems, and presents a framework for random DPPs and discusses how they can be applied to scalable sparse regression tasks. [1] explores the use of DPPs on graphs for image segmentation tasks. It discusses how DPPs can be applied to improve the diversity of selected regions in image segmentation.

[14] explores the application of DPPs in document summarization, specifically for selecting sentences to form informative and diverse summaries. It discusses the integration of DPPs into the summarization process and their effectiveness. [6] investigates the use of DPPs in the context of machine translation, where they are applied to select diverse translation candidates for sentences or phrases. It discusses the potential benefits of using DPPs in improving translation quality.

Chapter 3

Diverse Multi-Answer Retrieval with Determinantal Point Processes

3.1 Introduction

The goal of open-domain question answering is to answer to questions by using a substantial amount of information from the Internet, such as Wikipedia and other sources. Ambiguous questions make up more than 50% of the questions in a popular open-domain QA dataset (Natural Questions [28]). Ambiguous questions need a wide range of answers since they may be interpreted in a variety of ways. In this chapter, we concentrate on problems that have multiple distinct answers.

Open domain question-answering systems are designed to provide answers from a variety of sources of information. Redundancy is substantially increased since the same information may be found in several data sources. A Retriever is a component of traditional open-domain QA systems([3]) that retrieves passages pertinent to the issue. A passage retriever does not take into account repetition in the passages when retrieving them; instead, it is mainly concerned with retrieving passages that are pertinent to the question. The passages that are retrieved must be both relevant to the question and distinct from one another in order to be able to provide a diverse answers to it. We offer a novel re-ranking method for handling redundant passages after the retrieval step. The reranked passages would therefore cover the majority of the different comments to the question. Here, we look into the multi-answer retrieval challenge, which involves finding passages that may handle several answers.

Re-ranking techniques have been used in the past to greatly increase the accuracy of question-answering ([55];[42]; [35]; [10]). [35] proposes a re-ranker based on an auto-regressive framework in which each passage chosen is dependent on the passages picked at a prior time step in order to address diverse multi-answer retrieval.

Determinantal Point Processes (DPP) ([27]) are probabilistic models that are successful in recognizing various subsets of elements from a collection while maintaining quality. In natural language processing applications where diversity is required, DPP techniques have shown to be successful. [9],

[30], and [8], [52] have employed DPPs to perform summarization by selecting relevant but also diversified items to be included in the summaries. In this study, we offer an unsupervised re-ranking method for retrieving multiple answers adopting Determinantal point processes, where the kernels are modeled by BERT.

In recent years, the field of Natural Language Processing (NLP) has witnessed a transformative breakthrough with the advent of BERT (Bidirectional Encoder Representations from Transformers). BERT, developed by AI, has redefined the landscape of language understanding and representation.

At the heart of BERT's innovation lies its distinctive approach to language understanding. While traditional NLP models had predominantly relied on unidirectional processing, either from left to right or right to left, BERT introduced bidirectional context modeling. This bi-directionality allows BERT to consider the entire context surrounding each word within a sentence, enabling it to capture rich contextual information.

Self-attention, also known as intra-attention, is an attention mechanism that relates different positions of a single sequence to compute a representation of the sequence. In the Transformer, self-attention is used to compute a representation of the input sequence for each encoder and decoder layer.

Multi-head attention is a variant of self-attention that allows the model to jointly attend to information from different representation subspaces at different positions. In the Transformer, multi-head attention is used to compute the attention scores in parallel across multiple heads, which are then concatenated and projected to the output dimension.

This bidirectional context modeling is realized through the Transformer's attention mechanism. Formulated mathematically, BERT's attention mechanism calculates attention scores between query (Q) and key (K) vectors and uses them to weight value (V) vectors. The attention calculation can be expressed as follows, where softmax normalizes the attention scores:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

In this equation, d_k represents the dimension of the key vectors. The result is a dense and context-rich representation of each word in the input text.

BERT's ability to capture rich contextual information has revolutionized tasks related to semantic textual similarity. It excels in measuring the semantic likeness between two sentences or phrases. This capability finds application in various NLP tasks, including paraphrase identification, duplicate content detection, and sentiment analysis. BERT's contextual embeddings enable it to discern subtle variations

in meaning, making it a versatile tool for discerning textual similarity.

In the realm of information retrieval, BERT has emerged as a potent asset for document and passage retrieval. Fine-tuned BERT models can effectively match user queries with relevant documents or passages, capitalizing on their contextual understanding of language. This integration of BERT into retrieval models significantly elevates the quality of search results, enhancing tasks like document retrieval, passage retrieval, and open-domain question answering.

The integration of BERT's contextual embeddings into retrieval systems represents a pivotal advancement in information retrieval, offering a profound leap in the precision and relevance of retrieved documents or responses.

The following is a summary of our contributions:
1) We provide a re-ranking technique based on Determinantal point processes that focuses on retrieving diverse passages.

2) Unlike previous methods of re-ranking, our methodology does not need a significant amount of data since it is unsupervised. This is in contrast to the previous approaches ([35]). Instead, we depend on DPP to recognize the passages that are distinct from one another and most pertinent to the question.

3) We show that our approach is superior by comparing our results on the AmbigQA dataset with the state-of-the-art method utilizing the $\mathrm{MRECALL}$ @ k metrics.

3.2 Our method

The re-ranker functions as a filter, selecting a subset of the relevant passages so that they may be used in the process of producing answers to the questions. We frame the passage re-ranking task as a subset selection task. Our goal is to choose a subset of passages (Y) of size k from the ground set $\mathcal{Y}$ of passages (N) that contains all of the answers to a certain question (q). DPP simulates a distribution on each subset of the ground set $\mathcal{Y}$ while simultaneously taking the quality and diversity into account. According to the probability distribution P, a subset Y is selected.

$$P(Y; L) \propto det(L_Y) \tag{3.1}$$

$$P(Y; L) = \frac{\det(L_Y)}{\det(L + I)} \tag{3.2}$$

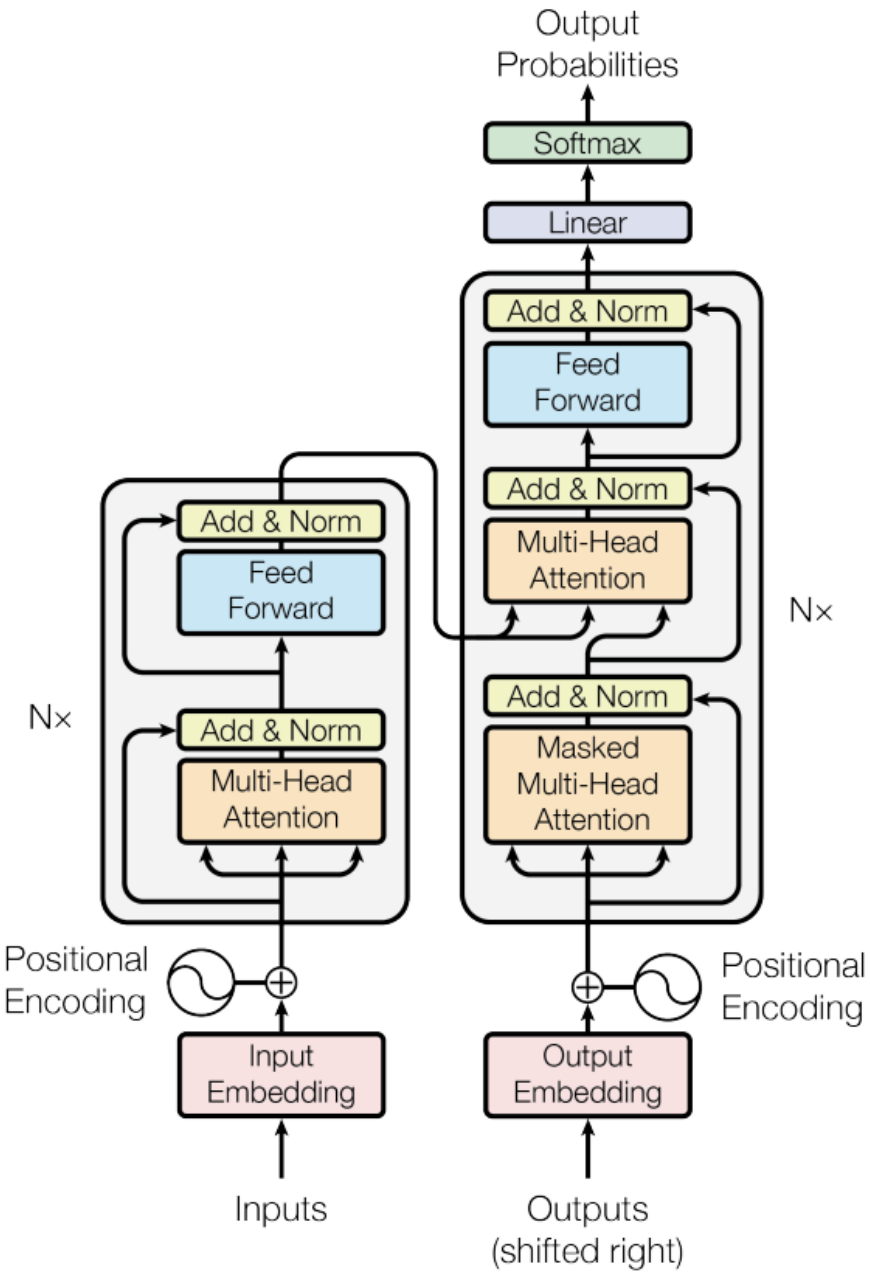

Figure 3.1 [54]Transformer architecture

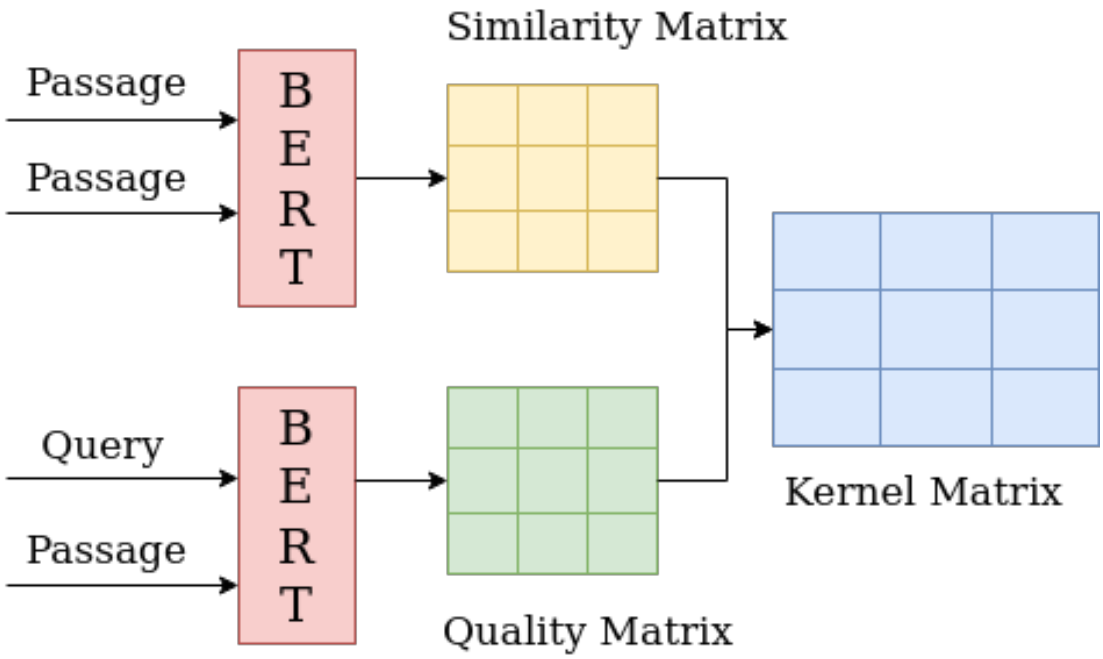

Figure 3.2 An overview of the proposed re-ranking method using DPP. A similarity score between the passages and a quality score between the question and passage are computed. These two scores are utilized to construct the DPP kernel matrix.

where I is the identity matrix, $L \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$ is a positive semi-definite matrix referred as L-*ensemble*, $\det(.)$ stands for the determinant of a matrix, and L_y is the submatrix of L indexed by items in Y. L matrix takes both query-passage relevance as well as passage-passage correlation through eq. 3.3.

$$L_{ij} = Q(i,q) \cdot S(i,j) \cdot Q(j,q) \tag{3.3}$$

The DPP places an emphasis on two different measures: quality and similarity (Fig). The salience of the passage i and whether it provides an answer to the query q are measured by the quality score $Q(i,q)$. To account for the diversity in the passages, similarity score $S(i,j)$ is calculated between two passages i and j. According to([27]), the DPP gives a set Y a probability inversely correlated to the determinant of the L-textit ensemble, which can be represented geometrically as the area of the parallelepiped occupied by the quality and similarity measurements. Since a diverse passage subset takes up more space than a subset of comparable passages does, DPP gives diverse and relevant passages a higher probability than the most relevant and similar passages do.

3.2.1 BERT for Similarity matrix

For the purpose of computing similarity scores, we make use of a pretrained BERT model ([12]; [47]) to construct embeddings for each passage. The model creates a 768 dimensional dense embedding using the passage as input. We generate a similarity matrix $S \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$ for the whole passage set using these embeddings to determine the cosine similarity of each passage. All of the similarity matrix's values fall within the range of $\lceil 0, 1 \rceil$. The similarity value $S(i, j)$ is closer to 1 if passages i and j are similar, closer to 0 if they are different, and it becomes equal to 1 if $i = j$.

$$S(i, j) = cosine\_sim(BERT_A(i), BERT_A(j)) \tag{3.4}$$

3.2.2 BERT for Quality matrix

We use a pretrained BERT model trained on MS MARCO ([41]) for computing the Quality matrix. The model takes in a query and a passage and generates the quality score. Higher quality score indicates that the passage is most relevant to the query and therefore most likely to answer the query. Unlike for computing similarity matrix, we do not perform cosine similarity over the model's outputs to produce a score, instead, we use a BERT encoder that concatenates both query and passage and generates a score. The quality matrix $Q \in \mathbb{R}^{\mathsf{N}\times\mathsf{N}}$ is computed by performing the matrix multiplication of the scores $(N \times 1)$ with it's transpose resulting in $N \times N$ dimensioned vector . These quality scores are then normalized to lie between $\lceil 0, 1 \rceil$.

$$Q(i, j) = Norm(BERT_B([i; j])) \tag{3.5}$$

3.2.3 Sampling

When L matrix is huge, the runtime complexity of traditional DPP sampling techniques increases. We use the effective sampling method known as BFGMInference ([30];[5]). BFGMInference approximates a greedy approach to pick a passage that maximizes the $det(L_Y)$ and adds it to the passage subset.

$$f(Y) = \log\det\left(L_Y\right) \tag{3.6}$$

$$k = \underset{i\in\mathcal{Y}\setminus Y}{\arg\max} f(Y \cup \{i\}) - f(Y) \tag{3.7}$$

3.3 Experiments

In this section, we will describe the passage retrieval technique, the dataset that we used in our research, the evaluation measure, as well as the results of our experiments .

3.3.1 Passage retrieval

The questions' passages are retrieved from Wikipedia using a corpus. Each Wikipedia article is divided into many paragraphs with the same word count. With the help of the Dense Passage Retriever (DPR) ([25]; [33]), we extract passages from Wikipedia that are pertinent to our queries. DPR creates an index and computes encodings for every text that was taken from the Wikipedia corpus. The similarity scores between the query and passage are calculated using the inner product of the two encodings. The most relevant passages—those with the highest scores—are those that are supplied as input to the re-ranker since they are the ones that are most pertinent to the query.

3.3.2 Dataset

An open-domain question-answering dataset known as AmbigQA ([37]) was used for the purpose of evaluating our approach. This dataset includes questions with multiple possible answers. An anonymized collection of Google search requests from people seeking information on various topics was used to construct the dataset. It is divided into train, validation, and test sets and contains 14,042 question-answer pairs drawn from the Natural questions dataset ([28]). 10,036 question-answer pairs make up the train set, 2,002 examples make up the validation set, and 4,042 instances make up the test set.

3.3.3 Evaluation metric

MRECALL @ k ([35]) is used to assess the re-ranking of passages for questions with multiple answers. $MRECALL@k$ stands for mean Recall at k and is used to evaluate the performance of retrieval and recommendation systems. It measures how well a retriever retrieves relevant items within the top-k positions for multiple users or queries.

$$mRecall@k = \frac{1}{N}\sum_{i=1}^{N} Recall@k_i \tag{3.8}$$

N represents the number of queries for which we are calculating the metric. $Recall@k_i$ is the is the Recall at k for the i-th user or query.

- The m in $mRecall@k$ usually means "mean" or "average." This signifies that we're looking at the average recall across several users or queries.
- $Recall$ is a measure of how many relevant items were retrieved compared to the total number of relevant items in the dataset. It's often expressed as a fraction or percentage.

Models	Top5	Top 10
	AmbigQA-Dev	AmbigQA-Dev
DPR$^+$([35])	55.2/36.3	59.3/39.6
DPR$^+$ + [42]	63.4/43.1	65.8/46.4
JPR([35])	64.8/45.2	67.1/48.2
QRR	62.0/42.3	70.8/57.6
DPP-R	**66.9/53.5**	**72.8/58.8**

Table 3.1 Performance of various models on AmbigQA dataset. Each row contains the $\mathrm{MRECALL}$ @ k metrics for single answer retrieval and multi-answer retrieval respectively.

- The $@k$ part specifically tells us that we're focusing on the top-k items within the retrieval. It evaluates how well the system performs in retrieving relevant items when we're only considering the top-k positions.

$$Recall@k = \frac{NumberofRelevantPassagesRetrievedinTop-k}{TotalNumberofRelevantPassages} \tag{3.9}$$

$Recall@k$ serves as a metric for assessing how well a recommendation or information retrieval system performs. It specifically measures the portion of relevant items that our system successfully finds among the top k items. In other words, it helps us understand how effective the system he system is at retrieving pertinent information from its top k suggestions.

- "Number of Relevant Passages Retrieved in Top-k" refers to the number of relevant passages that are among the top-k retrieved results.
- "Total Number of Relevant Items" is the total count of items in the passages list that are relevant to the query.

According to the $MRECALL@k$ measure, if a query returns n replies, the k sections that are returned must include each and every one of the responses. All answers must be included if $n = k$, and the sections obtained must have at least k answers if $n > k$. If all or at least k of the responses to the question are present in the sections that were found, the retrieval is considered successful.

3.4 Results

We contrast our method to a few other baselines, all of which were evaluated using the $\mathrm{MRECALL}$ @ k metric on the AmbigQA dataset.

- **DPR$^+$** [35] integrates REALM ([17]) with DPR ([25]). As described in Section 3.3.1, DPR is a dense retrieval based technique that utilizes the FAISS library to retrieve the relevant documents. Encoders for the query and passage are initialized using REALM and the DPR training method is followed.

- **DPR$^+$ + [42]** employs DPR$^+$ for the first stage of retrieval and the re-ranking method in [42] is applied on the retrieved passages.
- **JPR** [35] employs DPR$^+$ as the initial ranker and an auto-regressive framework is adopted as a re-ranker to generate diverse passages.
- **Query Relevance Re-ranking(QRR)** In this method, we first calculate the quality scores for each passage (described in section 3.2.2) and then we sort the passages based on these scores to pick the top-k passages. Here, similarity among the passages is not considered.
- **DPP-R** We employ our method described in section 3.6 to retrieve highly diverse and relevant passages.

We compute the performance of diverse multi-passage retrieval using the $\mathrm{MRECALL}$ @ k metric defined in the section 3.3.3. Evaluation on the AmbigQA dataset shows that our strategy performs better than the state-of-the-art re-ranking methods. Our method outperforms previous techniques while requiring no human annotations for multi-passage re-ranking, as demonstrated in Table 3.1. Our approach for this assignment works well since DPP is designed to choose a subset of passages that are both high-quality and varied. Experiments show that the DPP-based method yields good results for retrieving passages with multiple answers.

3.5 Discussion

Impact on QA system's performance: Three phases comprise the pipeline of an Open domain question-answering system. 1) Retrieval 2) Re-ranking followed by 3) Answer extraction. The capacity of the system as a whole to respond to a query is considerably improved by improvements in any of these steps. [42], [35] demonstrate how using a re-ranker has improved end-to-end QA. According to Table 3.1's findings, the DPP approach improves re-ranking for both single- and multi-answer questions. We believe that this re-ranking enhancement will also enhance the overall functionality of the end-to-end QA system.

Impact of diversity: Using the DPP and auto-regressive frameworks, respectively, DPP-R and JPR retrieve diverse passages. Other methods, like QRR, only retrieve passages that are relevant to the query and do not address passage repetition. We note that our DPP-based strategy outperforms the QRR approach. Simply evaluating a passage's relevance to the question serves as the basis for re-ranking in QRR, which then returns the top k passages for every given query. DPP-R, on the other hand, accounts for both how relevant a passage is to the query and how similar it is to other passages in order to remove repetitive passages that might otherwise result in diversity in the retrieved passages. In multi-answer retrieval, DPP-R and JPR perform better than other approaches that do not prioritize diversity. DPP-R and JPR have performed better than other approaches for single answer retrieval, with the small caveat

that QRR outperforms JPR in top-10 re-ranking. This highlights the significance of diversity as a factor to take into account when revising the rankings.

Question: What muscles attach to the medial border of scapula	*Answer: Rhomboid major, levator scapulae, rhomboid minor, Serratus anterior*
Re-ranked passage #1	The scapula and are responsible for the internal and external rotation of the shoulder joint, along with humeral abduction. The extrinsic muscles include the biceps, triceps, and deltoid muscles and attach to the coracoid process and supraglenoid tubercle of the scapula, infraglenoid tubercle of the scapula, and spine of the scapula. These muscles are responsible for several actions of the glenohumeral joint. The third group, which is mainly responsible for stabilization and rotation of the scapula, consists of the trapezius, **serratus anterior**, **levator scapulae**, and rhomboid muscles. These attach to the medial, superior, and inferior borders of the scapula. The
Re-ranked passage #2	and gives attachment to a few fibers of the **levator scapulae** muscle. The inferior angle of the scapula is the lowest part of the scapula and is covered by the latissimus dorsi muscle. It moves forwards round the chest when the arm is abducted. The inferior angle is formed by the union of the medial and lateral borders of the scapula. It is thick and rough and its posterior or back surface affords attachment to the teres major and often to a few fibers of the latissimus dorsi. The anatomical plane that passes vertically through the inferior angle
Re-ranked passage #3	to which the subscapularis muscle attaches. The medial two-thirds of the fossa have 3 longitudinal oblique ridges, and another thick ridge adjoins the lateral border; they run outward and upward. The ridges give attachment to the tendinous insertions, and the surfaces between them to the fleshy fibers, of the subscapularis muscle. The lateral third of the fossa is smooth and covered by the fibers of this muscle. At the upper part of the fossa is a transverse depression, where the bone appears to be bent on itself along a line at right angles to and passing through
Results of our method, Continued on the next page	

Table 3.2 – Results of our method, Continued from previous page

Re-ranked passage #4	”Spine of scapula” of the acromion, below with the neck of the scapula. It forms the medial boundary of the great scapular notch, which serves to connect the supra- and infraspinatous fossae. Spine of scapula The spine of the scapula or scapular spine is a prominent plate of bone, which crosses obliquely the medial four-fifths of the scapula at its upper part, and separates the supra- from the infraspinatous fossa. It begins at the vertical [vertebral or medial border] border by a smooth, triangular area over which the tendon of insertion of the lower part of the Trapezius glides, and, gradually becoming more
Re-ranked passage #5	small area of the medial border of the scapula at the level of the scapular spine. Together with the **rhomboid major**, the **rhomboid minor** retracts the scapula when trapezius is contracted. Acting as an antagonist to the trapezius, the rhomboid major and minor elevate the medial border of the scapula medially and upward, working in tandem with the **levator scapulae** muscle to rotate the scapulae downward. While other shoulder muscles are active, the rhomboid major and minor stabilize the scapula. The nerve supply comes from the dorsal scapular nerve, with most of its fibers derived from the C5 nerve root

3.6 Conclusion

We provide a Determinantal Point Processes-based re-ranking method to enhance the diversity of the retrieved passages. Unlike previous methods of re-ranking, our methodology does not need a significant amount of data since it is unsupervised. This is in contrast to the previous approaches ([35]). Instead, we depend on DPP to recognize the passages that are distinct from one another and most pertinent to the question. We compare our method to the state-of-the-art method and outperform it by 3% (top 5), 8% (top 10) for single-answer questions, and 18% (top5) and 21% (top10) for multi-answer retrieval on AmbigQA dataset.

Chapter 4

Reader mechanism utilizing pointer generator networks

4.1 Introduction

Natural language generation (NLG) has witnessed remarkable progress in recent years, enabling machines to produce coherent and contextually relevant text. One of the most challenging NLG tasks is question answering (QA), which demands the generation of human-like answers to user queries based on available documents or knowledge sources. Traditional QA systems have largely relied on predefined answer templates or rule-based approaches, which often fall short in capturing the nuances of natural language and handling diverse queries.

To address these limitations and pave the way for more sophisticated QA systems, we introduce a novel approach that harnesses the power of pointer generator networks. These networks, inspired by their success in abstractive summarization tasks, offer a promising avenue for improving the quality and fluency of generated answers.

The primary motivation behind the adoption of pointer generator networks in QA lies in their ability to dynamically select words or phrases from the input documents and queries, effectively overcoming the vocabulary limitations of predefined answer templates. This mechanism enables QA systems to generate answers that are not confined to a fixed set of responses but can adapt to the specific context and content of the question.

Furthermore, the need for refined answer generation techniques becomes evident when we consider the diversity and complexity of user queries. Traditional QA systems often struggle with questions that require a nuanced understanding of the document content or entail generating answers beyond the scope of simple extraction. Pointer generator networks equip QA systems to navigate through documents, extract relevant information, and compose answers that are contextually accurate and linguistically fluent.

In summary, we present a novel approach to question answering that leverages multi-pointer generator networks for enhanced answer generation. We aim to transform QA systems by solving document understanding and answer generation problems, resulting in more intuitive, accurate, and human-like responses.

4.2 Dataset

[40] introduced a dataset tailored for query-based abstractive summarization, albeit with a relatively limited corpus size, encompassing only 12,695 triples comprising documents, queries, and summaries. This modest dataset size poses challenges for training neural models effectively, leading to suboptimal performance in the domain of query-based summarization.

In response to the need for more extensive and informative datasets, [18] introduced the CNN/Dailymail dataset, which indeed offers a larger data pool. However, a notable caveat of this dataset is the brevity of queries, with an average length of a mere 1.52 words. This brevity raises concerns regarding the effective utilization of query information during model training, potentially resulting in generated summaries lacking meaningful contextual grounding.

The consequence of such succinct queries is the potential for generated summaries to lack the necessary context for a comprehensive understanding of the source material. Even for human annotators, deciphering the precise intent of a question and identifying the specific information sought from the passage becomes a challenging task in the presence of overly concise queries.

As a result, existing datasets for question-answering exhibit certain limitations, primarily in terms of dataset size and query expressiveness. These limitations underscore the need for more extensive and informative datasets to unlock the full potential of neural architectures in this domain.

In response to these challenges, we have introduced a novel dataset tailored specifically for the task of query-based abstractive summarization. Leveraging the MS Marco dataset ([41]), which has been meticulously annotated by human annotators for Machine Reading Comprehension tasks, we have curated a dataset comprising 1,010,916 queries, passages, and answers extracted from these passages. Notably, 182,669 samples within this dataset feature answers that have been rephrased and generated by human annotators after comprehensive examination of the passages and queries.

This dataset, comprising passages read by human annotators as source documents and corresponding questions as queries, serves as the foundation for query-based abstractive summarization. The responses thoughtfully generated by human annotators constitute abstract summaries. To facilitate comprehensive experimentation and evaluation, we have partitioned the dataset into three distinct subsets: 80% for

training, 10% for validation, and 10% for testing, ensuring a robust and systematic approach to model development and assessment.

4.3 Proposed Model

This section provides a high-level overview of 1) Sequence-to-Sequence Model with Attention and 2) Pointer generator methods for query-based summarisation.

4.3.1 Sequence-to-Sequence Model with attention

[53] proposed a sequence-to-sequence (Seq2Seq) paradigm. The Seq2Seq model is comprised of two main components: an encoder that processes the input sequence and generates a fixed-length vector concealed representation, and a decoder that takes the encoder's output and generates the output sequence token by token. The attention mechanism of seq2seq models enables the model to concentrate on specific parts of the input sequence while generating the output sequence.

These models are trained end-to-end. Seq2seq models can manage input and output sequences of variable lengths and can acquire sophisticated semantic understanding through the use of deep neural networks. The seq2seq models have numerous applications in natural language processing and computer vision, including Machine Translation, Speech Recognition, Text summarization, and conversational agents, among others.

The sequence-to-sequence with attention model for query-based answer generation is similar to the baseline model ([40]). The sequence to sequence with attention model consists of a document encoder, a query encoder, and a decoder. The passage encoder and query encoder compute hidden representations for the document and query, while the decoder decodes the hidden representations into a sequence of variable length generating the answer.

4.3.1.0.1 Query encoder A query encoder is a Seq2Seq component that accepts a query as input and generates a vector representation of the query with a fixed length. The query encoder is a single layer Bi-directional LSTM that reads each query word q_i from the query sentence sequentially and generates a hidden representation h_i^q .

$$h_i^q = \text{BiLSTM}(h_{i-1}^q, e(q_i)) \tag{4.1}$$

h_{t-1}^q is the hidden representation of the query word q_{t-1} at previous timestep.

4.3.1.0.2 Passage encoder A passage encoder is a Seq2Seq component that accepts a text passage as input and generates a vector representation of the passage with a fixed length. With the help of this vector representation, the semantic content of the passage is represented in a manner that makes it

simple to compare it to other passages or utilize it for downstream activities like question-response or information retrieval.

In natural language processing tasks such as reading comprehension, where the objective is to answer queries based on a given passage, passage encoders are frequently employed. In these tasks, the passage encoder accepts the passage's text as input and generates a vector representation that encapsulates the passage's essential semantic information.

Similar to query encoder, Passage encoder is also a single layer Bi-directional LSTM that reads the document words d_i sequentially to compute its hidden representation h_i^d

$$h_i^d = \text{BiLSTM}(h_{i-1}^d, e(d_i)) \tag{4.2}$$

h_{i-1}^q is the hidden representation of the document word d_{i-1} at timestep $i-1$.

4.3.1.0.3 Attention Mechanism for query and document The attention mechanism for query and passage is a technique in natural language processing (NLP) used to align and selectively weight different sections of a query and a passage during the answer of generating the answer.

In this mechanism, the input passage and query are initially encoded using neural networks such as LSTMs [19], [15], [7] or CNNs [60], [26], [11] to derive their vector representations. The query vector is then utilized to calculate the relevance score for each word or phrase in the text and the question. Each word or phrase in the paragraph is given a weight based on these relevance scores, indicating how significant it is to the query. This weight is used to compute a weighted sum of the passage vectors, resulting in a new vector representation that encapsulates the most pertinent information in the passage.

Attention mechanism helps the decoder to focus on the important parts of the input by giving weights to each input token and computing its representation vector. The query representation vector q_t is the weighted sum of query encoder hidden states. It is calculated as follows:

$$q_t = \sum_{i=1}^{k} \alpha_{t,i}^q h_i^q \tag{4.3}$$

The weight $\alpha_{t,i}^q$ for each h_i^q is calculated as follows

$$\alpha_{t,i}^q = \frac{\exp(a_{t,i}^q)}{\sum_{j=1}^{k} \exp(a_{t,j}^q)} \tag{4.4}$$

$$a_{t,i}^q = v_q^T \tanh(W_q s_t + U_q h_i^q) \tag{4.5}$$

To calculate the document representation d_t, we first compute the score using the following equations:

$$a_{t,i}^{d} = v_d^T \tanh(W_d s_t + U_d h_i^d + Z q_t) \tag{4.6}$$

The score is calculated by also considering the query representation q_t. This is to ensure that the decoder learns to pay attention to the query if it considers it to be essential. The scores are then normalized using a softmax layer to calculate the weights.

$$\alpha_{t,i}^{d} = \frac{\exp(a_{t,i}^{d})}{\sum_{j=1}^{n} \exp(a_{t,j}^{d})} \tag{4.7}$$

$$d_t = \sum_{i=1}^{n} \alpha_{t,i}^{d} h_i^d \tag{4.8}$$

4.3.1.0.4 Decoder The decoder generates an output sequence, typically one token at a time, by anticipating the next token based on the previously generated tokens and the input vector representation. At each time step, The decoder predicts y_t, the frequency distribution of each vocabulary word.

$$y_t = \text{softmax}(W^{'}(W[s_t, d_t] + b) + b^{'}) \tag{4.9}$$

The hidden state of decoder s_t is computed utilizing

$$s_t = \text{LSTM}_{dec}(s_{t-1}, [e(y_{t-1}), d_{t-1}]) \tag{4.10}$$

4.3.2 Pointer Generator model for query based summarisation

[51] introduced Pointer generator models for abstractive text summarization. A pointer generator model is a type of sequence-to-sequence (seq2seq) model that can generate novel tokens as well as copy tokens directly from the input sequence. The pointer generator model accomplishes this by integrating a pointer mechanism into the Seq2Seq architecture. Instead of creating a new token for a position, the pointer mechanism enables the model to copy tokens from the input sequence directly to the output sequence. We present several models for question answering employing a network of pointer generators, and we utilize a pointer generator network because -

1. The use of this technique ensures that the factual details in the summaries are accurate.
2. The answers may incorporate new words or entities that are considered important by the attention distribution, even if they are not included in the limited vocabulary.

4.3.2.0.1 PGD : Pointer generator applied to only document This is similar to the pointer generator model proposed by [51]. We compute the generation probability p_{gen}^{d} for the document, which determines the extent of vocabulary generation or document replication. The final distribution $P_{final}(w)$ is used to output the most probable word. The following equations calculate p_{gen} and $P_{final}(w)$.

$$p^d_{gen} = \sigma(w^T_d d_t + w^T_{s,d} s_t + w^T_{x,d} x_t + b_{ptr}) \tag{4.11}$$

$$P^d_{final}(w) = p^d_{gen} P(w) + (1 - p^d_{gen}) \sum_{i:w_i=w} \alpha^d_{t,i} \tag{4.12}$$

4.3.2.0.2 PGQ : Pointer Generator applied to only Query In this case, the document is disregarded when calculating the final vocabulary distribution. Using the following equations, we determine p^q_{gen} and $P_{final}(w)$.

$$p^q_{gen} = \sigma(w^T_q q_t + w^T_{s,q} s_t + w^T_{x,q} x_t + b'_{ptr}) \tag{4.13}$$

$$P^q_{final}(w) = p^q_{gen} P_{vocab}(w) + (1 - p^q_{gen}) \sum_{i:w_i=w} \alpha^q_{t,i} \tag{4.14}$$

4.3.3 Multi-pointer Generator for query based summarisation

Typical pointer-generator models are trained to generate an output token sequence from a single input token sequence. A multi-pointer generator model, on the other hand, is trained to produce output tokens from multiple input sources.

Inputs to a multi-pointer generator model may consist of multiple documents, queries, or other forms of contexts. The model is intended to learn how to attend to various input sources and produce output tokens that incorporate information from all sources. A multi-pointer generator is an extension of the pointer-generator model designed for text generation tasks that require more than one source of information. Each pointer is responsible for copying tokens from a particular input source, and the model learns how to use these pointers to combine information from multiple input sources when it is generating the output sequence.

Instead of applying Pointer generator to only the query or only the document, we hypothesized a model that can simultaneously point to words in both the query and the document.

4.3.3.0.1 MPG1: To calculate the final probability distribution, we average the probability distributions derived by applying the pointer generator to just the document and the query, eqn12 & 14.

$$P_{final}(w) = \frac{P^d_{final}(w) + P^q_{final}(w)}{2} \tag{4.15}$$

4.3.3.0.2 MPG2: In this model, we utilize the final distribution from the pointer generator on the document to generate words with a probability p^q_{gen}.

$$P_{final}(w) = p^q_{gen} P^d_{final}(w) + (1 - p^q_{gen}) \sum_{i:w_i=w} \alpha^q_{t,i} \tag{4.16}$$

4.3.3.0.3 MPG3: Along the same lines as the model described above, we use the final distribution from the pointer generator on the query, and we generate words with a probability p^d_{gen} .

$$P_{final}(w) = p^d_{gen} P^q_{final}(w) + (1 - p^d_{gen}) \sum_{i:w_i=w} \alpha^d_{t,i} \tag{4.17}$$

4.3.4 Loss function

Loss for timestep t during training is the negative log likelihood of the target word w^*_t for that timestep: The negative log-likelihood of an event or a set of events is essentially a measure of how well a probability distribution or a model's predictions align with the observed data. It's simply the negative natural logarithm of the likelihood. At each time step t, the goal is to find the model parameters (θ) that maximize the likelihood (or equivalently, minimize the negative log-likelihood).

$$loss_t = -\log P(w^*_t) \tag{4.18}$$

4.4 Experiments

In our study, we performed a series of experiments using the dataset outlined in Section 4.3.2. To evaluate our approach, we conducted a comparative analysis with the [18] model, which employs a pointer generator switch exclusively on the document within the CNN dataset.

1. **Seq2seq with Attention (Seq2seq with attn):** This serves as our baseline model, and its details are elaborated in 4.3.1.

2. **Pointer Generator on Document (PGD):** In this model, we apply the pointer generator solely to the document, as discussed in Section 4.3.2.0.1.

3. **Pointer Generator on Query (PGQ):** Here, the pointer generator is exclusively applied to the query, as outlined in 4.3.2.0.2.

4. **Multi Pointer Generator 1 (MPG1):** In this model, the final vocabulary distribution is the average of final distributions obtained by applying pointer generator to document and to query as explained in 4.3.3.0.1

5. **Multi Pointer Generator 2 (MPG2):** In this model, the resulting vocabulary distribution obtained after applying pointer generator to document is used to generate words with a probability p_{gen} as explained in 4.3.3.0.2

6. **Multi Pointer Generator 3 (MPG1):** In this model, pointer generator is first applied to query and then used as vocabulary distribution to compute the final vocab distribution as explained in 4.3.3.0.3

To assess the effectiveness of these models, we employ the ROUGE-1, ROUGE-2, and ROUGE-L metrics [31], [32], [43] for evaluation purposes. These metrics provide us insights into the quality and performance of various methods.

4.4.1 Embeddings

Word embeddings, also known as word vectors, provide a compact, continuous, and low-dimensional representation of words while preserving their semantic and syntactic characteristics. Essentially, these are real-valued vectors that encode the relationships between words by their spatial proximity in the vector space. Word vectors, or word embeddings, construct a dense vector for each word, carefully designed to be akin to vectors of words found in comparable contexts.

Character embeddings are a sort of word representation that encodes individual characters of a word into continuous, fixed-length vectors. Traditional word embeddings, such as Word2Vec and GloVe, represent complete words as vectors, whereas character embeddings concentrate on the individual characters.

Because the character embeddings are built on characters, they may represent words that are not present in the pre-trained word embeddings. This is very beneficial when dealing with uncommon or Out-of-Vocabulary (OOV) words. Morphological information and subword structures are captured by character embeddings, which may be especially useful in languages with complicated inflections or agglutinative languages.

In our approach, we employed an embedding layer to convert individual characters within words into word vectors, followed by their input into a single Long Short-Term Memory (LSTM) layer. These word vectors were derived from pre-trained 300-dimensional GloVe embeddings[44], [34], which capture semantic information about words based on extensive text corpora. Importantly, we combined these word embeddings with character embeddings and incorporated them into our models.

The inclusion of character embeddings proved to be a valuable enhancement for our models. One notable advantage was the model's capability to generate accurate word vectors for out-of-vocabulary words, ensuring effective handling of uncommon or previously unseen terms. This capability significantly contributed to the improved performance observed across all our model variants.

4.4.2 Experimental details

In our experimental setup, the choice of vocabulary sizes played a crucial role in optimizing model performance. With a word vocabulary size of 50000, we ensured that our model could effectively handle a wide range of words and their associations. Additionally, we defined a character vocabulary size of

Query: Is measles and mumps contagious?
Content: measles is a serious and highly contagious viral disease which causes fever runny nose cough and sore red eyes followed by a rash measles can sometimes lead to dangerous complications such as pneumonia about one person in hhh who contracts measles will develop inflammation of the brain easles mumps and rubella mmr vaccines the mmr vaccine also comes in combination with chickenpox mmrv for ## month old children and contains small amounts of each of the viruses at a reduced strength and a small amount of the antibiotic neomycin
Original Summary: yes measles and mumps are contagious
seq2seqwattn: measles is a serious and highly contagious viral disease which causes fever runny nose cough and sore **PGD:** measles and rubella vaccines are measles type of measles **PGQ:** no measles and mumps is not contagious **MPG1:** no measles and mumps is not contagious **MPG2:** no measles and mumps is not contagious **MPG3:** yes measles and mumps is contagious
Query: What family is the centipede in ?
Content: Scolopendra (through Latin from Greek skolopendra) is a species-rich genus of often very large centipedes of the family Scolopendridae.
True summary: the centipede is in the scolopendridae family
seq2seqwattn: [UNK] is genus of family **pgd:** scolopendra is the scolopendra in latin **pgq:** the centipede is in greek family **mpg1:** scolopendra is the centipede in the family **mpg2:** centipedes is the centipede in scolopendridae **mpg3:** the centipede is in scolopendra family

Table 4.1 Sample results of various models on our dataset

28 to capture the nuances of individual characters, enhancing the model's ability to process words at a finer level of detail.

Throughout our experiments, we maintained a consistent batch size of 30, allowing us to efficiently process and train our models. To manage the complexity of the task, we imposed limits on the length of documents, queries, and summaries. Specifically, we constrained documents to 120 words, queries to 20 words, and summaries to 20 words, enabling effective data processing and summary generation.

Character embeddings played a pivotal role in our model architecture, as they provided a compact 20-dimensional representation of characters. This representation facilitated the model's ability to capture subword structures and morphological information, contributing to improved summarization results. Additionally, the hidden size for the LSTM was set to 50, striking a balance between model complexity and computational efficiency.

Models	ROUGE-1	ROUGE-2	ROUGE-L
[18]	18.25	5.04	16.17
MPG3	**21.36**	**7.62**	**19.09**

Table 4.2 Results on CNN/Dailymail dataset.

For consistency in our network architecture, we maintained a hidden size of 400 across all layers. This uniformity ensured seamless information flow and feature extraction throughout the network, ultimately leading to more coherent and contextually accurate summaries.

In terms of optimization, we employed the Adagrad optimizer with a learning rate of 0.15. This allowed us to effectively update model parameters during training, leading to faster convergence and improved performance.

Finally, during the summary generation phase, we employed a beam size of 4 for decoding. This beam search strategy enabled the model to explore multiple possible summaries and select the most contextually relevant and coherent one, further enhancing the quality of our generated summaries.

Our model's learning process was significantly impacted by the choice to use teacher forcing. By consistently supplying the decoder with the ground truth target output at each time step, we not only facilitated stable training but also expedited the convergence of our model. This approach nurtured the model's ability to learn accurate sequence-to-sequence mappings, resulting in more coherent and contextually accurate summaries.

To effectively manage the training process and ensure that it remains within well-defined bounds, we set a maximum iteration limit of 50,000. This constraint served as a safeguard against overfitting and allowed us to strike a balance between model complexity and training time. The rigorous adherence to this iteration limit contributed to the model's robust generalization and reliable performance. We also applied the same parameters on cnn/dailymail dataset.

4.5 Results

In Table 4.2, we present the ROUGE-1, ROUGE-2, and ROUGE-L scores for two different models on the CNN/Dailymail dataset. The models under consideration are [18] and "MPG3."

Experiments	ROUGE-1	ROUGE-2	ROUGE-L
seq2seq+attn	59.88	41.73	56.24
PGD	62.64	44.98	58.63
PGQ	69.79	53.98	65.80
MPG1	68.49	51.88	64.31
MPG2	69.05	53.85	64.91
MPG3	**70.24**	**55.01**	**66.03**

Table 4.3 Results on our dataset.

Comparing the results, it is evident that the MPG3 model outperforms the [18] model across all three ROUGE metrics. Specifically, MPG3 achieves a ROUGE-1 score of 21.36, a ROUGE-2 score of 7.62, and a ROUGE-L score of 19.09, which are all higher than the corresponding scores achieved by the Hassalvist model. These findings highlight the superiority of the MPG3 model in generating summaries that exhibit better overlap with reference summaries, both in terms of unigrams, bigrams, and the longest common subsequence.

Overall, the results on the CNN/Dailymail dataset suggest that the MPG3 model excels in the task of abstractive summarization when compared to the [18] model, showcasing its effectiveness in generating high-quality summaries.

In Table 4.3, We present the ROUGE-1, ROUGE-2, and ROUGE-L scores for various summarization experiments. These experiments include the baseline seq2seq+attn model and several variations of the model with pointer generator techniques (PGD, PGQ) and multi-pointer generator methods (MPG1, MPG2, MPG3). These scores serve as metrics to evaluate the quality of the generated summaries.

Upon comparing the experiments, it is evident that the MPG3 variant outperforms the other models across all three ROUGE metrics. With a ROUGE-1 score of 70.24, a ROUGE-2 score of 55.01, and a ROUGE-L score of 66.03, MPG3 consistently generates summaries that exhibit the highest overlap with the reference summaries, both in terms of unigrams and bigrams, as well as the longest common subsequence.

While PGQ also shows strong performance, particularly in terms of ROUGE-2, MPG3 emerges as the top-performing configuration in this comprehensive analysis. These results emphasize the effectiveness of multi-pointer generator networks in enhancing the quality of answer-generation tasks and specifically highlight the MPG3 model as the most promising choice among the evaluated models.

Comparing the experiments, we observe that the MPG3 variant consistently outperforms the other models across all three ROUGE metrics. It achieves the highest ROUGE-1 score of 70.24, the highest

ROUGE-2 score of 55.01, and the highest ROUGE-L score of 66.03. This indicates that the MPG3 model generates summaries that better capture the overlap of unigrams and bigrams with reference summaries while also preserving the longest common subsequence.

4.5.1 Conclusion

In this paper, we presented a novel multi-pointer generator framework designed to address the challenge of contextually relevant summary generation in the context of query-based summarization. We introduce various techniques to effectively combine the query pointer module with the passage pointer module, enhancing the synthesis of relevant content for summary creation. To address the existing dearth of datasets tailored specifically for query-based answer generation, we contribute by adapting a Reading Comprehension dataset to create a specialized dataset for this purpose. Our model's performance surpasses that of state-of-the-art models on existing datasets, showcasing a substantial 3.5-point improvement in Rouge scores. This improvement underscores the model's efficacy in generating high-quality summaries that are tailored to query-based contexts, marking a significant advancement in query-based summarization research with implications for natural language understanding and information retrieval.

Chapter 5

Conclusion

In this thesis, we have demonstrated a retriever-reader model for end-to-end open-domain question answering. We presented a re-ranking model to retrieve passages for a question. Our main goal was to retrieve diverse yet relevant passages, hence we employed probabilistic models called Determinantal Point Processes which encode the pairwise similarities and dissimilarities between the passages. If a passage is selected to be part of the re-ranked set, passages similar to this passage would have less probability of getting selected. This allows our re-ranked passage set to always contain dissimilar yet query-relevant passages. We compare our method to the state-of-the-art method and outperform it by 3% (top 5), 8% (top 10) for single-answer questions, and 18%(top5) and 21% (top10) for multi-answer retrieval on AmbigQA dataset.

We have also discussed a reader model to generate answers on the MS MARCO dataset utilizing pointer generator networks, sequence-to-sequence models with attention and Bidirectional Long Short term Memory Network(Bi-LSTM) models. We introduce various techniques to effectively combine the query pointer module with the passage pointer module, enhancing the synthesis of relevant content for summary creation. Our model's performance surpasses that of state-of-the-art models on existing datasets, showcasing a substantial 3.5-point improvement in Rouge scores.

Chapter 6

Future Work

In the context of our future research work, we envision the integration of our diverse passage retrieval system as an integral component of the Retrieval Augmented Generation module within Large Language Models (LLMs). This initiative represents a significant advancement in the field of natural language processing, with implications for improved information retrieval and question-answering systems.

Our diverse passage retrieval system has demonstrated its ability to efficiently extract a wide array of pertinent information from extensive corpora. By seamlessly incorporating this retrieval mechanism into the Retrieval Augmented Generation module, we could create a comprehensive framework for addressing complex language understanding and generation tasks for ambiguous questions.

In this proposed framework, we propose employing an LLM as a reader, facilitating an end-to-end question-answering approach. This reader LLM will be responsible for comprehending user queries and generating responses by making informed selections from the retrieved passages. This approach streamlines the process and holds the potential to yield responses that are contextually relevant and coherent.

Additionally, we intend to utilize an LLM-based scoring mechanism as an evaluator to rerank the retrieved passages. Leveraging the language understanding capabilities of LLMs, this evaluator aims to enhance the quality and relevance of retrieved information.

In summary, the integration of our diverse passage retrieval system into the Retrieval Augmented Generation module of LLMs constitutes a promising avenue for future research.

Related Publications

[39] Poojitha Nandigam, Nikhil Rayaprolu, and Manish Shrivastava. 2022. Diverse Multi-Answer Retrieval with Determinantal Point Processes. In Proceedings of the 29th International Conference on Computational Linguistics, pages 2220–2225, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Bibliography

[1] G. AlRegib and A. K. Katsaggelos. Determinantal point processes on graphs for image segmentation. *IEEE Transactions on Image Processing*, 23(4):1778–1793, 2014.

[2] T. B. Brown and et al. Language models are unsupervised multitask learners. *arXiv preprint arXiv:2005.14165*, 2020.

[3] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading Wikipedia to answer open-domain questions. In *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, volume 1, pages 1870–1879, mar 2017.

[4] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051, 2017.

[5] L. Chen, G. Zhang, and E. Zhou. Fast greedy map inference for determinantal point process to improve recommendation diversity. *Advances in Neural Information Processing Systems*, 31, 2018.

[6] C. Cherry and G. Kondrak. Determinantal point processes for machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[7] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[8] S. Cho, L. Lebanoff, H. Foroosh, and F. Liu. Improving the similarity measure of determinantal point processes for extractive multi-document summarization. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 1027–1038, 2020.

[9] S. Cho, C. Li, D. Yu, H. Foroosh, and F. Liu. Multi-document summarization with determinantal point processes and contextualized representations. *arXiv*, 2019.

[10] C. Clark and M. Gardner. Simple and effective multi-paragraph reading comprehension. *arXiv preprint arXiv:1710.10723*, 2017.

[11] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] W. H. Gomaa, A. A. Fahmy, et al. A survey of text similarity approaches. *international journal of Computer Applications*, 68(13):13–18, 2013.

[14] C. Goutte and F.-M. De Rainville. Determinantal point processes for machine summarization. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 2013.

[15] A. Graves, N. Jaitly, and A.-r. Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pages 273–278. IEEE, 2013.

[16] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909, 2020.

[17] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. W. Chang. REALM: Retrieval-Augmented language model pre-training. In *37th International Conference on Machine Learning, ICML 2020*, volume PartF16814, pages 3887–3896, feb 2020.

[18] J. Hasselqvist, N. Helmertz, and M. Kågebäck. Query-based abstractive summarization using neural networks. *arXiv preprint arXiv:1712.06100*, 2017.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[20] G. Izacard and E. Grave. Distilling knowledge from reader to retriever for question answering, 2020.

[21] G. Izacard and E. Grave. Leveraging passage retrieval with generative models for open domain question answering, 2020.

[22] G. Izacard, F. Petroni, L. Hosseini, N. D. Cao, S. Riedel, and E. Grave. A memory efficient baseline for open domain question answering, 2020.

[23] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

[24] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense retrieval is all you need for open domain question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[25] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. tau Yih. Dense passage retrieval for open-domain question answering. 4 2020.

[26] Y. Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

[27] A. Kulesza and B. Taskar. Determinantal point processes for machine learning. jul 2012.

[28] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, M. Kelcey, J. Devlin, K. Lee, K. N. Toutanova, L. Jones, M.-W. Chang, A. Dai, J. Uszkoreit, Q. Le, and S. Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

[29] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[30] L. Li, W. Liu, M. Litvak, N. Vanetik, and Z. Huang. In conclusion not repetition: Comprehensive abstractive summarization with diversified attention based on determinantal point processes. *CoNLL 2019 - 23rd*

Conference on Computational Natural Language Learning, Proceedings of the Conference, pages 822–832, 2019.

[31] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[32] C.-Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 human language technology conference of the North American chapter of the association for computational linguistics*, pages 150–157, 2003.

[33] J. Lin, X. Ma, S.-C. Lin, J.-H. Yang, R. Pradeep, and R. Nogueira. Pyserini: An easy-to-use python toolkit to support replicable ir research with sparse and dense representations. *arXiv preprint arXiv:2102.10073*, 2021.

[34] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality, 2013.

[35] S. Min, K. Lee, M.-W. Chang, K. Toutanova, and H. Hajishirzi. Joint passage ranking for diverse multi-answer retrieval. *arXiv preprint arXiv:2104.08445*, 2021.

[36] S. Min, J. Michael, H. Hajishirzi, and L. Zettlemoyer. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online, Nov. 2020. Association for Computational Linguistics.

[37] S. Min, J. Michael, H. Hajishirzi, and L. Zettlemoyer. Ambigqa: Answering ambiguous open-domain questions. 4 2020.

[38] S. Mohamed and Z. Ghahramani. Fast determinantal point process sampling with application to clustering. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014.

[39] P. Nandigam, N. Rayaprolu, and M. Shrivastava. Diverse multi-answer retrieval with determinantal point processes, 2022.

[40] P. Nema, M. M. Khapra, A. Laha, and B. 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.

[41] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng. Ms marco: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*, 2016.

[42] R. Nogueira and K. Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

[43] K. Papineni, S. Roukos, T. Ward, and W.-J. 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.

[44] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[45] A. Piktus, L. Alt, D. Mirylenka, J. Michael, and T. Rocktäschel. Rag: Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

[46] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[47] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[48] S. Robertson, H. Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[49] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109:109, 1995.

[50] H. Schütze, C. D. Manning, and P. Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

[51] A. See, P. J. Liu, and C. 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.

[52] A. Sharghi, J. S. Laurel, and B. Gong. Query-focused video summarization: Dataset, evaluation, and a memory network based approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4788–4797, 2017.

[53] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[54] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[55] Z. Wang, P. Ng, X. Ma, R. Nallapati, and B. Xiang. Multi-passage bert: A globally normalized bert model for open-domain question answering. *arXiv preprint arXiv:1908.08167*, 2019.

[56] P. Yang, H. Fang, and J. Lin. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256, 2017.

[57] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin. End-to-end open-domain question answering with BERTserini. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Demonstrations Session*, pages 72–77, feb 2019.

[58] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49, 1999.

[59] X. Zhang, N. Razavian, and J. Solomon. Random determinantal point processes for scalable sparse regression. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015.

[60] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

[61] F. Zhu, W. Lei, C. Wang, J. Zheng, S. Poria, and T.-S. Chua. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv preprint arXiv:2101.00774*, 2021.