2024 IEEE 12th Conference on Systems, Process & Control (ICSPC), 7 December 2024, Malacca, Malaysia

Question Answering with Language Models

Jia Yin Wong
Faculty of Info. Science & Technology
Multimedia University
Melaka, Malaysia
1201101031@student.mmu.edu.my

Chin Poo Lee*
School of AI and Advanced Computing
Xi'an Jiaotong-Liverpool University
Jiangsu Province, China
ChinPoo.Lee@xjtlu.edu.cn

Kian Ming Lim*
School of Computer Science
University of Nottingham Ningbo China
Zhejiang Province, China
Kian-Ming.Lim@nottingham.edu.cn

Jit Yan Lim
Faculty of Info. Science & Technology
Multimedia University
Melaka, Malaysia
lim.jityan@mmu.edu.my

Jashila Nair Mogan*
Faculty of Info. Science & Technology
Multimedia University
Melaka, Malaysia
jashilanair@mmu.edu.my
*Corresponding author

2024 IEEE 12th Conference on Systems, Process & Control (ICSPC) | 979-8-3503-9139-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICSPC63060.2024.10862271

***Abstract*—With the significant increase in the number of resources providing various types of information, ranging from thousands to millions of sources including research papers, blogs, and more, extracting and providing meaningful insights has become a challenge. This study develops a Question Answering (QA) system using the Haystack framework, capable of retrieving relevant documents and extracting answers from lengthy texts. The implemented QA system consists of four main components: an indexing pipeline, a document store, a searching pipeline, and an evaluation module. Additionally, this study investigates how different model architectures affect performance in retrieving and extracting answers. Two different retrievers, BM25 Retriever and Dense Passage Retriever, and five different readers, BERT, RoBERTa, ALBERT, MiniLM, and ELECTRA, are examined. The models are tested and evaluated using the SQuAD datasets. The combination of BM25 Retriever and RoBERTa Reader achieved the best performance, with an F1-score of 0.9301 and an Exact Match score of 0.8956 on the SQuAD2.0 dataset.**

***Index Terms*—question answering, natural language processing, transformer, BM25 retriever, dense passage retriever, BERT, RoBERTa, ALBERT, MiniLM, ELECTRA**

I. Introduction

In the 21st century, Artificial Intelligence (AI) has emerged as a focal point, driven by advancements in computational technology. AI has found widespread application across diverse domains, demonstrating its versatility and impact, transcending traditional boundaries. Deep Learning (DL), a subset of AI, involves neural network architectures that allow models to learn and process data similarly to the human brain.

In this study, we focus on DL in Natural Language Processing (NLP) for building a Question Answering (QA) system. A QA system is an application or algorithm that retrieves and returns or generates answers based on users' questions. The type of answer given by the model depends on the implemented QA algorithm. An extractive QA system retrieves documents and extracts related spans or sentences from the corpus as answers to queries. A generative QA system can return novel answers that it has restructured or answers that are not explicitly stated in the corpus.

With the significant increase in the number of resources providing various types of information, ranging from thousands to millions, encompassing research papers, blogs, and a multitude of other sources, extracting and providing meaningful insights or desired outputs from this vast knowledge landscape has become a challenge. Thus, the development of QA systems is applied to websites, search engines, chatbots, and other platforms to provide direct and relevant answers instead of lengthy documents.

This study primarily aims to:

- Develop a QA system with the ability to retrieve relevant documents and extract answers to given questions.
- Investigate how different model architectures affect the performance of retrieving and extracting answers.

The rest of the paper is organized as follows: Section II presents a literature review of QA systems. Section III explains the implemented methodology. Section IV analyzes the results of the experiments. Lastly, Section V concludes this study and discusses future work and challenges.

II. Literature Review

There is a vast range of existing works, from traditional methods to deep learning methods, and advanced transformer-based architectures. QA systems can be categorized into two types: open-domain and closed-domain. Open-domain QA handles all kinds of questions across diverse topics, providing general and broad knowledge as output answers. In contrast, closed-domain QA focuses on specific domains such as medical, engineering, industrial, or other areas, and returns more professional and accurate answers. The following sections present a literature review of different domains and architectures of QA systems.

A. Open-domain Question Answering Systems

A QA system designed to answer factoid questions was presented in [1]. Term Frequency–Inverse Document Frequency

The research in this work was supported by the Telekom Malaysia Research & Development under grant number RDTC/221064, RDTC/231075, and RDTC/231084.

(TF-IDF) is used to retrieve relevant words or passages from SQuAD in the passage retrieval phase. During the searching phase, keywords are analyzed to extract the most relevant answer. The proposed system consists of an Information Retriever (IR) system, fuzzy team, server logic, answer extraction, query, and database. The top three relevant documents are returned, and a sentence retriever tokenizes the sentences and computes n-grams similarity between questions and sentences. The answer is generated in the answer processing part based on the expected answer type.

A humanoid robot with self-learning capabilities is built using a deep learning-based QA system [2]. The system implements a Recurrent Neural Network (RNN) as the encoder, used with Bidirectional Attention Flow (BiDAF) to understand and find answers. The robot learns from three sources: big data from crawled websites, new knowledge told by users, and a constructed knowledge base. Google API Speech Recognition recognizes voice, while Google Text to Speech speaks out the answer.

A QA system utilizing a stacked BiLSTM neural network and coattention mechanism was proposed in [3]. The model's encoder is a stacked BiLSTM neural network designed to extract hidden features from input sentences. Coattention and attention mechanisms encode the sentences, capturing the relationship between questions and answers. The model uses a combination of cosine similarity and Euclidean distance to measure similarity between the vector representations of questions and answers.

Vaishali Fulmal et al. [4] proposed a QA system that finds similarities between an input question and relevant questions in a dataset. Questions are clustered into multiple groups using hierarchical clustering. BiLSTM retains the context of the questions. The output is passed to a fully connected dense layer to classify similar types of questions. The system matches the input question with available questions and provides an answer.

An online QA framework to handle mixed languages, including Hindi, Telugu, and Tamil in English sentences, was presented in [5]. The framework translates mixed language questions into English using Google Translate API. The answer type component predicts the answer type, while the web search component retrieves five relevant documents. Named Entity Recognition (NER) identifies the entity type of each word. The result is displayed using the answer-type and entity type of each word. The implemented classifier model is RNN.

Min Yuh Day et al. [6] proposed an integrated QA model to answer factoid questions. The QA system combines fine-tuned BERT, question expected answer type (Q-EAT) classification model, and answer type (AT) classification model. The BERT QA model receives the input question and gives the position of text paragraphs as output. The Q-EAT model returns the question expected answer type based on the input question. The AT model receives answer sentences as input and returns the answer type. The research finds that identifying the types of questions and answers improves QA system performance.

An integrated QA model called BERTserini, proposed by Wei Yang et al., combines an IR with a BERT base reader [7]. The IR used is the open-source Anserini Information Retrieval, and the authors fine-tuned the BERT base model using the SQuAD1.1 dataset, implementing it as the reader. BERTserini is deployed as a chatbot that can be used on laptops or mobile phones to interact with users. The Anserini retriever is a single-stage retriever that retrieves text from Wikipedia and passes the relevant document to the BERT reader. To compare and aggregate results from different segments, the researchers removed the last layer of the different answer spans, which is a softmax layer.

B. Closed-domain Question Answering Systems

A Chinese QA framework combining Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) was proposed by Lin-Qin Cai et al. [8]. The CNN-BiLSTM combination allows the model to find and represent important textual and contextual information. Coattention and attention mechanisms obtain the semantic interaction and features representation of the QA pairs. Cai et al. also proposed a method to match the QA pairs, allowing the Chinese intelligent QA system to be implemented in a restricted domain.

A Collaborative Decision Convolutional Neural Network (CDCNN) framework in the domain of Health expert QA (HQA) services was presented by Ze Hu et al. [9]. The CDCNN learns semantic non-linear features embedded in the data. The CNN and dependency-sensitive CNN (DSCNN) first use word embeddings on an HQA dataset to learn semantic knowledge for short answer text input. The collaborative decision component combines semantic knowledge into a more precise representation. The multimodal learning component combines semantic representation and temporal features to produce a final joint representation.

Zhaohua Hu [10] constructed a QA system specializing in railway technology. The system accepts text or voice as input, converts it into text format, and processes Chinese word segmentation in the natural language processing layer. The matching algorithms layer calculates semantic matching between input and knowledge points, outputting the best matching knowledge points. The interactive processing layer binds the title and detailed content of the best-matched knowledge point to generate the answer. Semantic similarity is calculated using Deep Structured Semantics Models (DSSM) and cosine function. The CNN projects the isolated word vector to a semantic space in the presentation layer.

COBERT [11] is a closed-domain QA system in the field of COVID-19 infections, utilizing a retriever and reader. The system answers complex queries by returning a span from the corpus, the article's title, and the source of answer extraction. The retriever used is TF-IDF Retriever, retrieving the top 500 documents to pass into the reader. The reader used in this research is DistilBERT, fine-tuned to the specific domain. The ranker in COBERT uses cosine similarity to determine answers for input questions.

Andrew Wen et al. [12] studied and evaluated five different combinations of QA models in the clinical field to answer why-questions. The data used are emrQAwhy, i2b2notespre, and SQuADwhy. They fine-tuned the BERT base model with different combinations of the three datasets to assess the benefit of each data source. The five models are: (1) BERT + emrQAwhy, (2) BERT + SQuADwhy + emrQAwhy, (3) BERT + i2b2notespre + emrQAwhy, (4) BERT + ClinBERTpre + emrQAwhy, and (5) BERT + ClinBERTpre + SQuADwhy + emrQAwhy. The best result is obtained by the fifth model, showing that fine-tuning models with more specific datasets allows them to answer specialized questions accurately.

MedFuseNet [13] is a QA system in the medical domain, a multimodal deep learning model combined with an attention mechanism. The model receives image and text questions as input simultaneously, fusing these different formats using the attention mechanism. Image features are extracted using ResNet-512, and question features are extracted by the BERT model. The multimodal factorized bilinear (MFB) algorithm fuses both feature representations. Power normalization and L-2 normalization are applied to the MFB output. The model uses two types of attention mechanisms: Image Attention and Image-Question Co-Attention. An LSTM-based decoder model generates the answer.

Qinjun Qiu et al. [14] constructed a QA system in the geological domain with an automatically built Q&A dataset based on mineral exploration ontology. The QA model consists of two modules: an information retriever (IR) and a pretrained BERT model. The BERT model includes special flags such as [CLS] at the top of the first sentence, [SEP] between sentences, and [MASK] to hide certain words for prediction. The model learns long-term dependencies using a self-attention mechanism.

A RoBERTa-law QA model [15] was proposed for the legal domain, focusing on searching for relevant legal documents and extracting answers. The corpus data is pre-processed by splitting long articles into multiple sub-articles, using word segmentation with pyvi, converting text to lowercase, and removing stopwords and punctuation. The BM25 is used as a lexical matching model to find the most relevant articles. A pre-trained RoBERTa model, fine-tuned with a 4GB law corpus of legal text data, extracts the answer.

III. Methods

This study presents a QA system mainly comprises four parts: indexing pipeline, document store, searching pipeline, and evaluation, as shown in Fig. 1. Firstly, the input data or question-answer pairs pass through an indexing pipeline before being stored in a Document Store. Then, the data stored in the document store are passed to the searching pipeline, which includes a retriever and a reader. The searching pipeline extracts the answer for a question from the corpus stored in the document store. Lastly, the answers provided by the model are evaluated to measure system performance.

A. Indexing Pipeline

The indexing pipeline ensures that the input data are well-organized for downstream retrieval and question answering tasks. This pipeline comprises two components: file converter and preprocessor. The file converter converts the input dataset from a JSON format file into Haystack's internal document and label objects. After conversion, the document objects are cleaned and processed by the preprocessor before indexing and storage. The preprocessing includes segmenting documents into smaller passages based on word boundaries, with a split length of 200 and no overlap. The data are then indexed into documents and labels with their own indices and passed into the Document Store, facilitating traceability of desired answers in subsequent tasks.

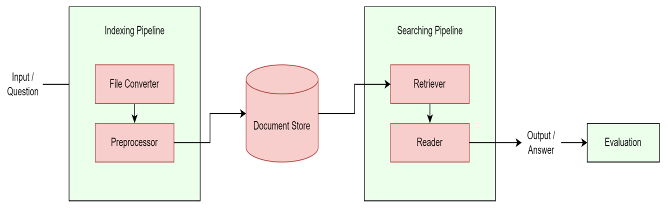

Fig. 1. Process flow of the question answering system.

B. Document Store

A document store is a database for the question answering system, keeping document objects that include text and metadata. The document store used in this study is the Elasticsearch Document Store, which employs Elasticsearch as the backend for storing and indexing documents, enabling fast searching. Elasticsearch is an open-source search engine that models the behavior of the data in real time using machine learning methods [16]. This document store is suitable for efficiently storing and retrieving large amounts of text data.

C. Searching Pipeline

The searching pipeline, also known as a query pipeline, is applied to user queries or questions. The components in this pipeline find relevant information within the document collection in the Document Store. The searching pipeline implemented in this study consists of two key components: Retriever and Reader.

1) Retriever: A retriever searches the Document Store and finds candidate documents that match the query. It acts as a lightweight filter, selecting the most relevant documents for further processing by the reader.

Two types of retrievers used in this study are:

- **BM25 Retriever:** Also known as Okapi BM25, this sparse retriever ranks documents based on their relevance to a search query. It works without the need for a neural network and is a variant of TF-IDF, mainly used for document search and information retrieval [17].
- **Dense Passage Retriever:** This retriever uses dense vector representation and is designed for open-domain question answering to retrieve related passages from a large corpus of unstructured text. It employs a dual-encoder architecture to map text passages and queries

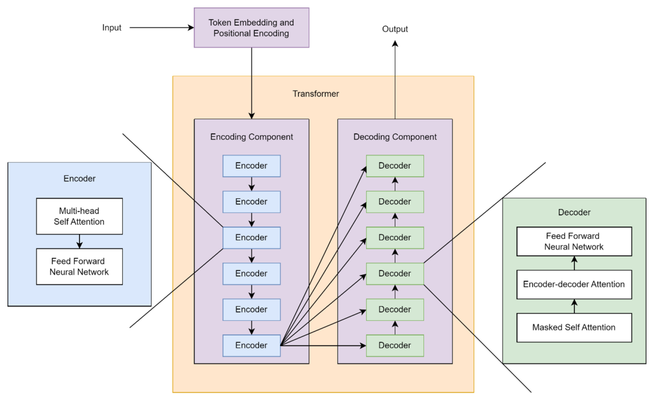
Input
Token Embedding and Positional Encoding
Output
Transformer
Encoding Component
Decoding Component
Encoder
Multi-head Self Attention
Feed Forward Neural Network
Encoder
Decoder
Decoder
Feed Forward Neural Network
Encoder-decoder Attention
Masked Self Attention

Fig. 2. The architecture of Transformer.

into d-dimensional vectors, retrieving the closest vectors [18].

In this study, the "top_k" parameter of the retriever is set to 5, meaning the retriever returns the 5 most relevant documents for each query, which are then passed to the reader for final answer extraction.

2) Reader: A reader in the question answering system extracts answers for a query from the retrieved documents. It understands the context of the documents to identify potential answer spans. For experimentation and comparison, five pre-trained models are used: BERT, RoBERTa, ALBERT, MiniLM, and ELECTRA. These models are based on the Transformer architecture and are fine-tuned with the SQuAD2.0 dataset.

The Transformer model, introduced in "Attention is All You Need" [19], focuses on attention mechanisms and excludes recurrence and convolutions. It comprises an encoder-decoder architecture, with six identical encoders and decoders stacked on top of each other, as shown in Fig. 2. Each encoder has a self-attention layer followed by a feed-forward neural network layer, allowing the model to emphasize certain parts of the input sequence. The decoder stack includes a masked self-attention layer, an encoder-decoder attention layer, and a feed-forward neural network layer. The final layers of the decoder stack are a linear layer and a softmax layer, transforming scores into probabilities.

- **BERT**: Bidirectional Encoder Representations from Transformers (BERT) is a language model developed by Jacob Devlin et al. [20]. It comprises only the encoder architecture and has two variations: BERT Base and BERT Large. The BERT Base model, used in this study, consists of 12 encoder layers and is trained on masked language modeling (MLM) and next sentence prediction (NSP). The model outputs a vector for each position, which can be passed to a classifier for downstream tasks. The BERT Base model used is loaded from [21].
- **RoBERTa**: RoBERTa, an improved version of BERT proposed by Liu et al. [22], uses larger mini-batches and learning rates, removes NSP, and focuses on MLM with dynamic masking. The RoBERTa model used in this system is loaded from [23].
- **ALBERT**: A Lite BERT (ALBERT), proposed by Lan et al. [24], reduces BERT's parameters while improving performance using factorized embedding parameterization and cross-layer parameter sharing. It replaces NSP with sentence-order prediction (SOP). The ALBERT model used is loaded from [25].
- **MiniLM**: MiniLM, a compressed version of large Transformers, uses deep self-attention distillation [26]. This model trains a smaller student model to mimic a larger teacher model's self-attention module. The MiniLM model used is loaded from [27].
- **ELECTRA**: Efficiently Learning an Encoder that Classifies Token Replacements Accurately (ELECTRA) [28] employs a pre-training task called replaced token detection. It consists of a generator and a discriminator, where the discriminator classifies whether tokens are original or replaced. The ELECTRA model used is loaded from [29].

IV. Experiment

To evaluate and compare the performance of the QA system, several experiments were conducted, involving different datasets for testing, two different retriever models, and five transformer-based models for the reader.

A. Evaluation Dataset

The Stanford Question Answering Dataset (SQuAD) was used to evaluate the models' performance in this study. Released in 2016 by researchers at Stanford University [30], SQuAD is a reading comprehension dataset primarily used for training and evaluating question-answering systems. It comprises question-answer pairs derived from Wikipedia articles, with answers typically being text spans from the articles, including some unanswerable questions. SQuAD has two main versions: SQuAD1.1 and SQuAD2.0.

SQuAD1.1 contains over 100,000 question-answer pairs from 536 Wikipedia articles, split into a training set with approximately 87,599 pairs and a development set with 10,570 pairs. SQuAD2.0 builds on SQuAD1.1 by adding over 50,000 unanswerable questions, resulting in 130,319 pairs in the training set and 11,873 pairs in the development set. The dataset is structured in JSON format, consisting of paragraphs (text passages), questions, answers, and metadata.

B. Evaluation Metrics

The performance of the extracted answers was evaluated using several metrics. For the retriever, recall, mean reciprocal rank (MRR), and mean average precision (MAP) were used. For the reader, exact match (EM) and F1 score were employed.

Recall measures how often the correct document is included in the retrieved documents. When multiple correct documents are involved, recall_single_hit measures if at least one correct document is retrieved, while recall_multi_hit measures how many correct documents are retrieved.

$$\text{Recall} = \frac{\text{Number of retrieved correct documents}}{\text{Total number of correct documents in the corpus}} \tag{1}$$

Mean Reciprocal Rank (MRR) evaluates the retriever's effectiveness in ranking relevant documents, calculated by the rank of the first correctly retrieved document.

$$\text{RR} = \frac{1}{\text{Rank of the first correct document}}$$

$$\text{MRR} = \frac{\sum \text{RR values for all questions}}{\text{Number of questions}} \tag{2}$$

Mean Average Precision (MAP) is the mean of average precision scores for each query, considering the rank of all correctly retrieved documents.

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AveP}(q)}{Q} \tag{3}$$

Exact Match (EM) measures the percentage of exactly matched answers provided by the reader.

$$\text{EM} = \frac{\text{Number of questions with exact match answers}}{\text{Total number of questions}} \tag{4}$$

F1-Score is the harmonic mean of precision and recall, considering false positives and false negatives.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{5}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

where TP is true positive, FP is false positive, and FN is false negative.

C. Results and Analysis

Table I shows that the performance of retrievers in the proposed QA system. Based on the performance metrics, the BM25 Retriever outperforms the Dense Passage Retriever in all evaluation methods. BM25 achieves higher recall for both single and multiple relevant documents (0.9165 and 0.9129, respectively), indicating its consistent ability to retrieve relevant information. It also has a superior Mean Reciprocal Rank of 0.8346, demonstrating more frequent top-ranking of correct documents compared to the Dense Passage Retriever's 0.5390.

Additionally, BM25 excels in precision (0.2326) and Mean Average Precision (0.8162), highlighting its effectiveness in retrieving and ranking relevant documents. This superior performance is due to BM25's probabilistic approach, which models term frequency and document length effectively, justifying its status as the best-performing retriever.

Table II presents the F1-score, EM, and computational time for the BERT, RoBERTa, ALBERT, MiniLM, and ELECTRA models. On the SQuAD1.1 dataset, the best-performing language model is MiniLM, achieving the highest F1-Score of 0.8843 and the highest Exact Match (EM) score of 0.8471, with a processing time of 48.96 minutes. This indicates that MiniLM excels in accurately identifying and extracting relevant answers from the given text, demonstrating its efficiency and precision on this dataset.

For the SQuAD2.0 dataset, RoBERTa stands out as the best-performing model with an impressive F1-Score of 0.9301 and an EM score of 0.8956. These results suggest that RoBERTa is particularly effective in handling the challenges of SQuAD2.0, which includes questions that may not have answers in the provided context, showcasing its superior ability to discern when information is absent and still maintain high accuracy.

TABLE I
RETRIEVERS' PERFORMANCE

Evaluation Method	BM25 Retriever	Dense Passage Retriever
Recall (single relevant document)	**0.9165**	0.7196
Recall (multiple relevant documents)	**0.9129**	0.7086
Mean Reciprocal Rank (MRR)	**0.8346**	0.5390
Precision	**0.2326**	0.1930
Mean Average Precision (MAP)	**0.8162**	0.5219

TABLE II
READERS' PERFORMANCE ON SQUAD1.1 AND SQUAD2.0

Model	Dataset	F1-Score	EM	Time (mins)
BERT	SQuAD1.1	0.8632	0.8154	54.27
	SQuAD2.0	0.4410	0.4147	64.43
RoBERTa	SQuAD1.1	0.8444	0.7797	44.89
	SQuAD2.0	**0.9301**	**0.8956**	49.72
ALBERT	SQuAD1.1	0.8658	0.8190	54.32
	SQuAD2.0	0.4409	0.4177	66.19
MiniLM	SQuAD1.1	**0.8843**	**0.8471**	48.96
	SQuAD2.0	0.4505	0.4304	59.27
ELECTRA	SQuAD1.1	0.8668	0.8197	55.64
	SQuAD2.0	0.4517	0.4266	64.94

V. CONCLUSION

This paper presents a QA system with various retriever and reader models. Our evaluation of retrieval systems indicates that the BM25 Retriever consistently outperforms the Dense Passage Retriever across all evaluation metrics. The BM25 Retriever's probabilistic approach, which effectively models term frequency and document length, allows it to retrieve and rank relevant documents with high accuracy.

Furthermore, our analysis reveals that MiniLM and RoBERTa are the best-performing models on the SQuAD1.1 and SQuAD2.0 datasets, respectively. MiniLM achieves the highest F1-Score and Exact Match on SQuAD1.1, showcasing its efficiency and precision in identifying and extracting relevant answers. On the other hand, RoBERTa excels on the more challenging SQuAD2.0 dataset, demonstrating superior ability to handle questions that may not have answers in the provided context.

These findings underscore the importance of selecting appropriate models and retrieval methods for different types of question answering tasks. Future work will focus on further refining these models and exploring hybrid approaches to enhance overall system performance.

References

[1] M. V. Sadhuram and A. Soni, "Natural language processing based new approach to design factoid question answering system," in *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2020, pp. 276-281.

[2] W. Budiharto, V. Andreas, and A. A. S. Gunawan, "Deep learning-based question answering system for intelligent humanoid robot," *Journal of Big Data*, vol. 7, pp. 1-10, 2020.

[3] L. Cai, S. Zhou, X. Yan, and R. Yuan, "A stacked BiLSTM neural network based on coattention mechanism for question answering," *Computational Intelligence and Neuroscience*, vol. 2019, 2019.

[4] V. Fulmal, K. P. Moholkar, and S. H. Patil, "The implementation of question answer system using deep learning," *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 1S, pp. 176-182, 2021.

[5] S. Thara, E. Sampath, and P. Reddy, "Code mixed question answering challenge using deep learning methods," in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, 2020, pp. 1331-1337.

[6] M. Y. Day and Y. L. Kuo, "A study of deep learning for factoid question answering system," in *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, 2020, pp. 419-424.

[7] W. Yang et al., "End-to-end open-domain question answering with bertserini," *arXiv preprint arXiv:1902.01718*, 2019.

[8] L. Q. Cai, M. Wei, S. T. Zhou, and X. Yan, "Intelligent question answering in restricted domains using deep learning and question pair matching," *IEEE Access*, vol. 8, pp. 32922-32934, 2020.

[9] Z. Hu et al., "Predicting the quality of online health expert question-answering services with temporal features in a deep learning framework," *Neurocomputing*, vol. 275, pp. 2769-2782, 2018.

[10] Z. Hu, "Research and implementation of railway technical specification question answering system based on deep learning," in *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 2020, pp. 5-9.

[11] J. A. Alzubi, R. Jain, A. Singh, P. Parwekar, and M. Gupta, "COBERT: COVID-19 question answering system using BERT," *Arabian Journal for Science and Engineering*, vol. 48, no. 8, pp. 11003-11013, 2023.

[12] A. Wen, M. Y. Elwazir, S. Moon, and J. Fan, "Adapting and evaluating a deep learning language model for clinical why-question answering," *JAMIA Open*, vol. 3, no. 1, pp. 16-20, 2020.

[13] D. Sharma, S. Purushotham, and C. K. Reddy, "MedFuseNet: An attention-based multimodal deep learning model for visual question answering in the medical domain," *Scientific Reports*, vol. 11, no. 1, p. 19826, 2021.

[14] Q. Qiu et al., "A question answering system based on mineral exploration ontology generation: A deep learning methodology," *Ore Geology Reviews*, p. 105294, 2023.

[15] H. N. Van, D. Nguyen, P. M. Nguyen, and M. Le Nguyen, "Miko team: Deep learning approach for legal question answering in alqac 2022," in *2022 14th International Conference on Knowledge and Systems Engineering (KSE)*, 2022, pp. 1-5.

[16] Elastic, "Elasticsearch introduction," *Elasticsearch Reference [7.6]*, www.elastic.co. [Online]. Available: https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html.

[17] Langchain, "ElasticSearch BM25," Python.langchain.com. [Online]. Available: https://python.langchain.com/docs/integrations/retrievers/elastic_search_bm25. [Accessed: Jan. 26, 2024].

[18] V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," *arXiv preprint arXiv:2004.04906*, 2020.

[19] A. Vaswani et al., "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[20] 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.

[21] deepset, "bert-base-cased-squad2," Huggingface.co, Jan. 4, 2024. [Online]. Available: https://huggingface.co/deepset/bert-base-cased-squad2.

[22] Y. Liu et al., "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[23] deepset, "roberta-base-squad2," Huggingface.co. [Online]. Available: https://huggingface.co/deepset/roberta-base-squad2.

[24] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[25] twmkn9, "albert-base-v2-squad2," Huggingface.co. [Online]. Available: https://huggingface.co/twmkn9/albert-base-v2-squad2. [Accessed: Jan. 26, 2024].

[26] W. Wang et al., "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 5776-5788.

[27] deepset, "minilm-uncased-squad2," Huggingface.co, Jan. 4, 2024. [Online]. Available: https://huggingface.co/deepset/minilm-uncased-squad2.

[28] W. Yang et al., "Electra: Pre-training text encoders as discriminators rather than generators," *arXiv preprint arXiv:2003.10555*, 2020.

[29] deepset, "electra-base-squad2," Huggingface.co, Apr. 5, 2023. [Online]. Available: https://huggingface.co/deepset/electra-base-squad2.

[30] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016.