# A SURVEY ON MULTI-HOP QUESTION ANSWERING

**Tushar Abhishek**
2019701015
tushar.abhishek@research.iiit.ac.in

**Vivek Kumar**
2019701004
vivek.k@research.iiit.ac.in

**Rishabh Maheshwary**
2019701023
rishabh.maheshwary@research.iiit.ac.in

## Abstract

In the ever increasing need of a human-like Question Answering system, we here discuss upon various approaches and techniques developed in the domain of Multi-hop Question Answering and try to provide a literature survey on the ongoing trends in this domain. This work provides an in-depth treatment on the approaches, their results shortcomings and future scope.

## 1 Introduction

Reading comprehension, ability to read the text and then answer it, is one of the challenging tasks which not only requires understanding of natural language but also about the world. The question answering(QA) provides a quantifiable and objective way to test the reasoning ability of intelligent systems.

In order to assess this, few datasets were previously presented, SQuAD(1) was introduced as a dataset which captures this task but answers were restricted to sentences within one paragraph. Most of the questions in SQuAD can be answered by carefully examining the similarity between a single sentence and given question. HotPotQA(7) dataset was introduced that requires reasoning over multiple documents, and does so in natural language, without constraining itself to any existing knowledge base or knowledge schema similar to QAngaroo(8). It provides the datasets where answering the question requires reasoning on more than one sentence and are called multi-hop QA. We have also worked on another multi-hop in open-domain question answering, WikihopQA(9). In this report we describe in detail and compare four different models which solves QA problems described in hotpotQA and wikiQA datasets.

DecomRC(10), model decomposes multiple-hops questions into different single hop sub-questions of different reasoning types. Then answers to sub-questions are obtained using a one-hop reading comprehension model. Another approach similar to question decomposition, Golden Retriever[10] which is an iterative method over reading paragraphs (contexts) and retrieving more supporting documents to answer open-domain multi-hop questions. This approach is different in that it works in open-domain QA with absence of documents which are required to answer the question. An altogether different approach was applied on multi-hop QA through PathNet(12) model. It relies on a path-based reasoning approach where a system needs to combine facts from multiple passages to answer a question. It attempts to extract implicit relations from text through entity pair representations, and compose them to encode each path along with passage representations for each path to compute a passage-based representation.

Natural Language Processing & Applications (2020).

## 2 Literature Survey

We will be describing three different datasets on multiple-hop question answering. All three datasets are conceptually different from one another on reasoning type and information associated with them.

### 2.1 HotpotQA

It contains 1,12,779 collection of question answer pairs collected on Amazon Mechanical trunk using the ParlAI(2) interface. It consists of both single and multiple hop questions, and to efficiently evaluate the learning of the models these are categorised into different categories as follows:

- **Train-easy**: The authors of the hotpotQA(7) paper have randomly sampled questions from top-contributing turkers and categorised all their questions as train-easy if a large percentage of their questions were answered by reasoning over a single paragraph.
- **Train-medium**: The baseline model of hotpotQA(7) paper was used on the remaining question(apart from train-easy examples) and three fold cross validation was performed on this question set. Among instances of question sets, those questions whose answer was predicted by the model with high confidence were selected to form the train-medium examples. This corresponds to about 60% of the total question.
- **Train-hard**: The remaining questions form hard examples. It was further subdivided into four categories namely : train-hard, dev, test-distractor and test-fullwiki.

| Name | Desc. | Usage | # Examples |
|---|---|---|---|
| train-easy | single-hop | training | 18,089 |
| train-medium | multi-hop | training | 56,814 |
| train-hard | hard multi-hop | training | 15,661 |
| dev | hard multi-hop | dev | 7,405 |
| test-distractor | hard multi-hop | test | 7,405 |
| test-fullwiki | hard multi-hop | test | 7,405 |
| Total | | | 112,779 |

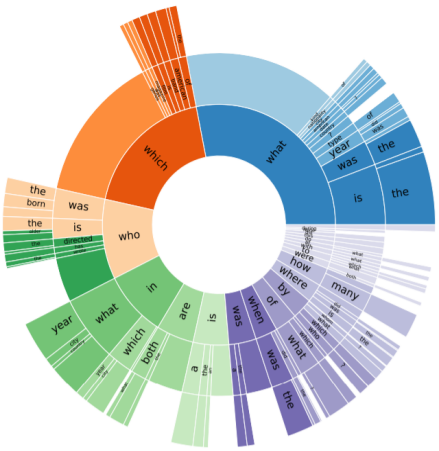


Figure 1: HotpotQA Statistics

Question and their answer types are heuristically analysed in the hotpotQA(7) paper. In order to identify the types of question, central question word (*CQW*) was identified. Words which are considered as question words are : wh-words, auxiliary verbs ('does', 'did') and copulas ('is', 'are'). Certain rules were described in the paper(7) to extract CQW from question words, like question often involves the relative clauses beginning with wh-words, so the first question word if it's found in the first three tokens or last two otherwise will be considered as CQW. The question type is then determined by extracting 2 tokens to the right. It can be visualized in the below figure and label the one shared among more than 250 questions.
Apart from question types these QA examples can also be identified into different answer types.These can be described as follows:

- **Type-1 (Chain reasoning)**: A majority of sampled questions (around 42%) requires chained reasoning, where the reader must first identify the bridge entity before the second hop can be answered by filling the in the bridge entity.
- **Type-2 (Checking multiple properties of chained entity**: To answer the question, one could find a set of all entities that satisfy each of the properties mentioned and take inter-

section to arrive at answers. Also note this is different from type-3 as answers to questions contain only bridged entities not the properties of the bridged entities.

- **Type-3 (Inferring about property of chained entity)**: In some questions, the entity in question shares certain properties with the bridge entity and can infer properties using the bridged entity.
- **Comparisons** : Questions which require the system to understand the comparison between the specific common property of two entities. Also answer to this comparison question might be simple as yes/no or can be arithmetic such as counting numbers or comparing numerical values.
- **Others (Requires more than 2 supporting facts)** : Sometimes the questions requires more than 2 supporting facts to answer. And these questions are classified as others in reasoning type.

### 2.2 WikiQA

WikiHop(13) is a large-scale multi-hop question answering data set consisting of about $51K$ questions. Each question is associated with an average of $13.7$ supporting Wikipedia passages, each with $36.4$ tokens on average. The questions are divided into training, development and test sets with 43,738 questions in training,$5,129$ questions in development and $2,451$ in test sets. Figure shown below depicts a sample question from the dataset. In WikiHop, a question $Q$ is given in the form of a tuple $(h, r, ?)$, where $h$ represents the head entity, and $r$ represents the relation between $h$ and the unknown tail entity, which is represented by a question mark. The task is to select the unknown tail entity from a given set of candidates

$c1, c2, ...cN$, by reasoning over supporting passages $P = p1, ..., pM$ .

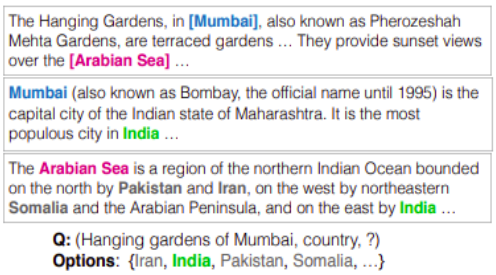


Figure 2: WikiHop Example

### 2.3 OpenBookQA

The OpenBookQA dataset(14) consists of about $6,000$ 4-way multiple-choice questions, each associated with one core fact from a "book" F of 1326 such facts, and an auxiliary set K of about 6000 additional facts. The small "book" $F$ consists of recurring science themes and principles, each of which can be instantiated into multiple questions.OpenBookQA additionally requires broad common knowledge, which is expected to come from large corpora, such as ConceptNet, Wikipedia, or a corpus with $14M$ science-related sentences used by some existing baselines.

## 3 Research Models

### 3.1 Baseline

The baseline QA model as described in HotpotQA(7) paper as the base model which is basically the re-implementation of architecture described in Clark and Gardner (2017) model(3). Apart from returning the answer span (sequence of words from paragraph), the model was modified to respond

**Question:**
*Which of these would let the most heat travel through?*
A) a new pair of jeans.
B) a steel spoon in a cafeteria.
C) a cotton candy at a store.
D) a calvin klein cotton hat.

**Science Fact:**
Metal is a thermal conductor.

**Common Knowledge:**
Steel is made of metal.
Heat travels through a thermal conductor.

Figure 3: OpenBookQA Example

to the question with either “yes“ or “no“. It also incorporates the latest advances in QA including character-level models, self-attention(5), and bi-attention(6).

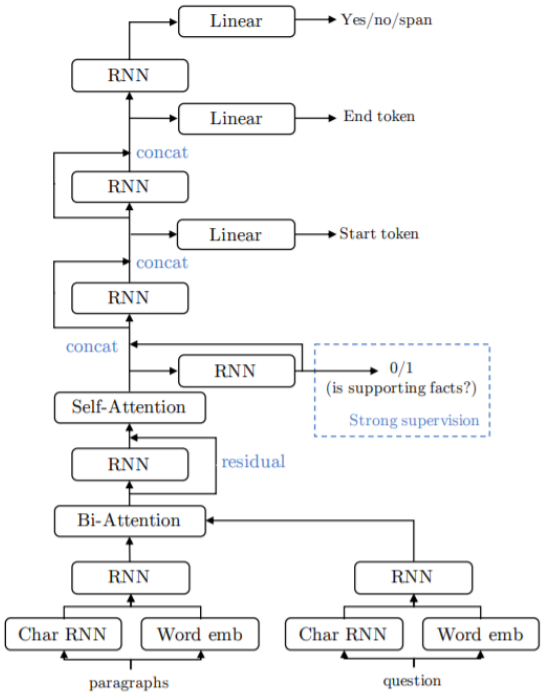


Figure 4: Model architecture for baseline.

The notable changes in the architecture apart from adding an extra layer and binary classification layer for supporting fact is concatenating this output with the start, end and comparison classifier. The supporting fact classifier is trained against the cross entropy loss function. This objective is jointly optimized with the normal question answering task in a multi-task learning setting.

### 3.2 Multi-hop RC through Question Decomposition and Rescoring

As the name suggests Decomposition and Rescoring are two different modules in DecomRC architecture, First, DecomRC decomposes the original, multi-hop question into several single-hop sub-questions according to a few reasoning types in parallel, based on span predictions. Then for every reasoning type DecomRC leverages a single-hop reading comprehension model to answer each sub-question, and combines the answers according to the reasoning type. There is another module called Decomposition scorer which connects the single-hop model to multiple-hop model, DecomRC

leverages a decomposition scorer to judge which decomposition is the most suitable, and outputs the answer from that decomposition as the final answer.

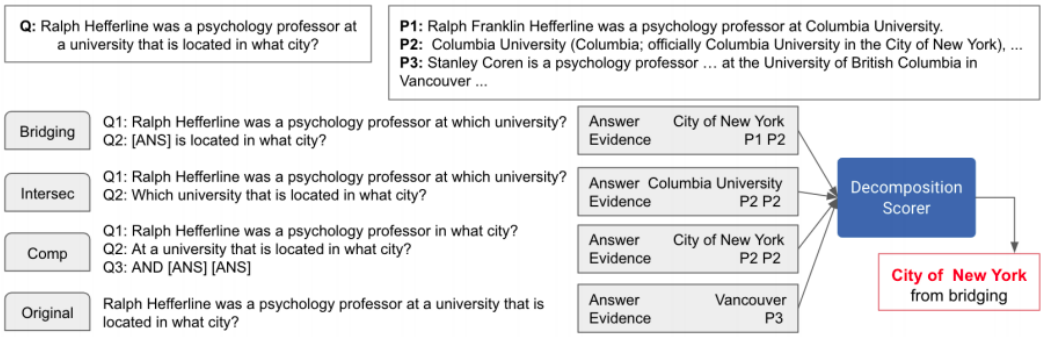


Figure 5: Model decision flow at various stages

The authors of this paper have identified several reasoning types in multi-hop reading comprehension, which they have used to decompose the original question and rescore the decompositions. These reasoning types are bridging, intersection and comparison. On a sample of 200 questions from the dev set of HotpotQa, they found out that 92% of multi-hop questions belong to one of these types. Specifically, among 184 samples out of 200 which require multi-hop reasoning, 47% are bridging questions, 23% are intersection questions, 22% are comparison questions, and 8% do not belong to one of three types.

### 3.2.1 Question Decomposition

Question decomposition module converts a multi-hop question into simpler, single-hop subquestions. It creates sub-questions using span prediction over the question. The key idea is that, in practice, each sub-question can be formed by copying and lightly editing a key span from the original question, with different span extraction and editing required for each reasoning type. They have trained the

| | |
|---|---|
| **Type** | **Bridging (47%)** requires finding the first-hop evidence in order to find another, second-hop evidence. |
| Q | Which team does the **player** named 2015 Diamond Head Classic's MVP play for? |
| Q1 | Which player named 2015 Diamond Head Classic's MVP? |
| Q2 | Which team does ANS play for? |
| **Type** | **Intersection (23%)** requires finding an entity that satisfies two independent conditions. |
| Q | Stories USA starred ✓ which actor and comedian ✓ from 'The Office'? |
| Q1 | Stories USA starred which actor and comedian? |
| Q2 | Which actor and comedian from 'The Office'? |
| **Type** | **Comparison (22%)** requires comparing the property of two different entities. |
| Q | Who was born earlier, Emma Bull or Virginia Woolf? |
| Q1 | Emma Bull was born when? |
| Q2 | Virginia Woolf was born when? |
| Q3 | Which_is_smaller (Emma Bull, ANS) (Virgina Woolf, ANS) |

Figure 6: The example multi-hop questions from each category of reasoning type on HotpotQA.

question decomposition model with little supervision ( 400 annotated samples). They have trained a model $Pointer_c$ that learns to map a question into $c$ points, which are subsequently used to compose sub-questions for each reasoning type. Also 'c' represents the hyper-parameter which was chosen in advance for each reasoning type. For example for bridging, 'c' is 3 (start, keyword and end position), for intersection 'c' is 2 (start and end positions) and for comparison 'c' is 4 (start & end of first entity and start & end of last entity).
$Pointer_c$ is a function that points to $c$ indices $ind_1, ..., ind_c$ in an input sequence. Let $S = [s_1, ..., s_n]$ denote a sequence of $n$ words in the input sequence. The model encodes $S$ using BERT(15).

$$U = BERT(S) \in \mathcal{R}^{n \times h} \tag{1}$$

where $h$ is the output dimension of the encoder. Let $W \in R^{h \times c}$ denote a trainable parameter matrix. It computes a pointer score matrix

$$Y = softmax(UW) \in \mathcal{R}^{n \times c} \tag{2}$$

where $\mathcal{P}(i = ind_j) = Y_{ij}$ denotes the probability that the ith word is the jth index produced by the pointer. The model extracts $c$ indices that yield the highest joint probability at inference:

$$ind_1, ..., ind_c = \underset{i_1 \leq ... \leq i_c}{\operatorname{argmax}} \prod_{j=1}^{c} \mathcal{P}(i_j = ind_j) \tag{3}$$

### 3.2.2 Single-hop Reading Comprehension

Given a decomposition, we use a single-hop RC model to answer each sub-question. Specifically, the goal is to obtain the answer and the evidence, given the sub-question and $N$ paragraphs. The answer is a span from one of paragraphs, yes or no. The evidence is one of $N$ paragraphs on which the answer is based. For single-hop reading comprehension, any off-the-shelf RC model can be used but the paper states the use of BERT reading comprehension model(15) combined with the paragraph selection approach from Clark and Gardner(3) to handle multiple paragraphs.

Given $N$ paragraphs $S_1, ..., S_N$ , this approach independently computes $answer_i$ and $y_i^{none}$ from

**Algorithm 1** Sub-questions generation using $\text{Pointer}_c$.[2]

**procedure** GENERATESUBQ($Q$ : question, $\text{Pointer}_c$)
- */* Find $q_1^b$ and $q_2^b$ for Bridging */*
- $\text{ind}_1, \text{ind}_2, \text{ind}_3 \leftarrow \text{Pointer}_3(Q)$
- $q_1^b \leftarrow Q_{\text{ind}_1:\text{ind}_3}$
- $q_2^b \leftarrow Q_{:\text{ind}_1}$ : ANS : $Q_{\text{ind}_3:}$
- article in $Q_{\text{ind}_2-5:\text{ind}_2} \leftarrow$ 'which'
- */* Find $q_1^i$ and $q_2^i$ for Intersecion */*
- $\text{ind}_1, \text{ind}_2 \leftarrow \text{Pointer}_2(Q)$
- $s_1, s_2, s_3 \leftarrow Q_{:\text{ind}_1}, Q_{\text{ind}_1:\text{ind}_2}, Q_{\text{ind}_2:}$
- **if** $s_2$ starts with wh-word **then**
  - $q_1^i \leftarrow s_1 : s_2,\ q_2^i \leftarrow s_2 : s_3$
- **else**
  - $q_1^i \leftarrow s_1 : s_2,\ q_2^i \leftarrow s_1 : s_3$
- */* Find $q_1^c$, $q_2^c$ and $q_3^c$ for Comparison */*
- $\text{ind}_1, \text{ind}_2, \text{ind}_3, \text{ind}_4 \leftarrow \text{Pointer}_4(Q)$
- $\text{ent}_1, \text{ent}_2 \leftarrow Q_{\text{ind}_1:\text{ind}_2}, Q_{\text{ind3:ind}_4}$
- $op \leftarrow \text{find\_op}(Q, \text{ent}_1, \text{ent}_2)$
- $q_1^c, q_2^c \leftarrow \text{form\_subq}(Q, \text{ent}_1, \text{ent}_2, op)$
- $q_3^c \leftarrow op\ (\text{ent}_1, \text{ANS})\ (\text{ent}_2, \text{ANS})$

Figure 7: Question decomposition algorithm

each paragraph $S_i$ , where $answer_i$ and $y_i^{none}$ denotes the answer candidate from $i^{th}$ paragraph and the score indicating $i^{th}$ paragraph does not contain the answer. The final answer is selected from the paragraph with the lowest $y_i^{none}$. Although this approach takes a set of multiple paragraphs as an input, it is not capable of jointly reasoning across different paragraphs. For each paragraph $S_i$, let $U_i \mathcal{R}^{nh}$ be the BERT encoding of the sub-question concatenated with a paragraph $S_i$, obtained by BERT. They have computed four scores, $y_i^{span}, y_i^{yes}, y_{noi}$ and $y_i^{none}$, indicating if the answer is a phrase in the paragraph, yes, no, or does not exist.

$$[y_i^{span}, y_i^{yes}, y_{noi}, y_i^{none}] = max(U_i)W_1 \in \mathcal{R}_4$$

where max denotes a max-pooling operation across the input sequence, and $W_1 \in \mathcal{R}^{h \times 4}$ denotes a parameter matrix. Additionally, the model computes $span_i$, which is defined by its start and end points $start_i$ and $end_i$.

$$start_i, end_i = \underset{j \leq k}{\operatorname{argmax}} \mathcal{P}_{i,start}(j) \mathcal{P}_{i,end}(k)$$

where $\mathcal{P}_{i,start}(j)$ and $\mathcal{P}_{i,end}(k)$ indicate the probability that the $j$th word is the start and the $k$th word is the end of the answer span, respectively. $\mathcal{P}_{i,start}(j)$ and $\mathcal{P}_{i,end}(k)$ are obtained by the $j$th element of $p_i^{start}$ and the $k$th element of $p_i^{end}$ from

$$p_i^{start} = softmax(U_i W_{start}) \in \mathcal{R}^n$$
$$p_i^{end} = softmax(U_i W_{end}) \in \mathcal{R}^n$$

The model is trained using questions that only require single-hop reasoning, obtained from SQUAD(1) and easy examples of HotpotQA(7) easy samples. Once trained, it is used as an off the-shelf RC model and is never directly trained on multi-hop questions.

#### 3.2.3 Decomposition scorer

Each decomposition consists of sub-questions, their answers, and evidence corresponding to a reasoning type. DecomRC scores decompositions and takes the answer of the top-scoring decomposition to be the final answer. The score indicates if a decomposition leads to a correct final answer to the multi-hop question.Let $t$ be the reasoning type, and let $answer_t$ and $evidence_t$ be the answer and the evidence from the reasoning type $t$. Let $x$ denote a sequence of $n$ words formed by the concatenation of the question, the reasoning type $t$, the answer $answer_t$, and the evidence $evidence_t$. The decomposition scorer encodes this input $x$ using BERT to obtain $U_t \in \mathcal{R}^{n \times h}$.The score $p_t$ is computed as

$$p_t = sigmoid(W_2^T max(U_t)) \in R$$

where $W_2 \in \mathcal{R}^h$ is a trainable matrix. During inference, the reasoning type is decided as $\mathrm{argmax}_t\, p_t$. The answer corresponding to this reasoning type is chosen as the final answer.

### 3.3 Exploiting Explicit Paths for Multi-hop Reading Comprehension

In this work authors have proposed a path-based reasoning approach for the multi-hop reading comprehension task where a system needs to combine facts from multiple passages to answer a question.It generates potential paths through passages and scores them without any direct path supervision2 The proposed model, named PathNet, attempts to extract implicit relations from text through entity pair representations, and compose them to encode each path. The model also composes the passage representations along each path to compute a passage-based representation and then is able to explain its reasoning via these explicit paths through the passages.The model is applied on WikiHopQA dataset and generalized to perform on OpenBookQA as well.

#### 3.3.1 Approaches Used

The model focuses on the multiple-choice RC setting: Given a question and a set of passages, the task is to find the correct answer among a predefined set of candidates. The proposed approach can be applied to m-hop reasoning, as discussed briefly in the corresponding sections for path extraction, encoding, and scoring.

In WikiHop, a question $Q$ is given in the form of a tuple $(h, r, ?)$, where h represents the head entity, and r represents the relation between h and the unknown tail entity. In OpenBookQA, different from WikiHop, the questions and candidate answer choices are plain text sentences. To construct paths, we extract all head entities from the question and tail entities from candidate answer choices, considering all noun phrases and named entities as entities.Next, the extracted paths are encoded and scored. Following, the normalized path scores are summed for each candidate to give a probability distribution over the candidate answer choices. Figure below shows a sample example for the same.

#### 3.3.2 Path Extraction

Path extraction proceeds as follows:

- We find a passage $p_1$ that contains a head entity $h$ from the question $Q$. In our example figure it would identify the first supporting passage that contains always breaking my heart.
- We then find all named entities and noun phrases that appear in the same sentence as $h$ or in the subsequent sentence. Here, we would collect Belinda Carlisle, A Woman and a Man, and album as potential intermediate entity $e_1$ .

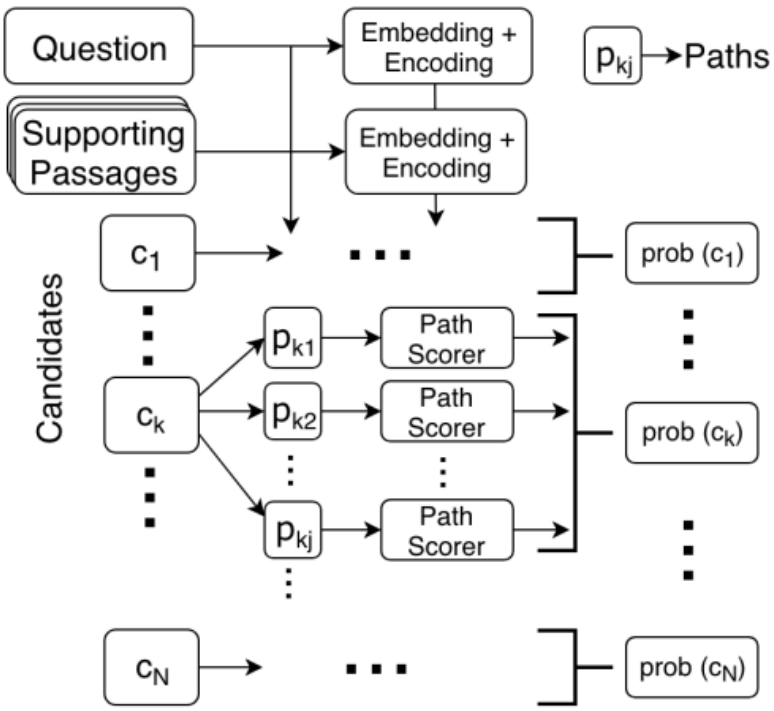


Figure 8: Architecture of model proposed.

- Next, we find a passage $p_2$ that contains the potential intermediate entity identified above.For instance, we find the second passage that contains both Belinda Carlisle and A Woman and a Man.
- Finally, we check whether $p_2$ contains any of the candidate answer choices. For instance,$p_2$ contains chrysalis records and emi group.

We compute the probability of a candidate $c_k$ being the correct answer by summing the normalized scores of the paths associated with $c_k$ :

The resulting extracted paths can be summarized as a set of entity sequences. Below figure gives an architecture of the proposed model. Next we discuss embedding, path encoding and path scoring.

#### 3.3.3 Embedding and Encoding

For word embedding, we use pre-trained 300 dimensional vectors from GloVe, randomly initializing vectors for out of vocabulary (OOV) words. For contextual encoding,we use bi-directional LSTM (BiLSTM). Let $T$ , $U$ , and $V$ represent the number of tokens in the $p^{th}$ supporting passage, question, and $k^{th}$ answer candidate, respectively and $H$ is the number of hidden units for BiLSTM. We then have:

#### 3.3.4 Path Encoding

Path Encoding is done in two ways:

- **Context Based Path Encoding:** This component aims to implicitly encode the relation between $h$ and $e_1$ , and between $e_1'$ and $c_k$ .These implicit relation representations are them composed together to encode a path representation for $h...e_1...e_1'..c_k$ . First, we extract the contextual representations for each of $h$ , $e_1$ , $e_1'$, and $c_k$ . Based on the locations of these entities in the corresponding passages, we extract the boundary vectors from the passage encoding representation.The location encoding vectors $g_{e1}$ , $g_{e_1'}$ and $g_{ck}$ are obtained similarly. Next we extract the implicit relation between $h$ and $e_1$ using a feed forward layer.

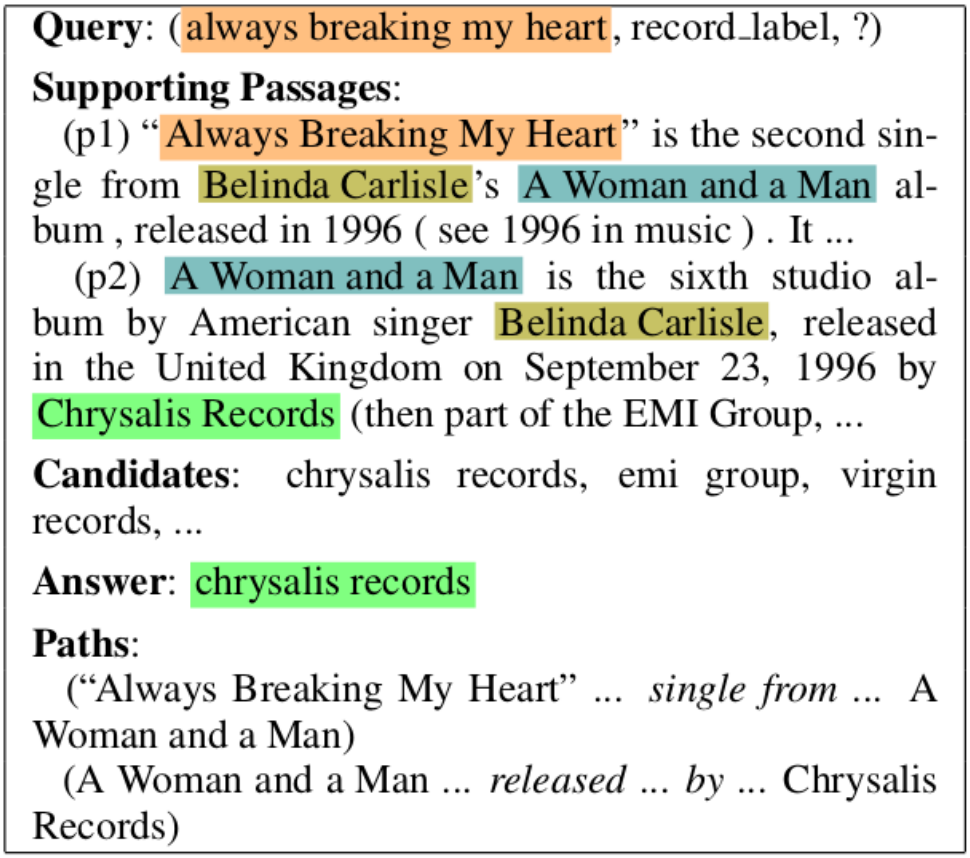


Figure 9: Example illustrating the proposed path extraction and reasoning approach.

$$\text{prob}(c_k) = \sum_j \text{score}(\text{p}_{kj}).$$

Similarly, we compute the implicit relation between $e'_1$ and $c_k$. Finally, we compose all implicit relation vectors along the path to obtain a context-based path representation $x_{ctx}$.

- **Passage Based Path Encoding:** We encode each of p1 and p2 into a single vector based on passage-question interaction. We first compute a question-weighted representation for passage tokens and then aggregate it across the passage. We first calculate attention matrix A capturing similarity between question and passage tokens. Next we calculate question aware passage representation $S_p^{q1}$ and passage aware question representation $Q_p$.

  Now, we move to aggregate representation of passages. We concatenate the two representations $S_p^{q1}$ and $S_p^{q2}$ and then use an attentive pooling mechanism to aggregate their token representations.

  We compose the aggregated passage vectors to obtain the passage-based path representation $x_{psg}$.

#### 3.3.5 Path Scoring

Path scoring is also done in two ways:

$$\mathbf{S}_p \in \mathbb{R}^{T \times H} \quad \mathbf{Q} \in \mathbb{R}^{U \times H} \quad \mathbf{C}_k \in \mathbb{R}^{V \times H}$$

– Context Based: First, we aggregate the question into a single vector. We take the first and last hidden state representations from the question encoding $Q$ to obtain an aggregated question vector representation.After that, we derive representation of context based path and derive final score $z_{ctx}$.

– Path Based: In this only candidate encoding is further used to evaluate $z_{psg}$.score for passage-based path is then computed as follows, where $z$ represents final score.

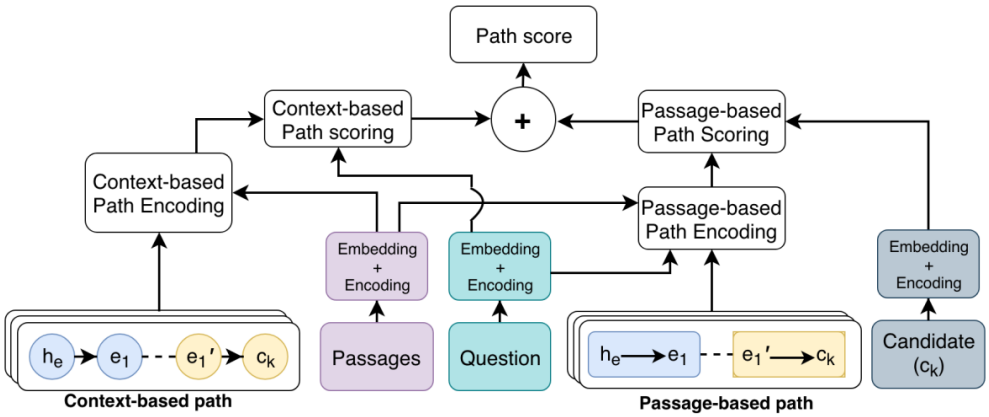


Figure 10: Architecture of the path scoring module, shown here for 2-hop paths.

### 3.4 Answering Complex Open-domain Questions Through Iterative Query Generation

In this work authors proposed *GoldEN Retriever* an approach to solve the problem of multi hop open domain question answering. As multi hop questions cannot be answered in a single step because for many such questions, not all the relevant context can be obtained in a single retrieval step. Therefore to solve such complex tasks authors propose a novel iterative retrieve-and-read framework. At each time step they launch an elastic search query to retrieve top K wikipedia articles, then for the next step they use the original question and the context (retrieved wikipedia articles) to generate another search query. The key insight authors based their approach on is that at any step of open-domain multi-hop reasoning, there is some semantic overlap between the retrieval context and the next document(s) we wish to retrieve. Finding this semantic overlap between the retrieval context and the desired documents not only reveals the chain of reasoning naturally, but also allows us to use it as the search query for retrieval. They used HotpotQA dataset for multihop reasoning. The statistics of the dataset are shown in Section 3, and the results are shown in Section 5.

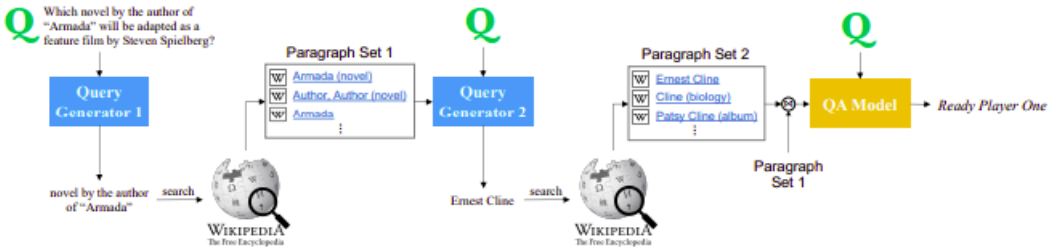


Figure 11: GoldEN Retriever architecture

### 3.4.1 Approach Used

- To reduce the potentially large space of possible queries, authors favor a QA model that extracts contiguous text spans from the retrieval context over one that generates free-form text as search queries. They employ DrQA's Document Reader model (Chen et al., 2017), which is a relatively light-weight recurrent neural network QA model.
- For each reasoning step $k = 1, ..., S$, given a question q and some retrieval context Ck which ideally contains the gold supporting documents $d_1, ..., d_{k1}$, they generate a search query $q_k$ that helps us retrieve $d_k$ for the next reasoning step. A Document Reader model is trained to select a span from Ck as the query

$$q_k = G_k(q, Ck), \tag{4}$$

  where $G_k$ is the query generator. This query is then used to search for supporting documents, which are concatenated with the current retrieval context to update it

$$C_{k+1} = CkIR_n(q_k) \tag{5}$$

  where IRn(qk) is the top n documents retrieved from the search engine using qk, and C1 = q. At the end of the retrieval steps, they provide q as question along with CS as context to the final few document QA components to obtain the final answer to the original question.
- To train the query generators, they follow the steps above to construct the retrieval contexts, but during training time, if dk is not part of the IR result, the lowest ranking document is replaced with dk before concatenating it with Ck to make sure the downstream models have access to necessary context.

**Query Dataset Generator**
For query dataset generation computing the longest common string/sequence between the current retrieval context and the title/text of the intended paragraph ignoring stop words, then taking the contiguous span of text that corresponds to this overlap in the retrieval context. This allows us to not only make use of entity names, but also textual descriptions that better lead to the gold entities. To generate candidates for the oracle query, they apply various heuristics between combinations of question and context. Once oracle queries are generated, we launch these queries against Elasticsearch to determine the rank of the desired paragraph.

**QA Component**
The final QA component is based on the BiDAF++ (Clark and Gardner, 2017). Two changes have been made; each wiki article is processed separately with shared encoder RNN parameters to obtain paragraph order-insensitive representations for each paragraph. The second change is that all attention mechanisms in the original model are replaced with self attention layers over the concatenated question and context. To differentiate context paragraph representations from question representations in this self-attention mechanism, the question and context tokens are indicated by concatenating a 0/1 feature at the input layer.

### 3.4.2 Ablation Studies

- To show the effectiveness of the oracle query they specifically compared the recall of the gold paragraph in both (Oracle & Single). If the recall is higher than the downstream QA system would have to deal with much less noise. The results are shown below.
- They also performed an ablation in which they replaced the query generators with their oracles. The results are shown in the table below.

## 3.5 Revealing the Importance of Semantic Retrieval for Machine Reading at Scale

In this work authors proposed*MRS* a hierarchical semantic retrieval pipeline to solve open domain downstream tasks such as QA and NLI. The authors highlighted that the upstream retrieval modules mostly focus on getting better coverage of the downstream information such that the upper-bound of the downstream score can be improved, rather than finding more exact information. The QA systems had to deal with more noise if the relevant passages are not provided. To solve they first rank all the wiki passages which are relevant to the

| | Question | Predicted $q_1$ | Predicted $q_2$ |
|---|---|---|---|
| (1) | What video game character did the voice actress in the animated film Alpha and Omega voice? | voice actress in the animated film Alpha and Omega *(animated film Alpha and Omega voice)* | Hayden Panettiere |
| (2) | What song was created by the group consisting of Jeffrey Jey, Maurizio Lobina and Gabry Ponte and released on 15 January 1999? | Jeffrey Jey *(group consisting of Jeffrey Jey, Maurizio Lobina and Gabry Ponte)* | Gabry Ponte and released on 15 January 1999 *("Blue (Da Ba Dee)")* |
| (3) | Yau Ma Tei North is a district of a city with how many citizens? | Yau Ma Tei North | Yau Tsim Mong District of Hong Kong *(Hong Kong)* |
| (4) | What company started the urban complex development that included the highrise building, The Harmon? | highrise building, The Harmon | CityCenter |

Figure 12: Examples of Oracle query (blue) vs Predicted query

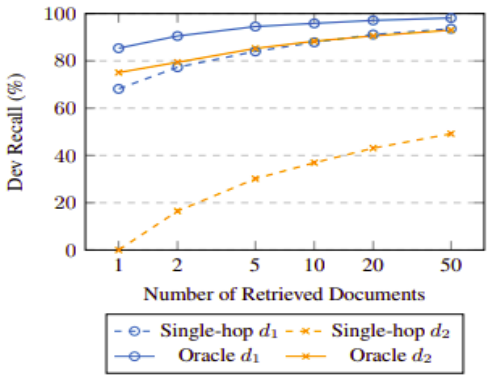


Figure 13: Single Hop vs Orcale query

question. These passages are passed to the sentence level module which filters out sentences which are irr-relevant to the question. Downstream QA and NLI modules use the upstream retrieval context to solve their respective tasks. For experimentation they used HotpotQA dataset and FEVER for QA and fact checking respectively. We implemented their approach only on the HotpotQA dataset

### 3.6 Approach Used

The MRS system is formulated as a function that maps an input tuple $(q, K)$ to an output tuple $(y, S)$ where q indicates the input query, $K$ is the textual $K_B$, y is the output prediction, and $S$ is selected supporting sentences from Wikipedia. Let E denote a set of necessary evidence or facts selected from $K$ for the prediction. For a QA task, $q$ is the input question and $y$ is the predicted answer. For a verification task, $q$ is the input claim and $y$ is the predicted truthfulness of the input claim. For all tasks, $K$ is Wikipedia.

- **Term Based Retrieval** A combination of TF-IDF and entity matching is used to narrow down the search space to relevant passages. This step ensures that the downstream modules have enough context to carry out their respective operations.
- **Document based Retrieval** This step basically checks the relevance of each paragraph with the question. A neural model is trained to provide a relevance score of each paragraph with the question. A threshold is used to filter out paragraphs whose relevance scores are very low.
- **Sentence based Retrieval** This step is the same as the Document based Retrieval step, the only difference is that it focuses on sentence level relevance. So a neural model is trained to compute the relevance score of each sentence in each paragraph with the question and a threshold is used to filter out sentences with low scores.
- **Semantic Retrieval Component** They used BERT for both Document based Retrieval and Sentence based Retrieval. The input representation and the loss function used for optimization are shown below. During training the gold documents plus some negative

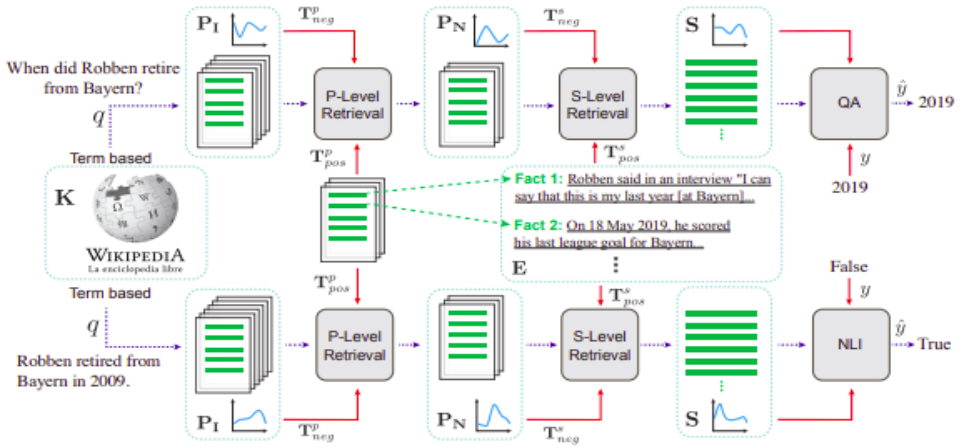


Figure 14: MRS Architecture

examples sampled from the retrieved wiki articles are used collectively for training each module. Below $T_{pos}$ is the positive set and $T_{neg}$ is the negative set and $p_i$ is the predicted probablility.

$$\textit{[CLS] Query [SEP] Context [SEP]}$$

$$J = \sum_{T_{pos}}^{n} \log p_i - \sum_{T_{neg}}^{n} \log 1 - p_i$$

– **Downstream Components** All the retrieval context from above is passed to the downstream QA component for multihop reasoning. The QA component used in the paper is BERT. The input representations for BERT are shown below and the parameters of the BERTA model were trained to maximize the log probabilities of the true start and end indexes (shown below). *yes*, *no* labels are added in the input layer to handle yes/no questions in HotpotQA.

$$\textit{[CLS] yes no Query [SEP] Context [SEP]}$$

$$J = \sum_{i}^{n} [\log y_i^s + \log y_i^e)$$

where $y_i^s$ and $y_i^e$ are the predicted probability on the ground-truth start and end position for the $ith$ example, respectively

It is worth noting that each sub-module in the system relies on its preceding sub-module to provide data both for training and inference. This means that there will be upstream data distribution misalignment if we trained the sub-module in isolation without considering the properties of its precedent upstream module. The problem is similar to the concept of internal covariate shift where the distribution of each layer's inputs changes inside a neural network. Therefore, it makes sense to study this issue in a joint MRS setting rather than a typical supervised learning setting where training and test data tend to be fixed and modules being isolated.

### 3.6.1 Ablation Studies

In this section we highlight the ablation and analysis performed by the original paper to show the effectiveness of each neural module. To show the importance of each neural module, each neural module is removed and the end to end system is trained again without that particular neural module. The results obtained are used for analyzing each neural module.

– **Removing of Paragraph level Retrieval** In this the paragraph based retrieval module is removed and the system is trained end to end and evaluated. The results are shown below. The results indicate that the negative effects on downstream modules induced

by the omission of paragraph-level retrieval can not be amended by the sentence-level retrieval module, and focusing semantic retrieval merely on improving the recall or the upper-bound of final score will risk jeopardizing the performance of the overall system.

– **Removal of Sentence based Retrieval** In this the sentence based retrieval module is removed and the system is trained end to end and evaluated. This results shows that rather than just enhance explainability for QA, the sentence-level retrieval module can also help pinpoint relevant information and reduce the noise in the evidence that might otherwise distract the downstream comprehension.

| Method | P-Level Retrieval | | | S-Level Retrieval | | | | Answer | | Joint | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Prec. | Rec. | F1 | EM | Prec. | Rec. | F1 | EM | F1 | EM | F1 |
| Whole Pip. | 35.17 | 87.93 | 50.25 | **39.86** | **75.60** | **71.15** | **71.54** | **46.50** | **58.81** | **26.60** | **49.16** |
| Pip. w/o p-level | 6.02 | 89.53 | 11.19 | 0.58 | 29.57 | 60.71 | 38.84 | 31.23 | 41.30 | 0.34 | 19.71 |
| Pip. w/o s-level | 35.17 | 87.92 | 50.25 | - | - | - | - | 44.77 | 56.71 | - | - |

Figure 15: Ablation Table

– **Sub Module Change Analysis**

1. Fix $h_p$ = 0 (the value achieving the best performance) and re-train all the downstream parameters and track their performance as $k_p$ (the number of selected paragraphs) changes from 1 to 12. The increase of $k_p$ means a potential higher coverage of the answer. When $k_p$ is changed from 1 to 2 the results are at its peak. This is consistent with the fact that at least two paragraphs are required to ask each question in HOTPOT. Then, after the peak, every score decreases as $k_p$ becomes larger except the recall of supporting fact which peaks when $k_p$ = 4. This indicates that the final QA module is more sensitive to upstream data and fails to maintain the overall system performance.

2. To study the effects of neural sentence level retrieval module towards downstream QA, we fixed ks to be 5 and set hs ranging from 0.1 to 0.9 with a 0.1 interval. More interestingly, the EM score for answer prediction peaks when hs = 0.2 and where the recall is higher than the precision. This misalignment between answer prediction performance and retrieval performance indicates that unlike the observation at paragraph-level, the downstream QA module is able to stand a certain amount of noise at sentence-level and benefit from a higher recall.

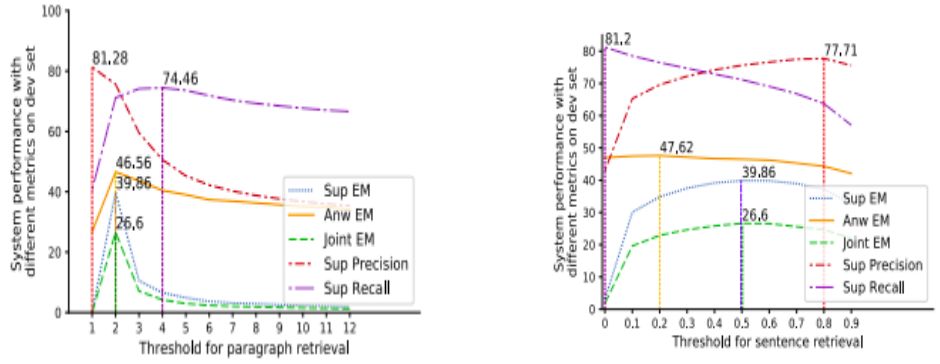


Figure 16: Ablation Study of MRS

# 4 Result & Analysis

This section highlights the results of all the models implemented above.

## 4.1 Results of Baseline

| settings | answer EM | answer F1 | sup fact EM | sup fact F1 | joint EM | joint F1 |
|---|---|---|---|---|---|---|
| distractor | 41.8 | 55.8 | 17.5 | 61.3 | 8.7 | 36.3 |
| full wiki | 16.0 | 24.0 | 1.78 | 21.9 | 0.87 | 7.6 |

Table 1: Baseline results

## 4.2 Results of DecompRC

| Models | All | Bridge | Single-hop |
|---|---|---|---|
| decomRC | 65.8 | 69.3 | 79.31 |
| BERT | 67.08 | 69.41 | 82.98 |
| BiDAF | 58.28 | 59.09 | - |

Table 2: F1 comparison across different models in Distractor settings

## 4.3 Results of PathNet

| PathNet | Accuracy |
|---|---|
| Wikihop | 65.4 |
| OpenBookQA | 52.0 |

Table 3: Accuracy of PathNet Model

## 4.4 Results of *GoldEN Retriever* & *MRS*

| Model | Ans EM | Ans F1 | Sup. EM | Sup. F1 | Joint. EM | Joint F1 |
|---|---|---|---|---|---|---|
| GoldenR | 36.20 | 47.6 | 30.0 | 63.4 | 18.00 | 38.1 |
| MRS | 46.00 | 58.00 | 38.00 | 69.2 | 25.0 | 46.00 |

Table 4: Comparison of GoldenRetriever & MRS

# 5 Limitations & Future Work

## 5.1 Decomposition & Rescorer

- some questions are not compositional but require implicit multihop reasoning, hence cannot be decomposed.
- there are questions that can be decomposed but the answer for each sub-question does not exist explicitly in the text, and must instead by inferred with commonsense reasoning.
- Lastly, the required reasoning is sometimes beyond our reasoning types (e.g. counting or calculation)

Addressing these remaining problems promising area for future work.Also simple elastic-search based model can help to gain increase F1 score as seen in golden retriever.

## 5.2 PathNet

This model works multiple choice question settings. Although, it provides an insight to the decision making but it has worked well only for this constrained setting and 2 hops. For higher hops, the model is likely to suffer. Further improvements could be including character encoding as well to take care of OOV words.

### 5.3 Golden Retriever

- Sometimes the model selects wrong entities for multi hop which degrades the over QA performance. It clearly signifies that a more powerful iterative read and retrieve is desirable.
- Because it used LCS and sub string matching methods, it is prone to fail if the same entity is refereed by a slightly different name in wiki articles.
- Through iterative reasoning and retrieval, GOLDEN Retriever greatly improves the recall of gold supporting facts, thus providing the question answering model a much better set of context documents to produce an answer from, and demonstrates competitive performance to the state of the art.

### 5.4 MRS

- As every module depends upon two upstream retrieval modules, so changing the threshold in each module results in high variation in output.
- Authors proposed a simple yet effective hierarchical pipeline system that achieves state-of-the-art results on two MRS tasks. Ablation studies demonstrate the importance of semantic retrieval at both paragraph and sentence levels in the MRS system.

## 6 Acknowledgments

We would like to thank **Dr. Manish Srivastava** for his constant guidance, support and motivation which helped us in completing our project successfully. Also, we would like to thank the Teaching Assistants **Hiranmai** and **Sumukh** who have helped us in every possible aspect.

## 7 Git Repository

Our code is available at: https://github.com/tushar117/multihopQA

## References

[1] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing(EMNLP).

[2] Alexander H Miller, Will Feng, Adam Fisch, Jiasen Lu, Dhruv Batra, Antoine Bordes, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform.

[3] Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension. In Proceedings of the 55th Annual Meeting of the Association of Computational Linguistics.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate

[5] Attention Is All You Need,Ashish Vaswani et.al

[6] Bi-Directional Attention Flow for Machine Comprehension (Minjoon Seo et. al, 2017)

[7] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen,Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In EMNLP.

[8] Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. Transactions of the Association of Computational Linguistics.

[9] WikiQA: A Challenge Dataset for Open-Domain Question Answering Yi Yang, Wen-tau Yih, Christopher Meek

[10] Sewon Min, Victor Zhong, Luke Zettlemoyer, Hannaneh Hajishirzi. Multi-hop Reading Comprehension through Question Decomposition and Rescoring. Published as a conference paper at ACL 2019 (long)

[11] Answering Complex Open-domain Questions Through Iterative Query Generation,PQi, Peng and Lin, Xiaowen and Mehr, Leo and Wang, Zijian and Manning, Christopher D., Oct 2019

[12] Exploiting Explicit Paths for Multi-hop Reading Comprehension Souvik Kundu and Tushar Khot and Ashish Sabharwal and Peter Clark

[13] Constructing Datasets for Multi-hop Reading Comprehension Across Documents Johannes Welbl1 Pontus Stenetorp1 Sebastian Riedel

[14] Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering Todor Mihaylov, Peter Clark, Tushar Khot, Ashish Sabharwal

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In NAACL.