Dense Passage Retriever

Implementing a state-of-the-art Open Domain Question Answering System

A.Rishith Reddy-210020001, M.Siva mohan-210020025, A.V.S.Sreenivasu-210020001, Akash Anand-210030002

***Abstract*—Open-domain question answering (QA) systems aim to provide accurate responses to factoid questions by leveraging vast document collections. Traditional QA approaches involve complex systems with multiple components, but recent advancements in reading comprehension models have simplified the process into a two-stage framework. First, a context retriever selects relevant passages, followed by a machine reader to extract answers. Despite this simplification, performance degradation is observed, highlighting the need for improved retrieval methods. Typically, retrieval employs sparse representations like TF-IDF or BM25, while dense embeddings offer a complementary approach, capturing latent semantic similarities. However, dense retrieval methods often require large labeled datasets for effective training and have not consistently outperformed traditional methods. This paper addresses this challenge by proposing a Dense Passage Retriever (DPR) trained using pairs of questions and passages without additional pretraining. Leveraging BERT and a dual-encoder architecture, our approach optimizes embeddings to maximize inner products between question and passage vectors. By incorporating modern reader models into the retrieval process, we achieve competitive or superior results across various QA datasets in the open-retrieval setting.**

***Index Terms*—Dense Passage Retriever, Dual-Encoder Model, Open-Domain QA, BERT-encoder**

I. Introduction

Open-domain question answering (QA) is a challenging task in natural language processing (NLP) that aims to develop systems capable of comprehending and responding to questions across a broad range of topics, drawing information from vast knowledge sources without relying on pre-defined databases or domain-specific contexts. Unlike traditional QA systems that operate within constrained domains or structured databases, open-domain QA requires a deeper understanding of language and the ability to retrieve relevant information from unstructured textual data available on the internet or other large-scale repositories.

The goal of open-domain QA systems is to mimic human-like comprehension and reasoning, enabling them to effectively respond to a wide variety of questions by synthesizing information from diverse sources. These systems must navigate through the complexities of language, including ambiguity, inference, and context, to provide accurate and informative answers.

Key challenges in open-domain QA include information retrieval, where the system must efficiently locate and extract relevant passages or documents from vast corpora; natural language understanding, involving the interpretation of questions and context; and answer synthesis, which requires generating concise and accurate responses based on retrieved information.

In recent years, advancements in machine learning, particularly in the fields of deep learning and representation learning, have led to significant progress in open-domain QA. Techniques such as transformer-based models and dense vector representations have improved the ability of systems to capture semantic relationships and contextual information, enhancing their performance on complex language tasks.

Overall, open-domain QA represents a frontier in NLP research, with applications ranging from virtual assistants and search engines to information retrieval systems and automated customer support. Continued innovation in this area promises to unlock new capabilities in human-computer interaction and information access, pushing the boundaries of what AI systems can achieve in understanding and processing natural language.

A. Related Works

Traditional sparse vector space models like TF-IDF (Term Frequency-Inverse Document Frequency) and BM25 (Best Matching 25) have historically served as fundamental tools in open-domain question answering (QA) systems. These models compute document relevance based on term frequencies and matching scores with the query. While efficient, TF-IDF and BM25 have limitations in capturing semantic nuances and contextual understanding required for complex QA tasks involving unstructured data. As a result, recent research has focused on exploring dense retrieval methods to enhance the performance of QA systems beyond the capabilities of these traditional sparse models.

Recent research in open-domain question answering (QA) has focused on advancing dense retrieval methods to surpass traditional sparse vector space models like TF-IDF and BM25. Historically, dense retrieval techniques were believed to require extensive labeled data, making them less competitive compared to sparse methods. However, breakthroughs like ORQA (Lee et al., 2019) have demonstrated the potential of dense representations in achieving superior performance. ORQA introduced the Inverse Cloze Task (ICT) objective for pretraining, enhancing dense vector representations by predicting blocks containing masked sentences. By jointly fine-tuning the question encoder and reader model using question-answer pairs, ORQA achieved state-of-the-art results across various open-domain QA datasets. Despite its success, ORQA faces computational challenges due to the intensive nature of ICT pretraining and uncertainties about the effectiveness of regular sentences as surrogates for questions within the objective function.

B. Motivation

Dense passage retrieval offers several compelling motivations in the realm of open-domain question answering (QA), especially when contrasted with traditional sparse retrieval methods like TF-IDF or BM25.

- **Semantic Understanding**: Dense encodings capture semantic information more effectively compared to sparse representations. They can handle synonyms, paraphrases, and contextually similar phrases more efficiently. For instance, in the example **”Who is the bad guy in lord of the rings?”**, ”bad guy” and ”villain” are conceptually related but may not share exact lexical similarity. Dense representations can bridge such semantic gaps, leading to more accurate retrieval.
- **Improved Context Matching:** Sparse representations may struggle with nuanced context matching, especially in cases where keyword overlap is limited. Dense encodings excel in capturing the nuanced relationships between words and phrases, enabling more precise matching of query and context. In the given example, a dense retrieval system can recognize the semantic equivalence between ”bad guy” and ”villain” to fetch the relevant passage accurately.
- **Task-Specific Representation:** Dense encodings offer flexibility in tailoring representations to specific tasks. By adjusting embedding functions during training, these representations can be fine-tuned to optimize performance for the given QA task. This adaptability ensures that the retrieval system can better understand and respond to the nuances of different queries and contexts.
- **Efficient Retrieval:** Dense passage retrieval can leverage specialized in-memory data structures and indexing schemes, along with maximum inner product search (MIPS) algorithms, to achieve efficient retrieval. While initially dense retrieval might seem computationally intensive, efficient algorithms and indexing techniques ensure that it remains scalable even for large-scale QA systems.

In summary, dense passage retrieval addresses the limitations of sparse methods by offering superior semantic understanding, improved context matching, task-specific adaptability, and efficient retrieval techniques. These advantages make it a compelling choice for enhancing the performance of open-domain QA systems.

II. Dense Passage Retriever

The Dense Passage Retriever (DPR) is a crucial component in open-domain question answering (QA) systems, focusing on efficiently retrieving relevant passages from a large collection of text passages. In essence, DPR aims to map both the input question and the passages into a low-dimensional continuous space, allowing for quick retrieval of the most relevant passages based on their similarity to the question. When faced with a sizable collection of M text passages, DPR’s objective is to index all these passages into a low-dimensional and continuous space. This facilitates efficient retrieval of the top k passages that are most relevant to the input question for the reader during runtime. It’s noteworthy that the scale of M can be substantial, as evidenced by experiments involving up to 21 million passages , while k typically remains relatively small, ranging from 20 to 100.

Here’s an overview of how DPR works:

- **Indexing Passages**: DPR utilizes a dense encoder (denoted as EP()) to map each text passage into a d-dimensional real-valued vector. These vectors serve as representations of the passages, allowing for efficient indexing. Even with a large collection of passages (M), DPR builds an index for all passages.
- **Question Encoding**: At runtime, DPR applies a different encoder (EQ()) specifically designed to map the input question to a d-dimensional vector. This vector representation captures the semantic meaning of the question.
- **Retrieval Process**: To find the most relevant passages for a given question, DPR computes the similarity between the question vector and the passage vectors using the dot product operation. The similarity function, sim(q, p), is defined as the dot product of the question vector and the passage vector: sim(q, p) = EQ(q) • EP(p).
- **Choice of Similarity Function**: While various forms of similarity functions exist, DPR opts for the inner product function due to its simplicity and effectiveness. Inner product search has been widely studied and connects to other similarity metrics like cosine similarity and L2 distance.
- **Encoders**: DPR employs two independent BERT networks (base, uncased) as encoders for both questions and passages. The representations at the [CLS] token are used as the output, resulting in d = 768-dimensional vectors.
- **Inference**: During inference, DPR applies the passage encoder (EP) to all passages offline and indexes them using FAISS, a highly efficient library for similarity search and clustering of dense vectors. When a question is posed, its embedding (vq = EQ(q)) is derived, and the top k passages with embeddings closest to vq are retrieved.

Overall, DPR’s design enables efficient retrieval of relevant passages in open-domain QA scenarios, even when dealing with a vast collection of passages. By leveraging dense vector representations and efficient indexing techniques, DPR significantly improves the performance and speed of open-domain QA systems.

A. Dataset

The Natural Questions (NQ) dataset is a large-scale benchmark dataset developed by Google Research for training and evaluating question-answering systems. It contains real user queries sourced from the internet, paired with relevant passages from Wikipedia articles that may contain the answer. The dataset is challenging due to the inclusion of lengthy context passages and diverse question types, including both extractive and ”yes/no” answer formats. This dataset serves as a standard benchmark for evaluating the performance and robustness of question-answering systems across various domains and scenarios, contributing to advancements in natural language processing and information retrieval.In this experiment we used a subset of the NQ Dataset for training and testing convenience.

B. Dual Encoder Model

The dual encoder model employs two BERT (Bidirectional Encoder Representations from Transformers) models: one for encoding questions and another for encoding passages. BERT is a transformer-based language model that utilizes bidirectional attention mechanisms to capture contextual information from text. In the dual encoder setup, the question encoder processes input questions to generate dense representations, while the passage encoder encodes candidate passages. These encoded representations are then used to compute similarity scores between questions and passages, enabling effective retrieval of relevant information in open-domain question answering tasks. This architecture leverages BERT's capabilities in understanding complex language patterns and contextual relationships to enhance the accuracy and efficiency of information retrieval.

C. ***Training***

Dataset Preparation:

- We define a custom dataset class **QADataset** to handle pairs of questions and passages. This class tokenizes questions and passages using a **BERT** tokenizer and returns tokenized inputs suitable for the model.

Model Definition:

- We define a **DualEncoderModel** class, which serves as the architecture for our QA model. This class consists of two **BERT** encoders: one for questions and another for passages. During the forward pass, it encodes both questions and passages and extracts the pooled output embeddings of the **[CLS]** token.

Triplet Loss Function:

- We implement the **triplet_loss** function, which computes the triplet loss between anchor, positive, and negative embeddings. The loss encourages the model to minimize the distance between anchor and positive embeddings while maximizing the distance between anchor and negative embeddings.

  The triplet loss function is computed as follows:

  $$\mathcal{L}(a, p, n) = \max\left(d(a, p) - d(a, n) + \text{margin}, 0\right)$$

  where:
  - a is the anchor embedding,
  - p is the positive embedding (same class as anchor),
  - n is the negative embedding (different class from anchor),
  - $d(x, y)$ is a distance metric (e.g., Euclidean distance or cosine similarity) between embeddings x and y,
  - margin is a hyperparameter that controls the minimum difference between positive and negative distances.

  The triplet loss function encourages the model to minimize the distance $d(a, p)$ between the anchor and positive embeddings while simultaneously maximizing the distance $d(a, n)$ between the anchor and negative embeddings.

Model Initialization:

- We initialize **BERT** models for question encoding (**question_encoder**) and passage encoding (**passage_encoder**) using the **bert-base-uncased** pre-trained weights. These models are then incorporated into the **DualEncoderModel**.

Training Setup:

- We initialize an **Adam** optimizer with a learning rate of **1e-5** and set the number of epochs and margin for triplet loss.
- We define a dataset using the **QADataset** class and create a data loader to iterate over batches of data during training.

Training Loop:

- We iterate over each epoch and batch of data, computing question and passage embeddings using the model.
- Negative embeddings are created by rolling the passage embeddings to introduce negative examples for triplet loss.
- Triplet loss is computed and backpropagated to update model parameters.
- Progress is monitored using **tqdm**, a progress bar library, displaying the loss for each batch.

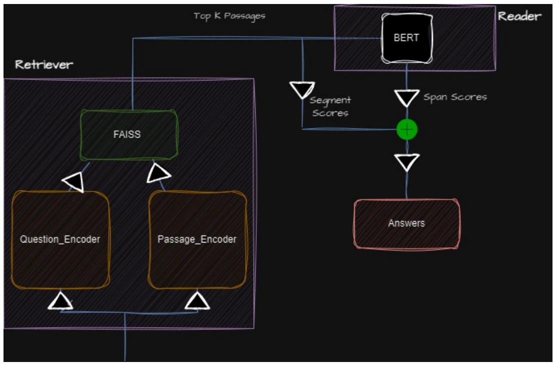


Fig. 1. Open Domain Question Answering with Dual Encoder Retriever

However during training we encountered some issues with loss function in fine-tuning of BERT encoder. This we later realized was an issue with using the wrong loss function. While a part of the team was in pursuit of fixing this issue, the other part tried alternatives and stumbled on SBERT which proved a very valuable asset as well.

III. Reader Model

Tokenization and Chunking:

- **Tokenization**: The question and context are tokenized using a BERT tokenizer (**tokenizer.encode**). This converts text into numerical token IDs suitable for model input.

- **Chunking**: The context is divided into chunks of a specified size (**chunk_size**) to handle large contexts efficiently.

Model Prediction:

- **Model Forward Pass**: For each chunk of the context (combined with the question), the model is invoked (**model(input_ids)**) to obtain predictions.
- The model outputs include **start_logits** and **end_logits**, representing scores for potential start and end positions of the answer within the context.

Top Answer Extraction:

- From the **start_logits** and **end_logits**, indices corresponding to the top N scoring positions are identified (**possible_starts** and **possible_ends**).
- The answer spans (start and end positions) corresponding to these top scoring indices are extracted.

Answer Post-Processing:

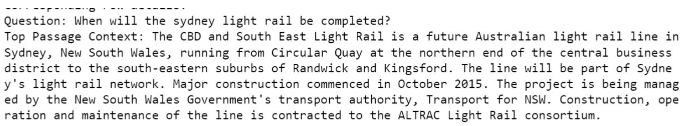
- **Answer Conversion**: The identified answer spans are converted back from token IDs to a string representation (**tokenizer.convert_tokens_to_string**).
- **Irrelevant Answer Filtering**: Extracted answers are filtered to retain only those containing relevant information related to the question (**filter_irrelevant_answers**).

IV. Results

In this section, we present the results of our experimentation, including comparisons and examples of our retriever and reader models.The model has been trained and tested on subsets of NQ dataset and the result will likely improve with larger dataset and more rigorous training.

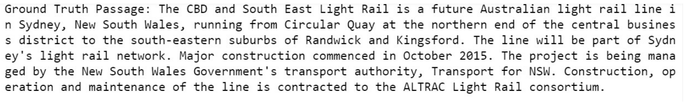
A. Retriever Model Evaluation

We evaluate the performance of our retriever model by comparing the recommended context to the ground truth context. Figure 2 illustrates this comparison.



```
corresponding row details:
Question: When will the sydney light rail be completed?
Top Passage Context: The CBD and South East Light Rail is a future Australian light rail line in
Sydney, New South Wales, running from Circular Quay at the northern end of the central business
district to the south-eastern suburbs of Randwick and Kingsford. The line will be part of Sydne
y's light rail network. Major construction commenced in October 2015. The project is being manag
ed by the New South Wales Government's transport authority, Transport for NSW. Construction, ope
ration and maintenance of the line is contracted to the ALTRAC Light Rail consortium.
```

Fig. 2. Comparison of Context Recommended by Retriever to Ground Truth Context - Retriever Output



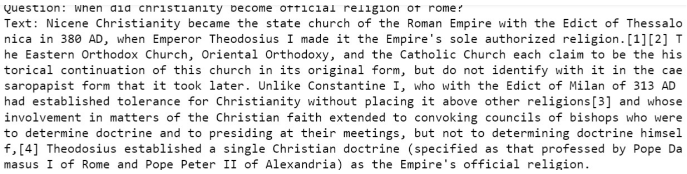
```
Ground Truth Passage: The CBD and South East Light Rail is a future Australian light rail line i
n Sydney, New South Wales, running from Circular Quay at the northern end of the central busines
s district to the south-eastern suburbs of Randwick and Kingsford. The line will be part of Sydn
ey's light rail network. Major construction commenced in October 2015. The project is being mana
ged by the New South Wales Government's transport authority, Transport for NSW. Construction, op
eration and maintenance of the line is contracted to the ALTRAC Light Rail consortium.
```

Fig. 3. Comparison of Context Recommended by Retriever to Ground Truth Context - Ground Truth

More on accuracy will of the retriever is discussed in upcoming subsections

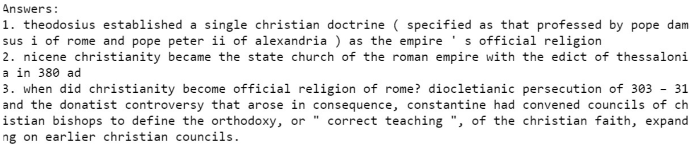
B. Reader Model Evaluation

To showcase the efficacy of our reader model, we provide an example of a question, context, and the extracted answer. Figure 4 demonstrates this example.



```
Question: When did christianity become official religion of rome?
Text: Nicene Christianity became the state church of the Roman Empire with the Edict of Thessalo
nica in 380 AD, when Emperor Theodosius I made it the Empire's sole authorized religion.[1][2] T
he Eastern Orthodox Church, Oriental Orthodoxy, and the Catholic Church each claim to be the his
torical continuation of this church in its original form, but do not identify with it in the cae
saropapist form that it took later. Unlike Constantine I, who with the Edict of Milan of 313 AD
had established tolerance for Christianity without placing it above other religions[3] and whose
involvement in matters of the Christian faith extended to convoking councils of bishops who were
to determine doctrine and to presiding at their meetings, but not to determining doctrine himsel
f,[4] Theodosius established a single Christian doctrine (specified as that professed by Pope Da
masus I of Rome and Pope Peter II of Alexandria) as the Empire's official religion.
```

Fig. 4. Question, Context, and Extracted Answer using Reader Model - Retrieved Passage

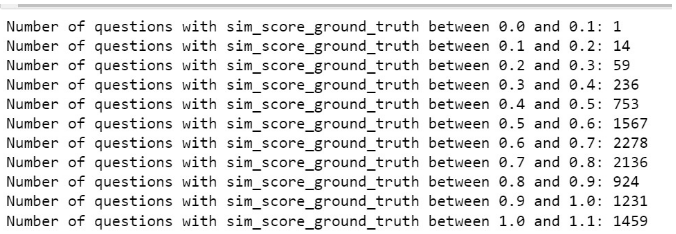


```
Answers:
1. theodosius established a single christian doctrine ( specified as that professed by pope dam
sus i of rome and pope peter ii of alexandria ) as the empire ' s official religion
2. nicene christianity became the state church of the roman empire with the edict of thessaloni
a in 380 ad
3. when did christianity become official religion of rome? diocletianic persecution of 303 - 31
and the donatist controversy that arose in consequence, constantine had convened councils of ch
istian bishops to define the orthodoxy, or " correct teaching ", of the christian faith, expand
ng on earlier christian councils.
```

Fig. 5. Question, Context, and Extracted Answer using Reader Model - Top 5 Answers

C. Prediction Accuracy

We analyze the similarity between the recommended passage and the ground truth passage using a frequency distribution of similarity scores. Figure 7 presents the distribution of similarity scores.



```
Number of questions with sim_score_ground_truth between 0.0 and 0.1: 1
Number of questions with sim_score_ground_truth between 0.1 and 0.2: 14
Number of questions with sim_score_ground_truth between 0.2 and 0.3: 59
Number of questions with sim_score_ground_truth between 0.3 and 0.4: 236
Number of questions with sim_score_ground_truth between 0.4 and 0.5: 753
Number of questions with sim_score_ground_truth between 0.5 and 0.6: 1567
Number of questions with sim_score_ground_truth between 0.6 and 0.7: 2278
Number of questions with sim_score_ground_truth between 0.7 and 0.8: 2136
Number of questions with sim_score_ground_truth between 0.8 and 0.9: 924
Number of questions with sim_score_ground_truth between 0.9 and 1.0: 1231
Number of questions with sim_score_ground_truth between 1.0 and 1.1: 1459
```

Fig. 6. Similarity Distribution

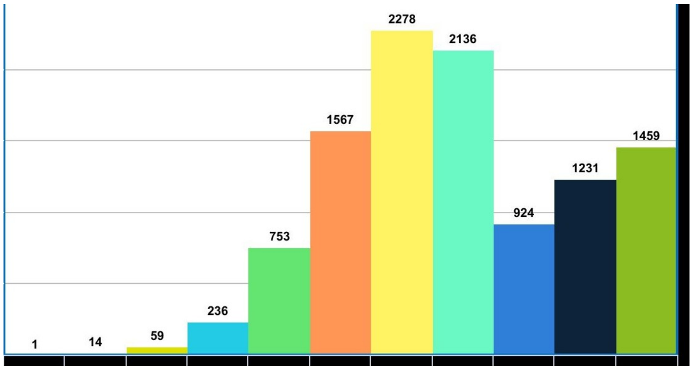


Fig. 7. Plot

V. Conclusion

In conclusion, the results of our experimentation appear promising and show performance trends that are comparable to those observed by the original author. The performance of both our retriever and reader models demonstrates potential for further improvement with additional training and fine-tuning.

Specifically, we observed that the retriever model successfully recommended contexts that closely resemble the ground truth passages, as depicted in Figure 2. This suggests that the retriever's ability to retrieve relevant information is effective, and further optimization could enhance its accuracy.

Moreover, the reader model showcased its capability to extract accurate answers from the provided contexts, as illustrated in Figure 4. The extracted answers align well with the corresponding questions, highlighting the reader's understanding and comprehension of the input data.

Looking ahead, more extensive training and optimization of our models could yield even better results, potentially achieving performance levels closer to state-of-the-art benchmarks. Additionally, increasing the size and diversity of the training dataset would likely enhance the generalization and robustness of our models.

Furthermore, our experimentation confirms that even with a relatively small dataset, training a BERT-based encoder for question answering can yield promising results. This underscores the effectiveness of transfer learning and pre-trained language models in leveraging existing knowledge to perform specific tasks with limited data.

In summary, our findings suggest that with continued development and refinement, our models have the potential to achieve higher accuracy and effectiveness in question answering tasks. The successful application of BERT-based models on a modest dataset emphasizes the accessibility and practicality of leveraging advanced NLP techniques for real-world applications.

VI. Future Scope

Looking ahead, there are exciting opportunities to enhance our question-answering system and improve its performance:

- **Integration of Generative Models**: Consider incorporating generative models like GPT (Generative Pre-trained Transformer) alongside the reader component. This integration could enable the system to generate more nuanced and contextually rich answers, enhancing response quality.
- **Exploring RAG Models for Enhanced Retrieval**: Explore advanced retrieval models such as RAG (Retrieval-Augmented Generation) to augment the retriever's capabilities. RAG models leverage large-scale language models for both retrieval and generation tasks, improving the relevance and diversity of retrieved passages.
- **Combining Techniques for Comprehensive QA**: Investigate a hybrid approach that combines generative and retrieval-based methods. By leveraging the strengths of different architectures, we can address complex query scenarios and provide more comprehensive answers.

These future directions require rigorous experimentation and validation to assess their impact on system performance. By continuously refining and iterating on our approach, we aim to develop a robust and effective question-answering system that excels in various domains and scenarios.