SURVEY: OPEN DOMAIN QUESTION ANSWERING

Rajan Sah
Data Science
Universität Trier
Trier, Rhineland-Palatinate 54295
Email: S4rasahh@uni-trier.de

Monotosh Kumar Das Chandan
Data Science
Universität Trier
Trier, Rhineland-Palatinate 54295
Email: s4modasc@uni-trier.de

ABSTRACT

Open Domain Question Answering is an important task of answering questions from any doamin.It has become one of the popular work to research in NLU. OpenQA should formulate answer based on question being asked. If the question is when was Barak Obama born? The answer would be 1961.There has been done lately lot of research work as it could optimize the search engine, recommendation system, chatbots. There has been large push for QA model for web scale services. We also discuss how multi datasets is modelled with collection of single-datasets experts by training lightweights adapter. OpenQA is not only limited to text, it is being implemented for voice and video QA too.Even though there has been lot of research work has been done for this field, there is lack of survey. In this work, we review different research work and method which has been implemented. We also write about origin of OpenQA. For answering complex and diverse open domain question answering model, We also discussed about QA which require multi hop reasoning at web scale, multi step retriever-reader model. Also some model related to iterative retrieval on knowledge based text and iterative query generator. we give brief introduction on different approaches which has been implemented and how different methods work along with another paper which integrates BERT with open source Anserini Information Retrieval. We discuss basic architecture of OpenQA. we hope our research survey work is going to help and inform researcher about their recent advancement.

1 INTRODUCTION

The goal of question answering (QA) is to respond to the user's questions in natural language with exact responses. It has been an ongoing project since the 1960s. The QA system strives to provide the ultimate response to a question directly rather than returning a collection of pertinent snippets or hyperlinks, making it more user-friendly and effective than a search engine By integrating open domain QA approaches into their search functions, several modern web search engines, including Google and Bing, have been progressing towards better intelligence. With the aid of these methods, search engines can now provide specific answers for specific questions.But it's challenging to extract answer from large pool of documents for complex and diverse questions, means the model needs to iteratively retrieve the similar documents in order to approach towards the correct piece of answer for given query. In recent years there have been lots of work on this topic, we have also talked about those papers which focus on this problem

FIGURE 1. An illustration of OpenQA. Given a natural language question, the system infers the answer.

In this paper [1], OpenQA has been formulated as phrase retrieval problem .This paper tried to solve the limitation of sparse representation of QA.In this work, they found effective method to learn representation of phrase densely coupled with negative sampling and in batch along with fine tuning strategy.Wikipedia is being used as data source which has billions of phrases. The answer is retrieved on passage level.The goal is to give an-

swer for the question based on documents.Wikipedia is divided into documents. DensePhrase, a retrieval model based on dense representation .All the phrase is pre-indexed and stored offline. asai2019learning

[2] describes a graph based recurrent retrieval approach learns to retrieve reasoning paths over wikipedia. The retrieval model trains RNN to learn and retrieve sequential paragraph.The reader model ranks path and extract answer from best path.This paper solve problem of multi hop question.

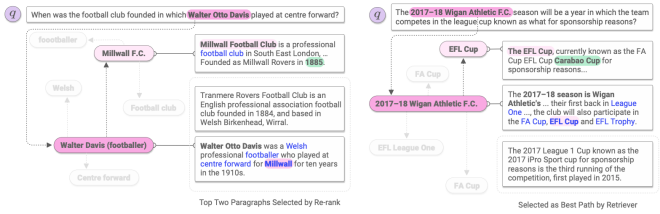

FIGURE 2. An example of open-domain multi-hop question from HotpotQA

In this paper [3], Transformer based QA models use self-attention for both question and passage, causing it to be slow and memory intensive, which is being solved by using Deformer, a decomposed transformer, substituting question wise and passage wise self attention.This changes lead to reduction in runtime compute by large scale.The Deformer is initialized with pretraining weights and fine tuned on large QA datasets.

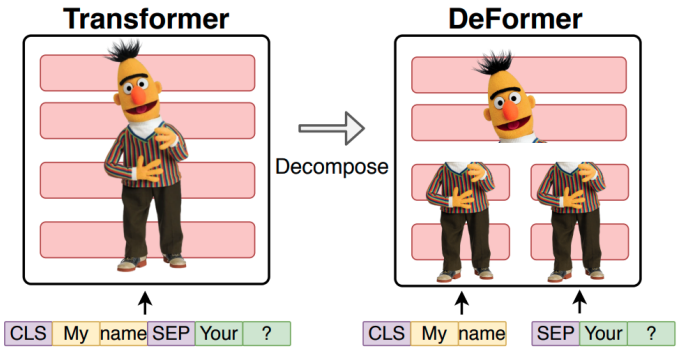

FIGURE 3. Original Transformer applies full self attention to encode the concatenated question and passage sequence, while DeFormer encodes the question and passage independently

In this paper [4], the author has tried to overcome the shortcoming of multi-datasets model which performs well on training distributions and also serve as strong starting point for transferring learning to new datasets by combining advantages of single and multi-datasets of single and multi-datasets by training collection of single datasets experts which share underlying transformer models.This system is based on adapter which is task specific modules interleaved between layers of pre-trained Transformer.A shared transformer is trained in a multi-adapter set up before refining adapters for each datasets which is called Multi-Adapter DataSet Experts(MADE).This ultimately result in more accurate and robust specialization models.

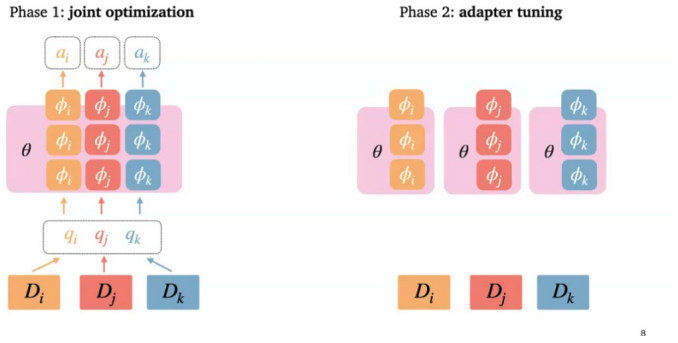

FIGURE 4. : MADE consists of a set of dataset-specific adapters and classifiers with a shared Transformer model . They are optimized jointly on a set of training datasets (left). To transfer to a new dataset (right), either average the parameters of the adapters, or fine-tune all adapters on the target dataset and take the weighted average at the end

In this paper [5], authors are trying to demonstrate how they have integrated BERT with the open source Anserini IR toolkit to create BERT-serini, which is end to end open domain question answering system.The system directly works on a large corpus of wikipedia articles.They also tried to fine-tune pretrained BERT with SQuAD, which is enough to achieve high accuracy.They also deployed BERTSerini as a chatbot that user can interact with on diverse platforms.Its a series of neural model. The approach [6] for responding to open-domain queries described in this work involves iterative dialogue between the reader and the retriever. The PullNet [7] framework builds a question-specific subgraph and extracts the answer from subgraph via an iterative method. This paper focused on [8] an iterative process to build question specific subgraph that containing information relevant to the question. This paper combines [9] data from several documents that call for sequential (multi-hop) processing or logical reasoning in order to deduce the solution. A new technique introduced by this paper [10] to extract a discrete reasoning chain over the text, which is made up of a string of sentences that lead to the solution.

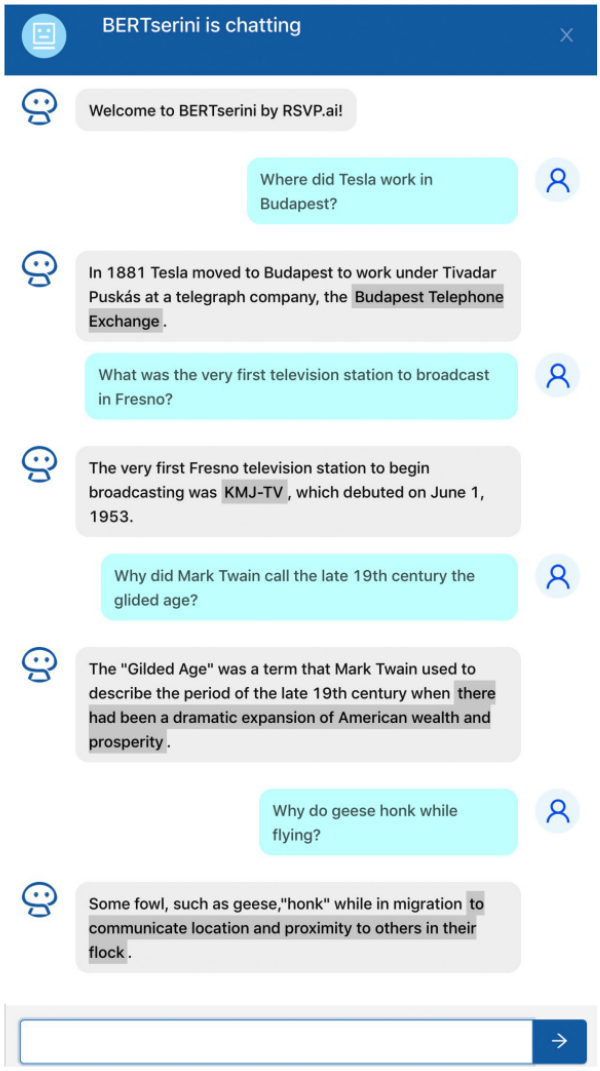

FIGURE 5. A screenshot of BERTserini

2 Evolution of QA

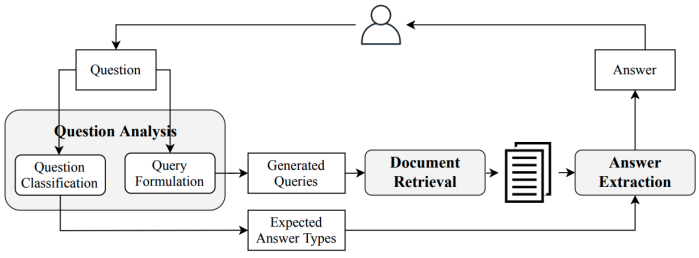

FIGURE 6. Traditional QA Architecture

Origins of question answering date back to the 1960s, which can be traced to the Text Retrieval Conferences. In the beginning, it was thought of only as retrieving documents and extracting answer spans.The oldest question answering system were Baseball and Lunar.Baseball mainly focused on major baseball league question and lunar was for analysis of rock brought by apollo moon mission.They used the concept of database which was handwritten by experts.In 1970, knowledge bases were developed which targeted domain of knowledge.it further got improved and started working on unix operating system.Then QA evolved to database centric system which further moved towards natural language system.

In modern times, most of the big corporate has their own question answering system like facebook has new generative AI MLQAT. IBM has its system called techQA. Microsoft has come with Azure Cognitive Services.Google came up with BERT and working on developing BARD to compete with chatGPT. OpenAI is leader i would say by coming with something called ChatGPT, a chatbot which respond to text based queries and give natural language generated answer.

3 Basic Architecture

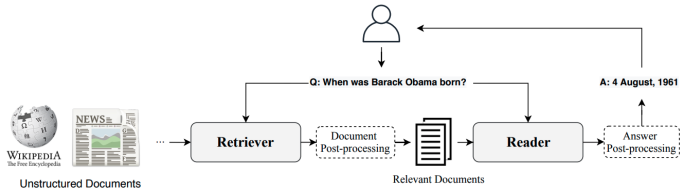

FIGURE 7. Basic Retriever-Reader Architecture of QA

[11] Modern question answering system might use retriever-reader architecture where retriever tries to retrieve relevant documents and reader is used to get answer from retrieved documents .The more recent system such as GPT-3 and BART is based on transformer architecture. Dense representation which is based on CNN based LSTM models to make retrieval process more efficient. Unlike traditional TF-IDF and BM25, that use sparse representation, deep retrieval methods learn to encode question and documents into vector space.

Document retriever-It has different models like vector space, probabilistic model and language model.its one of main goal is to retrieve relevant documents from wikipedia or some unstructured documents.Then retrieved documents is processed through document filtering, document re-ranking and document selection to get relevant documents .

Answer Extraction/Reader-From relevant document, answer is retrieved by retrieving right passage, right sentence and based on score, correct answer is returned.It is one of the core component of modern QA system.Reader can be Extractive Reader, predicts answer span from retrieved documents and generative reader, generates answers in natural language using Seq2Seq models.

There is also different type of retriever called sparse and dense retriever. If we go more in details with dense retriever, there is representation based retriever, interaction-based retriever

4 Methodologies

In this paper [1], document is denoted as D. Each phrase consist of continous words .QA returns phrase which is argmax where f is scoring function.System map s to an answer string and evaluation as done by comparing predicted answer with gold answer.Retriever reader is a paradigm is being used to leverage first stage document retriever.It read top k documents and then scoring function is used .

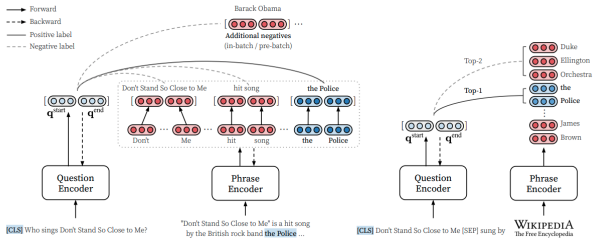

FIGURE 8. Single Passage Training and fine tuning

They learn high quality phrase encoder and question encoder along with incorporating negative sampling.By using negative sampling, the goal is to minimize similarity when words appear in diverse contexts and maximize similarity when they do.All the phrase is encoded and indexing is stored for efficient search .On top of it, Query-side fine tuning is being applied.There is phrase encoder and question encoder. Passage is divided into phrase, each phrase has start and end there is gold phrase .for given question, we find answer using argmax concept by denoting start and end indices. Then argmax is computed by performing MIPS and retrieve top k phrases for the given question.

A pre-trained language model is applied to get contextualized word representations for each passage token.Each phrase is represented as concatenation of start and end .Two different pre-trained encoders is being used M_s, start and M_e, end and which is used to get q_start and q_end .They maximized log-likelihood of the start and end position of gold phrases .using MIPS, get desired result for given question based on scoring function.They have used data augmentation, distillation to get better accurate result .To train question generation model, gold answer is fed and model is trained to maximize log-likelihood of question.

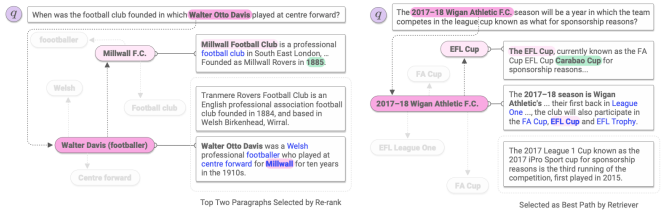

FIGURE 9. How Multihop QA works

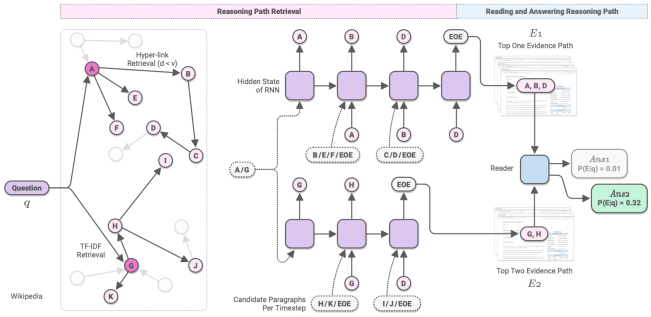

Figure 2: Overview of our framework.

FIGURE 10. Overview of [2] Framework

[2] worked on method which sequentially retrieve documents given history of previously retrieved documents .In this paper, they have constructed wikipedia paragraph graph using wiki links and model relationship between paragraphs.The retrieval model trains RNN to score reasoning path by maximizing probability of selecting correct paragraph and minimise the probability to select wrong paragraph.They used BERT to find tune paragraph.They used data augmentation and negative example to train models.They have used beam search decoding process which encode nodes on reasoning paths.Each article is divided into paragraphs lead to millions of paragraphs.Each paragraph become retrieving target.When we provide question q, the framework aims to derive answer by retrieving and reading reasoning paths.. They have used concept of hyperlinks to construct relationship between articles on web .The wiki graph covers wide range of topics.A RNN model is being used, models select paragraph from candidate paragraph.Bert is being used to encode each candidate paragraph.and then probability is computed that

pi is selected.The process is terminated, when end of evidence symbol is selected.They used wikipedia for open domain QA, article is divided into paragraphs, which is target to retrieve .The framework tries to derive answer based on question q by retrieving and reasoning paths.Retriever object which retrieve and selects reasoning paths E, related to question and reader obejective find answer .The reader model first verifies path and then give outputs .

Unlike [1], which generally give u paragraph based information and not so good with multihop question, [2] works really efficiently with multi hop questions.While in dense phrase, they use concept of vector space, in [2] they used the feature of graph.It selects answer and verify from selected reasoning paths which is not the case with [1] where they retrieve top k documents and then try to get the answer based on which has high probability score.In [2], they have also converted original machine reading comprehension datasets to sets of question and answer pairs, which is not the case with most of the papers.In [2] they have used TF-idf which is not the case with paper1.

while in [1], they selected top k documents, [2], they use the concept of timestamp and candidate set of paragraphs.They used beam search to overcome limitation of BERT which can also be used in some other model. while [1] has used negative sampling and data distillation to train model and get relevant result, [2] have used negative paragraph and data augmentation example .

Unlike in [1] and [2], in [3], the process of handling question and context is done separately at different layer.Unlike [1] and [2], it uses cross-attention mechanism which is better than self-attention.

Even when representing documents using graph which can be convenient when searching for dependent answer, it could be resource demanding and might not be so scalable.They tried to maximise likelihood of selecting correct document evidence while dense phrase does the same with paragraph.

unlike [2], [9] and [10] also solve multihop queries but [9] solve problem using dense retrieval and [10] uses Reasoning chains and [10] would work better because it uses the concept of entities and graph but [9] try to aggregate the documents which can be time consuming . [10] try to create graph using entities of similar context .

[3] explain mechanism of deformer, which is decomposition of pre-trained transformer-based models where lower layers process question and context separately higher layer process it together.Deformer process context through k layers offline and then cache it offline.During runtime, question is first processed and then text representation is loaded from cache.These K layer representation are fed into k+1 layer as input to continue through higher layers.In this paper they have worked on reducing loss by learning from original model along with distillation.Computation of transformer is defined for paired task containing two segments T_a and T_b.Then token embedding of segment T_a and T_b is concatenated as X =[A;B] .The Transformer encoder has n layers

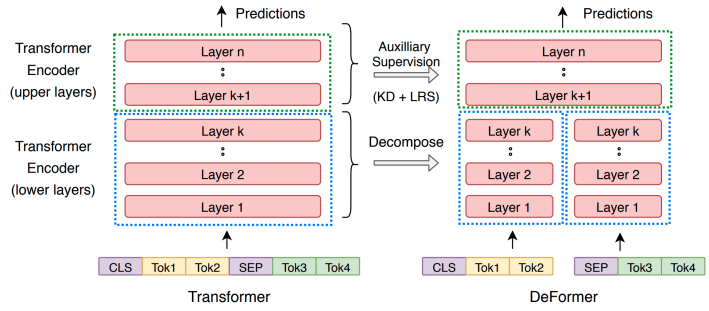

FIGURE 11. Decomposing Transformers up to layer k, which enables encoding each segment independently from layer 1 to layer k.

(L_i for layer i), transform input sequentially :$X^{l+1} = (\ L_i(\mathrm{X}^l))$

The output representation of full transformer, A^n and B^n, written as : $[A^n;B^n] = L_1 : n([A^0;B^0])$ The output representation of the decomposed Transformer,$_n$ and $_n$ can be expressed as:$[A^n;B^n] = L_k+1 : n([L_1 : k(A^0);L_1 : k(B^0))$.Transformer based QA process question and context together through stack of self-attention layers.This decomposition makes to process question and context independently.

Then they decomposed computation of lower layers by removing cross-interactions between T_b and T_b representation.Then using capability of deformers of retaining original structure, used to fine tune the model with pre-trained weights of original transformer.They initialized deformer parameters with pre-trained parameters and then downstream task is fine tuned.

one of the amazing thing author has done is that they have used cross attention for different layers.

[4] illustrate the goal of reading comprehension is to model the distribution p(a | q,c) where $q,c,a \in v$ represent a question,supporting context and answer respectively and v is vocabulary.Then the author focused on extracting reading comprehension where every question can be answered by selecting span of tokens .Then,it is assumed the probability of context span can be decomposed into product of p(start=i | q,c) and p(end=j | q,c) .They tried to model on Datasets having both source S and target database T, which achieves high in-domain accuracy.Then they fine tune multi datasets,The method to multi datasets reading comprehension is to fit a single model drawn uniformly from the datasets, where pre-trained transformer maps a question and context to a sequence of contextualized token embedding, to predict start and end tokens .They decompose model parameters into shared Transformer theta,token classifier and adapters.They used two phase Optimization process i.e Zero-shot Generalization and Transfer Learning Using MADE.In Zero shot generalization, they initialized new adapter and classifier by averaging parameters of pre-trained adapters and classifiers and return answer with highest probability.

While In BERTSerini, they have used IR toolkit as retriever

and BERT model as reader, this is not the case with this paper .In [5], they have used score of both reader and prediction score which is usually not the case with other method we are discussing here.

Unlike [3], [4] uses transformer with adapter which are specialized to different reading comprehensions.they have used lot of datasets for fine tuning unlike other papers.

[5] shows the process is composed of two main modules, The Anserini retriever and the Bert reader.The retriever is responsible for selecting parts of text from wikipedia that contains answer, passed to reader to identity answer span.They adopted single-stage retriever .At training time, they tried to retrieve l text segments using questions as a bag of words .The reader is based on google references implementation.They used BERT base models for training and find tuned model on SQuAD.Bert reader is applied on each paragraph.Then best sentence is retrieved .The reader selects best text span while combining score of reader and retriever.

$$S = (1-\mu) * S_{Anserini} + \mu * S_{Bert} \quad (1)$$

I could pin point that even though using BERT gave really significant improvement, Bert prove to work very bad in commonsense scenario.Also Bert reader work as reading document paragraph by paragraph which can also be challenging and time consuming .

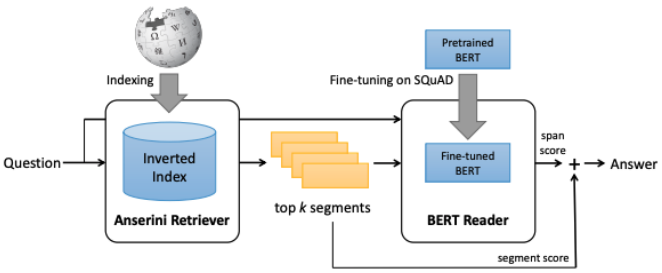

FIGURE 12. Architecture of BERTserini

Multi step retriever reader for open domain question answering [6]

The methodology for open-domain question answering presented in this paper involves iterative interaction between the reader and the retriever

Main focus on retriever model, it should be fast and interactive with reader model and includes precomputed caches stored offline, full of paragraph embeddings represent as context.

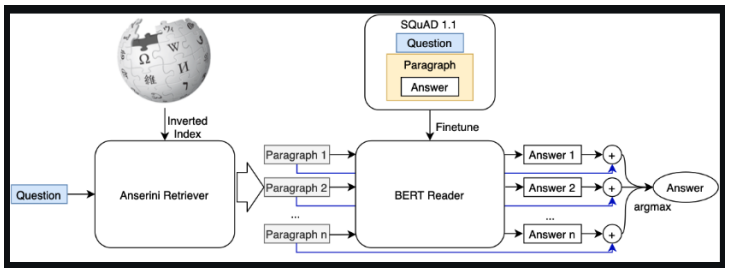

FIGURE 13. Architecture of BERTserini on Paragraph level

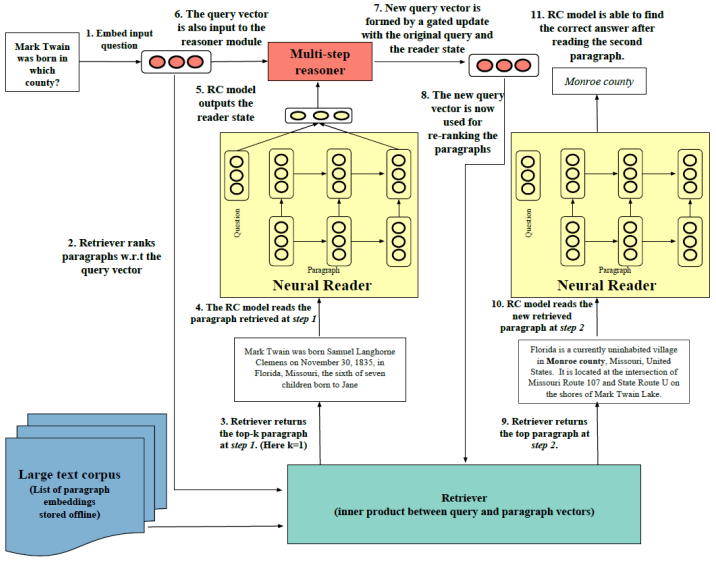

FIGURE 14. structure is illustrated by two stages in this paper. The reader receives the top k paragraphs after the first query has been encoded. The internal state of the reader model and the previous query vector are fed into the multi-step-reasoner component of our model, which performs a gated update to create a reformulated question. The retriever reorders the paragraphs and sends new paragraphs to the reader using this new query vector. As a result, the multi-step-reasoner makes it easier for the reader and the retriever to interact iteratively (QA model)

The architecture contains three components which is described in this paper:

paragraph retriever: The paragraph retriever calculates a score for how likely it is for a paragraph to include an answer to a certain inquiry. Given paragraph contains tokens, multilayer RNN encodes each token. Contextual information's are hidden state for LSTM, combine the representation that forward and backward LSTM's computed. Then compute paragraph vector combined with weights, generate the significance of every token. To obtain a query vector, use a different network with the same architecture. Than the relevance rate of paragraph with respect to question computed by simple inner product.

reader: The retriever sends the first few paragraphs to the reader, a highly developed neural machine reading comprehension (MRC) model, which then generates a range of answer text. This model is independent of the reader's precise architecture.
multi-step-reasoner: It provides multi step interaction between search engine and QA model to reformulate the query and re-rank the paragraph.
The framework just needs access to the reader's token-level hidden representations, regardless of how the machine reading model is built. The retriever scales to corpora including millions of paragraphs by using quick nearest neighbour search.
The question is updated at each stage depending on the reader's state by a gated recurrent unit, and the retriever uses the newly reformulated query to re-rank the paragraphs.

PullNet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text [7]

The PullNet framework uses an iterative process to construct a question specific subgraph and find the answer from subgraph.

[6] only work for large corpora but this model can get answer by using corpus, Knowledge graph or both

It will build a question subgraph that can be used to answer a given question using PullNet. The question subparagraph is built incrementally. At first, the subparagraph solely depends on the query. PullNet then extends the subgraph repeatedly by selecting nodes from which to "draw" data from, the KB or corpus as necessary. The question subparagraph is heterogeneous and includes entity-linked text as well as KB triples and entities.The question subgraph iteratively expanded until it has the needed knowledge for generating the answer.

Pull operations are used to extract information from knowledge sources, entities, facts, or documents. Information extraction similarly in [6] is done by LSTM but here sigmoid function of dot product of the last stage of LSTM is used to find the similar facts which relevant to the question

Classify operation gives a likelihood that an entity node should be used up in the following iteration. After the subgraph completion, The final answer is the entity node with the greatest score.

Update operation add the recently retrieved entity nodes to the update question subgraph.

[7] uses graph and subgraph, where node store entities from the knowledge base,documents from corpus, unlike in [2], uses graph where node has wikipedia documents whereas edge contain hyperlinks.

Answering Complex Open-domain Questions Through Iterative Query Generation [8]

This paper introduced a method called GOLDEN (Gold Entity) Retriever, which responds to open-domain multi-hop queries by interpreting context and retrieving more supplementary materials. The main novelty is that the model continuously generates new natural language queries and retrieves additional evidence to answer the first inquiry using IR (Information retrieval) results from prior hops of reasoning.

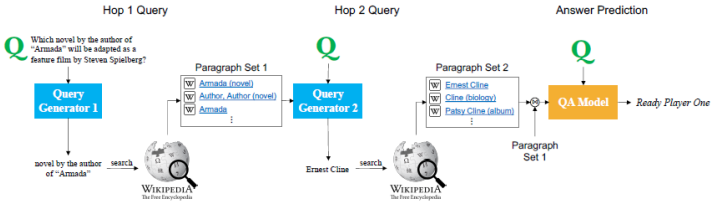

FIGURE 15. GOLDEN Retriever model overview. The model concatenates all retrieved context for a QA model to answer from, iteratively retrieving new context documents in response to an open-domain multi-hop query.

The original query is delivered to GOLDEN Retriever at the first hop of reasoning. Instead of totally depending on question for next reasoning steps, text-based IR engines used.

In every hop of reasoning step, there is query generator. It is generally search query (gold supporting documents) initially given the original question and some context of document previously retrieved by query generator (initially empty), which will help to find the next reasoning step.

A Document Reader model is trained to select a span from retrieval context as the query, which is ideally gold supporting documents. These documents are concatenated with the current retrieval context to update it.

At the end of the retrieval steps, IR retrieves top n documents from the search engine using query generated by query generator.

Finally question and retrieved context are feed into the Question Answering Component for getting the final output.

In comparison to its neural retrieval competitors [6], GOLDEN Retriever is a lot more effective, scalable, and interpretable at retrieving gold documents. At each step also

incorporates IR results from prior hops of reasoning to construct a new natural language query and collect new evidence to support the original question, rather than solely depending on the original question to do so. Which defer from [7] .It work better for more complex and diverse question

Both [5] and [8] uses IR as part of answering question but [8] uses golden retriever which is unusual to other papers,find most similar documents,ideal answer to questions

Answering complex open domain questions with multi hop dense retrieval [9]

This paper introduce a simple efficient multi hop dense retrieval approach for answering the complex question by aggregating information from multiple documents which requires logical reasoning or sequential (multi-hop) processing in order to infer the answer

In order to answer a difficult topic that necessitates the use of sequential (multi-hop) processing or logical reasoning, this study introduces a straightforward, effective multi hop dense retrieval approach.

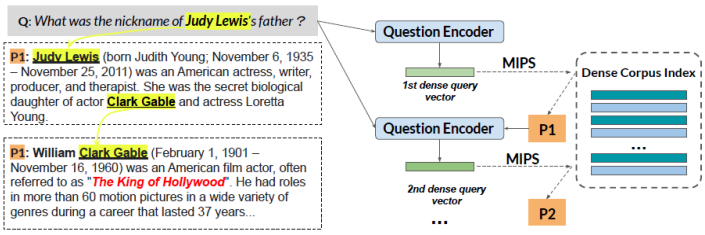

FIGURE 16. An overview of the multi-hop dense retrieval approach from this paper.

For choosing passage, constructed a new query representation based on prior results at each retrieval step, and the retrieval is carried out using maximum inner product search over the dense representations of the entire corpus.

For train retriever model, to approach the SoftMax overall passage, a positive and numerous negative passage are paired with each input query. Used a memory bank Similar to [6], in addition to in-batch negatives to store a large number of density passage vectors.

The entire corpus was encoded into a passage vector index for inference. Get the top k passage sequence candidates using beam search, where the candidates for beam searching are produced by MIPS using the query encoder at each step. The top-k sequences will then be routed into task-specific downstream modules to produce the desired outputs, then beams are rated based on the total of inner products.

Previous multi hop open domain question answering [7] models exponentially grow with the every retrieval step. So for overcome that back lock, MIPS used to get the next relevant document by recursively apply on question and prior extracted documents.

Multi-hop Question Answering via Reasoning Chains [10]

The methodology for open-domain question answering presented in this paper involves iterative interaction between the reader and the retriever

Introduced a technique for extracting a discrete reasoning chain, a collection of phrases that leads to an answer, from a text was developed. In order to forecast the final result, the extracted chains input into a BERT-based QA model. It's a two-stage paradigm that separates the answer from the intermediate chains of reasoning.

This model learn to extract chains by deriving heuristic reasoning chains. It's actually, from representation of sentence as node in the graph, checks the connection between sentences on shared entities and add an edge between similar nodes. This exhaustive search could lead to the answer and the answer lies on the chain with highest score.

The chain extraction model has 2 parts:

Sentence encoding: By using pre-Trained BERT encoder computes the representation of a sentence. These schema of sentence representation is called BERT-Para

Unlike in [10], [2] The only nodes on the reasoning paths are encoded throughout the beam search process, which greatly decreases the processing expenses..In [10] using beam search to explore a set of possible chains,which gives more close answer to question. In [9],uses beam search to obtain top k passage,not the documents using mips similar to [**?**] but in [?] they didn't use beam search

Chain prediction: LSTM-based pointer network for the purpose of chain reasoning used and treat all of the encoded sentence representations as a bag of sentences.

Training the **chain extractor** with WikiHop and HotpotQA datasets. Following [9] at evaluation, time beam search used to

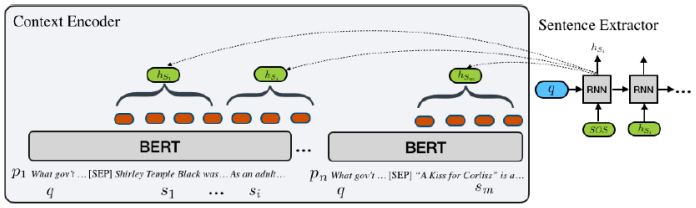

FIGURE 17. The BERT-Para variant of chain extractor from this paper. Left side: each document encoded by jointly with the question and employ pooling to create sentence representation Right side: pointer network used to extracts a sequence of sentences.

explore possible number of chains containing different number of sentences. Taking into account the top k beams predicted by the chain extractor as input to the answer prediction model, then the final answer is projected.

5 Future Work

The text based open domain question answering could also be used for visual open domain question answering on large scale.Even though over the years, a lot of improvement has been one and it shows more accuracy, there is still opportunity to improve the accuracy and how the model can be made more efficient so that new sources/current,recent data could be fed to model so that the whole QA system could be updated and could give better precised correct answers .If this is achieved,it can replace all the search engine and application like ChatGpt can become one door solution for all kinds of questions.There could be lot of new work in this field which will follow path as followed by chatGpt and make in near future, QA can give u prediction based answer on futute event like who is going to be president of USA in next election based on model which is being trained on datasets which take the consideration of human views on election likewise it would have similar application in different domain. There is enormous opportunity to do amazing work in open domain question answering .

6 Conclusion

In this survey, we discussed about 10 different methods which is being developed at different timeline and how different method works. We think Single-dataset Experts for Multi-dataset Question Answering works so better than most of the methods. we have talked about how to answer complex open domain question answering by using either multiple steps or hops on retriever model, or iterative query generator. Others model are used chain and graph architecture. [6] introduced iterative model where intra components(retriever and reader) of the model interact each other iteratively, which helps the method to scale up to millions of paragraphs. This model is independent of the machine reading system, as long as it have access to the token level hidden representation of the reader. On the other hand, [7] has the advantage to learning from Knowledge Base, Corpus or both. It not only iteratively operate as [6] but also build a query related subgraph. GCNN was utilized to determine the subgraph nodes that should expand in order to keep the graph smaller. In every iteration the documents grows exponentially, this problem was focused by [8] and introduce GOLDEN Retriever. Compared to its neural retrieval competitors [6], it is far more effective, scalable, and interpretable for retrieving gold documents. For answering the complex query need vast information from different parts of the documents. Previous models could do better by multi hop reasoning but often display a constrained set of multi-hop phenomena. In order to resolve this issue [10] introduced intermediate reasoning chain. Which shows better performance on two multi-hop question answering datasets: WikiHop and HotpotQA. [1], worked on wikipedia scale,they used encoders, negative sampling to get accurate result .reduces storage footprint.improve latency . [2],we see they have used RNN approach using help of reasoning path over wiki graphs.It gets the best answer from best reasoning paths. [3] solves one of the bottleneck of input wide self-attention computation at each layer in transformer based model .The evaluation of both BERT and XL-Net,deformer version is done.Bert version is better than smaller Bert version. while original bert had cross attention but deformer bert doesn't have. [4],Multiset solves the problem of overfitting and underfitting.This approach gives better in-domain accuracy . [5],As we discussed they have removed softmax layers .They integrated BERT and the Anserini IR toolkit which increased performance really well.As we have elaborated above the [2], [9] deals with real life most practical situation where user generally ask question which can't be found in one single documents or paragraph.

REFERENCES

[1] Lee, J., Sung, M., Kang, J., and Chen, D., 2020. "Learning dense representations of phrases at scale". *arXiv preprint arXiv:2012.12624*.

[2] Asai, A., Hashimoto, K., Hajishirzi, H., Socher, R., and Xiong, C., 2019. "Learning to retrieve reasoning paths over wikipedia graph for question answering". *arXiv preprint arXiv:1911.10470*.

[3] Cao, Q., Trivedi, H., Balasubramanian, A., and Balasubramanian, N., 2020. "Deformer: Decomposing pre-trained transformers for faster question answering". *arXiv preprint arXiv:2005.00697*.

[4] Friedman, D., Dodge, B., and Chen, D., 2021. "Single-dataset experts for multi-dataset question answering". *arXiv preprint arXiv:2109.13880*.

[5] Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M., and Lin, J., 2019. "End-to-end open-domain question answering with bertserini". *arXiv preprint arXiv:1902.01718*.

[6] Das, R., Dhuliawala, S., Zaheer, M., and McCallum, A., 2019. "Multi-step retriever-reader interaction for scalable open-domain question answering". *arXiv preprint arXiv:1905.05733*.

[7] Sun, H., Bedrax-Weiss, T., and Cohen, W. W., 2019. "Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text". *arXiv preprint arXiv:1904.09537*.

[8] Qi, P., Lin, X., Mehr, L., Wang, Z., and Manning, C. D., 2019. "Answering complex open-domain questions through iterative query generation". *arXiv preprint arXiv:1910.07000*.

[9] Xiong, W., Li, X. L., Iyer, S., Du, J., Lewis, P., Wang, W. Y., Mehdad, Y., Yih, W.-t., Riedel, S., Kiela, D., et al., 2020. "Answering complex open-domain questions with multi-hop dense retrieval". *arXiv preprint arXiv:2009.12756*.

[10] Chen, J., Lin, S.-t., and Durrett, G., 2019. "Multi-hop question answering via reasoning chains". *arXiv preprint arXiv:1910.02610*.

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