The Chinese University of Hong Kong

Final Year Project Report (First Term)

Build Your Own Chatbot

Author:
CHAU Tak Ho
WANG Weixiao

Supervisor:
Prof. LYU Rung Tsong Michael

LYU2203
Faculty of Engineering
Department of Computer Science and Engineering
November 30, 2022

Abstract

Chatbot is a computer system that allows humans to interact with computers using Natural Human Language. There are many types of chatbots and can be simply categorized as open-domain or closed-domain.[1] Our purpose is to build a closed-domain chatbot that can discuss a certain topic with the user, for example, explaining an article or talking about a book, with some background knowledge in this domain, like an expert or experienced people in this field. Since conversation is a complicated task and involves many different techniques, during the first term, we focus on making a system that can understand questions in natural language, and generate appropriate responses. That is, building a close-domain question-answering (QA) system, that generates answers from a specific domain. After that, we will make it able to have more complicated conversations. We also tried different approaches to achieve it. Finally, we build a model that carries out this task by dividing it into 2 tasks: Question decomposition, which decomposes the original question into simpler sub-questions that can be easily answered, and Question answering, which finds the answer to a simple question. It will also be trained to have some "basic knowledge" in the corresponding field, so it could have better performance in answering the questions in the field. Our evaluations of this model show that the model has high accuracy on many complicated questions, and the basic knowledge also helps improve its performs. However, some drawbacks of the current system are found, and the analysis of how they appear and how to avoid them will also be discussed.

Acknowledgment

We would like to express our gratitude to our supervisor Professor LYU Rung Tsong Michael and our advisor Mr. Wenxuan WANG for guiding us through the final year project as well as giving us valuable suggestions.

Contents

1 Introduction

Chatbots have a significant impact on modern life. There are many types of chatbot systems that handle different tasks. Our project focus on building a chatbot that can have a discussion with the user about a certain topic. In practice, it chats with the user about the information of a/some given document(s). The QA system will answer your questions according to some given document in the same field, for example, a chapter in a novel, an introduction to a service, or a set of academic papers that research similar topics.

The advantage of focusing on a certain domain is that the responses it generates will have higher accuracy and be easily controlled. The model concentrates on answering questions on a specific domain and thus can have knowledge and algorithms that open-domain QA model cannot have. To achieve this, we understand the terms, abbreviations, or other proper nouns in this domain, together with synonyms, basic facts, and other basic knowledge, to improve its performance in the domain. Experimental results also show the model can make use of those "professional knowledge", and general models could not do that obviously.

The QA model of our system is based on roberta-base-squad2[2], it is an extractive QA model, that could extract the answer from given documents. We also use a question decomposition model[3] to make it capable of answering some indirect questions. The detail of those models is discussed in the Preliminary section.

We also tested the behavior of the model and showed its correctness on different evaluating metrics. It performs well with high correctness in decomposing questions and finding answers for most of the questions.

However, current system is not faultless. Some incorrect outputs are found during our experiment and led to the drawback of the model. We analyze them and make some improvement plans for our future work.

1.1 Overview

The main goal of the final year project is to build a chatbot. To be more specific, the chatbot built should be able to converse over a certain domain, i.e. a closed-domain chatbot. As one of the biggest strengths of a closed-domain chatbot is that it could be able to answer questions that are related to a certain topic, we focus on building a question-answering (QA) system of the chatbot first, and the work done in the first semester will be elaborated through this report. A brief introduction about the topic, background information, and objectives will be provided in this section.

1.2 Background

Chatbots are conversational computer programs that aim at simulating natural human conversation[4]. A chatbot can process user input, usually a natural language sentence, and produce relevant output.

Chatbots are currently used in many different domains and applications. Closed-domain chatbots are one of the mainstream applications. Closed-domain chatbots are chatbots that talk about a specific topic or field[5]. For example, chatbots that provide customer service are usually closed-domain, as they will mainly respond to frequently asked questions. The advantages of closed-domain chatbots include accuracy due to the limited knowledge they need to know about a certain topic, and efficiency as they can search for answers through a smaller amount of data.

Question answering (QA) of closed-domain chatbots, which is called closed-domain QA, is one of the main tasks of closed-domain chatbots. In closed-domain QA, user input will be a question related to a certain domain and output will be the answer to the question. There are many types of QA models that can achieve the task of closed-domain QA[6]. Two types of QA models are explored in this project, namely Knowledge Base Question Answering (KBQA) and extractive QA.(Detailed descriptions of these two types of models will be discussed in section 2.) Both models can perform well in closed-domain QA. In this project, KBQA is first explored. KBQA models will perform QA with a given knowledge base. The knowledge base contains knowledge of a certain domain and the KBQA model will search through the knowledge base according to the question to find out the answer. Extractive QA model is also explored. Instead of a knowledge base, Extractive QA model will use input documents as its background knowledge. Questions will be answered by searching through the documents.

1.3 Objective

Our purpose at this stage is to build a QA system that could answer questions based on a/some given document(s). It can handle complicated question, and have some background knowledge to help answering questions within the domain, like discriminate special expression like terms or synonyms in this field.

At first, we tried to use Knowledge Base Question Answering (KBQA) to achieve our purpose, but it could not meet our expectation. The reasons will be discussed at the begining of Methodology section.

Then, we planned to use extractive QA instead, as it does well on closed-domain QA tasks. And use a question decomposition model to pre-process the questions. The extractive QA model is trained on data of basic knowledge of a certain domain. The detail of model structure and the training detail are in later sections.

2 Preliminary

2.1 Natural Language Processing

Natural language processing (NLP), also known as computational linguistics, is a branch of computer science that focuses on utilizing computational methods to learn, comprehend, and create information in human languages.[7] The objectives of computational language systems can be varied. The objective may be to facilitate human-human communication, as in machine translation, facilitate human-machine interaction, as in conversational agents, or facilitate both human and machine profit from the massive amount of human linguistic information that is already available online. Question-Answering is one of the objectives of NLP, which is also the focus of this project and will later be introduced.

2.2 Knowledge Graph

Human knowledge is always in the form of (subject, predicate, object) or (head, relation, tail). Thus, knowledge graphs are developed to formally represent human knowledge. A knowledge graph is an organized representation of facts that contain entities, relationships, and semantic descriptions.[8] Entities can be real-world objects or abstract concepts, such as a car or a car brand. Relationships represent the relation between entities. Semantic descriptions of entities, and the corresponding relationships are types and properties with well-defined definitions. A knowledge graph is a directed graph with nodes as either subjects or objects and edges from subject to object as relation.

In mathematical notations, a knowledge graph is denoted as $G = \{\xi, \mathcal{R}, \mathcal{F}\}$, where ξ, $\mathcal{R}$ and $\mathcal{F}$ are sets of entities, relations and facts respectively. A fact is denoted as a triple $(h, r, t) \in \mathcal{F}$. Knowledge graph also involves two definitions:

- **Definition 1** A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge.[9]

- **Definition 2** A knowledge graph is a multi-relational graph composed of entities and relations which are regarded as nodes and different types of edges, respectively.[10]

2.3 Question Answering

Question Answering (QA) gives out a human-language question and gives a proper answer for it. Sometimes a specific question is asked and sometimes an open-ended question is asked. Answering goes with building defined systems in which it answers the question that is posted by humans in a natural language format[11]. There are different types of QA systems, and we only focus on closed-domain QA in this project.

Closed-domain QA deals with a particular domain or topic. It is an easier task of asking questions and getting answers because NLP systems are very good at finding a specific topic and retrieving the corresponding answers.[11] Closed-domain QA system gives more accurate answers than the open-domain QA system according to the literature survey of Question Answering Systems.[12]

2.4 Neural Relation Extraction

Neural relation extraction(NRE) discovers semantic relations between entities from unstructured text using deep learning methods. [13] It's a useful method to represent the information from original text in a form that well-structured and could be easy processed by machines. The extracted information represent as a triple (h, r, t), in which h denotes the head entity, t denotes the tail entity, and r denotes the relation between two entities.

Then the triples will form a knowledge graph that each triple become a edge of it.

Knowledge graphs such as FreeBase and DBpedia are examples of such representations. They are directed and labeled graphstructured data which aim to express such explicit semantics and relations of entities in triple form.[13]

REBEL (Relation Extraction By End-to-end Language generation) is a well performed NRE model. They use an approach that express triplets as a sequence of tokens. In detail, they introduce a set of new tokens as markers, like ¡triplet¿ ¡subj¿ and ¡obj¿, marks s the start of a new triplet, end of the head entity and the start of the tail entity surface form, and the end of the tail entity, individually.[14] They extract the triplets in the form of original text, like the example in Figure 1.

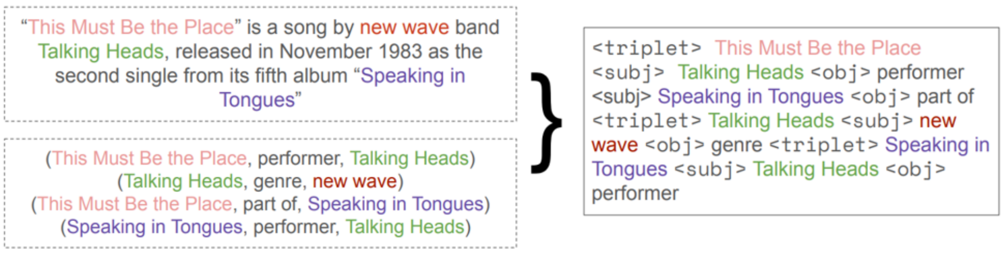

"This Must Be the Place" is a song by new wave band Talking Heads, released in November 1983 as the second single from its fifth album "Speaking in Tongues"
(This Must Be the Place, performer, Talking Heads)
(Talking Heads, genre, new wave)
(This Must Be the Place, part of, Speaking in Tongues)
(Speaking in Tongues, performer, Talking Heads)
<triplet> This Must Be the Place <subj> Talking Heads <obj> performer <subj> Speaking in Tongues <obj> part of <triplet> Talking Heads <subj> new wave <obj> genre <triplet> Speaking in Tongues <subj> Talking Heads <obj> performer

Figure 1: The overall structure of two mainstream methods of complex KBQA[15]

We run the model on a set of documents that scratched from wikipedia pages, and build a set of graphs form the extracted triples. But as the graphs are in the format of original texts, and need to transform into the format of wide-used knowledge bases like Freebase[16], DBPedia[17], or Wikidata[18] to let most of Knowledge Base Question Answering to run on it.

2.5 Knowledge Base Question Answering

Knowledge base (KB) is an organized database with a collection of triple-formatted facts (i.e. subject, relation, object)[15]. It is basically the same as knowledge graph except it involves formal semantics. Its purpose is to support the modeling of entity relationships. Many large-scale KBs, such as Freebase[16], DBPedia[17], Wikidata[18], have been built to support numerous downstream tasks. One of them is knowledge base question answering (KBQA), a task that uses KBs as its knowledge source to respond to inquiries in natural language.

The task of KBQA is described formally as follows:

A KB is denoted as $G = \{< e, r, e' > | e, e' \in \xi, r \in R\}$, where $< e, r, e' >$ denotes the relation r between subject e and object e', ξ and R denote the entity set and relation set respectively. Given KB G, the goal of KBQA is to answer natural language question $q = \{w_1, w_2, ..., w_m\}$ in the format of a sequence of tokens W (usually structured with a distinct vocabulary set V) and the predicted answers are denoted as A^*. It is typically assumed that the correct answer A^* can be derived from the entity set ξ of the KB or a natural language sequence. In general, a KBQA model is trained using a dataset $D = \{(q, A^*)\}$.

Simple KBQA and complex KBQA are the two categories of KBQA tasks[15]. Simple KBQA focuses on answering a simple question that involves only a single fact. Complex KBQA focuses on answering complex questions that contain multi-hop reasoning, constrained relations, or numerical operations. Semantic Parsing-based Methods (SP-based methods) and Information Retrieval-based Methods (IR-based methods) are two mainstream ways to achieve the complex KBQA task. The overall structures of the two methods are illustrated in Figure 2. Similarly, both methods make use of topic entity detection to perform the following jobs. The main difference between the two methods is that SP-based methods will transform the question into an executable logic form (i.e. SPARQL) while IR-based methods will retrieve a sub-graph that contains the topic entity, entities that are related to the topic entity and the corresponding relations and perform reasoning over the sub-graph.

RNG-KBQA[19] which uses Freebase[16] as its KB and GrailQA[20] as its training dataset is an SP-based model used for achieving the goal of this project. The overall structure is illustrated in Figure 3. First, for every entity detected in the

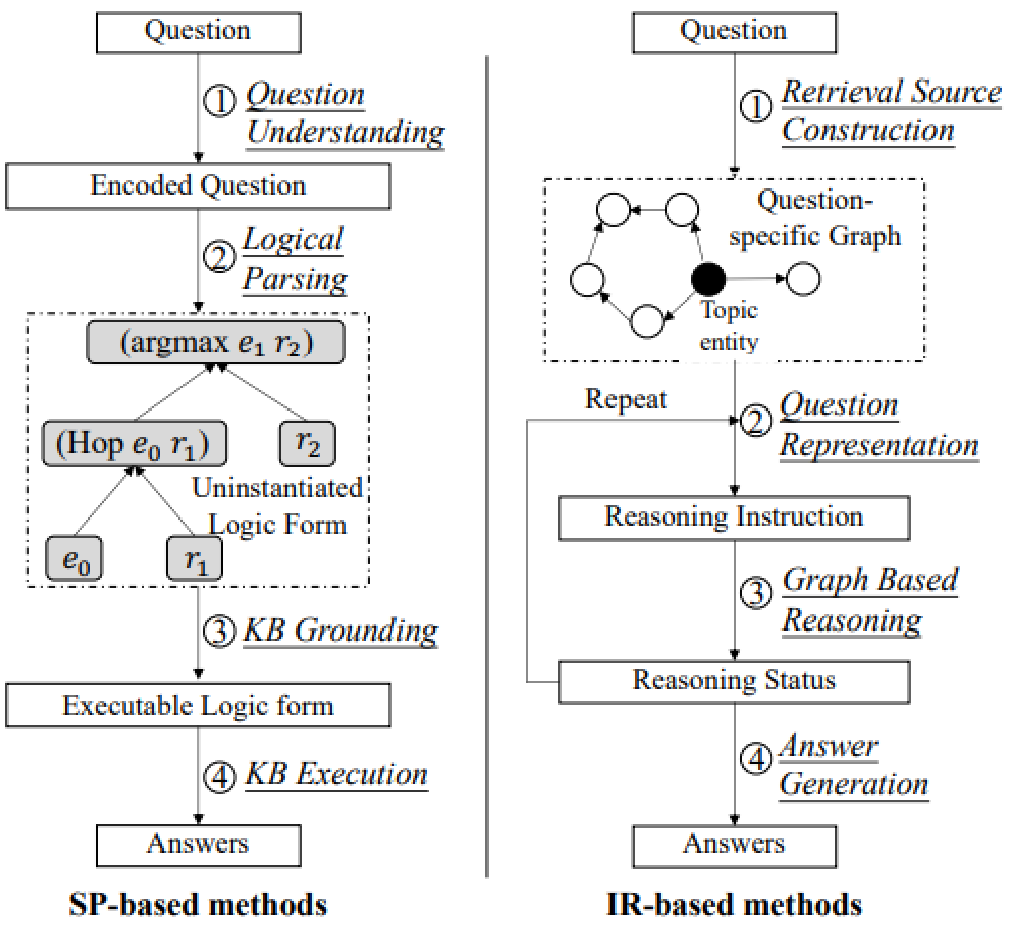
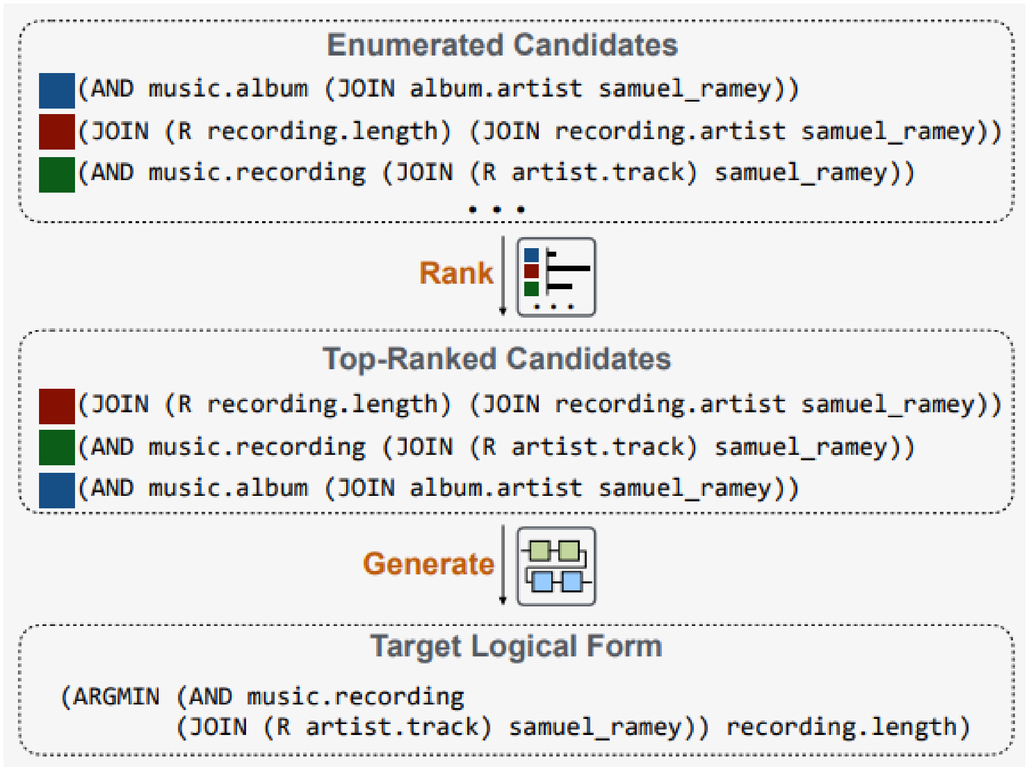


Figure 2: The overall structure of two mainstream methods of complex KBQA[15]

question, the query of the knowledge base for paths reachable within two hops will be enumerated as possible candidates. The candidates will be in logical form. Then, a ranker module will rank the candidates by using a BERT-based encoder to check the similarity between the question and the query. In the generation module, the top-k candidates will be used to generate a target logical form using a transformer-based seq-to-seq model[21] instantiated from T5[22].

The reason for choosing RNG-KBQA is that it achieves relatively good performance among other models that use GrailQA as the training dataset. GrailQA is designed to test three levels of generalization in KBQA: i.i.d. (i.e. independent and identically distributed), compositional, and zero-shot. Using the model trained based on this dataset could make the QA system more generalizable.



Enumerated Candidates
(AND music.album (JOIN album.artist samuel_ramey))
(JOIN (R recording.length) (JOIN recording.artist samuel_ramey))
(AND music.recording (JOIN (R artist.track) samuel_ramey))
...
Rank
Top-Ranked Candidates
(JOIN (R recording.length) (JOIN recording.artist samuel_ramey))
(AND music.recording (JOIN (R artist.track) samuel_ramey))
(AND music.album (JOIN album.artist samuel_ramey))
Generate
Target Logical Form
(ARGMIN (AND music.recording (JOIN (R artist.track) samuel_ramey)) recording.length)

Figure 3: The overall structure of RNG-KBQA[19]

2.6 Extractive QA

Given a document and a question regarding the content of the document, extractive QA is a task that aims at understanding the question and finding the answer (i.e. a segment of the document, typically an entity or an explanation) of the question[23, 24]. Currently, many of the extractive QA models use Pretrained Language Models (PLM), such as BERT[25], RoBERTa[26], to achieve the task.

The extractive QA model used in the project (i.e. roberta-base-squad2[2]) is a RoBERTa-base model. RoBERTa is a transformers model that was self-supervisedly pretrained on a sizable corpus of English data. The Masked Language Modeling (MLM) objective was used for its pretraining. When given a sentence, the model randomly selects 15% of the input words to be hidden, after which it must predict the words that were hidden. This makes it possible for the model to learn a bidirectional representation of the sentence.

Roberta-base-squad2 then uses RoBERTa to fine-tune on SQuAD2.0 dataset[27]. The Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset made up of questions that crowd workers have submitted about a selection of Wikipedia articles. Each question's answer is either a segment of text, or a span, from the corresponding reading passage, i.e. the context, or the question might be unanswerable. SQuAD2.0 is the newest version of the dataset which combines 100,000 questions in SQuAD1.1 (the older version) with over another 50,000 questions written by crowd workers. By using the SQuAD2.0 dataset, roberta-base-squad2 could be able to perform QA over a given document or text.

2.7 Question Decomposition

Multi-hop Reading Comprehension (RC) requires reasoning and aggregation across several paragraphs.[3] Some of the questions asked may be compositional, that is, combined with several sub-questions. In that case, the answer to those sub-questions may be from different paragraphs or documents that a single-hop QA method cannot correctly gather them.

The goal of question decomposition is to convert a multi-hop question into simpler, single-hop sub-questions.[3] Then, those questions can be answered by single-hop QA methods easily, and the final answer can be generated from them.

For DecompRC, it uses a token to replace the answer with a sub-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.[3] After the answer to a sub-question is found, it replaces the text back into the answer.

There are 3 types of decomposition, namely Bridging, Intersection and Comparison:

- Bridging is dividing the original question into a sequence of sub-questions, that the answer of the formal part will be inserted in the text of later parts.Used for the the question that could be solved by finding the first evidence in order to find later ones.[3]

- Intersection is extracting independent restrictions from the original question, and each restriction forms a sub-question. This is used to solve the questions that have multiple conditions by finding the answer that satisfies all of them.[3]

- Comparison is converting a comparison-like question into 2 searching task, i.e. find the property of 2 entity. Then, compare them to get the appropriate one.[3]

The examples are in Figure 4.

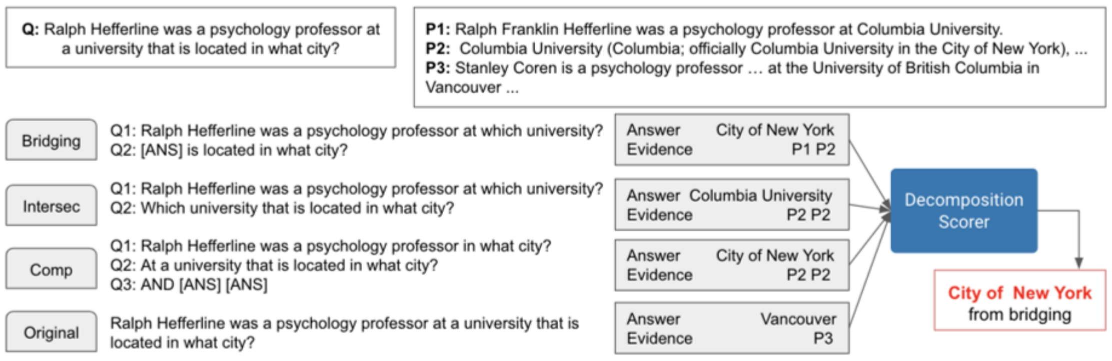

Q: Ralph Hefferline was a psychology professor at a university that is located in what city?
P1: Ralph Franklin Hefferline was a psychology professor at Columbia University.
P2: Columbia University (Columbia; officially Columbia University in the City of New York), ...
P3: Stanley Coren is a psychology professor ... at the University of British Columbia in Vancouver ...
Bridging
Q1: Ralph Hefferline was a psychology professor at which university?
Q2: [ANS] is located in what city?
Answer City of New York
Evidence P1 P2
Intersec
Q1: Ralph Hefferline was a psychology professor at which university?
Q2: Which university that is located in what city?
Answer Columbia University
Evidence P2 P2
Comp
Q1: Ralph Hefferline was a psychology professor in what city?
Q2: At a university that is located in what city?
Q3: AND [ANS] [ANS]
Answer City of New York
Evidence P2 P2
Original
Ralph Hefferline was a psychology professor at a university that is located in what city?
Answer Vancouver
Evidence P3
Decomposition Scorer
City of New York from bridging

Figure 4: Decomposition, source from[3]

Inside DecompRC, there is a model PointerC, which is a function that points to c indices $ind_1, ..., ind_c$ in an input sequence. Let S = $[s_1, ..., s_c]$denote a sequence of n words in the input sequence. The model encodes S using BERT[3, 28]

$U = BERT(S) \in R^{n\times h}$ where h is the output dimension of the encoder. Let $R^{n\times h}$ denote a trainable parameter matrix. There is a pointer score matrix $Y = softmax(UW) \in R^{n\times c}$[3].

Where Y denotes the probability that the i_{th} word is the j_{th} index produced by the pointer. Finally, it extracts c indices that yield the highest joint probability at inference.[3] (See Figure 5):

$$\mathrm{ind}_1, \ldots, \mathrm{ind}_c = \operatorname*{argmax}_{i_1 \leq \cdots \leq i_c} \prod_{j=1}^{c} \mathbb{P}(i_j = \mathrm{ind}_j)$$

Figure 5: calculate highest joint probability

3 Methodology

In this section, we discuss the architecture of our system and how we trained our models to achieve our goal.

We have different attempts for this purpose. The first approach is using a Neural Relation Extraction model and a KBQA model, that build a knowledge graph and answering quetsions based on it. But it failed to handle the given documents.

Then, we use another structure that do not modify the document, but decompose the questions to let make it easier to find the answer on the original text. And add the background information by fine-turn the models in advance.

3.1 First Approach

Our first approach is to run a Neural Relation Extraction model on the document and then build a knowledge graph from the triples extracted. And we can add extra edges on the graph as the background knowledge. After that, we can use a KBQA model to answer questions about them. The first structure of the model is as follows:

1. Build a knowledge graph that includes the background knowledge in a field.
2. Use a Neural Relation Extraction (NRE) model to extract triple from documents.
3. Use a transform function to build a knowledge graph,i.e. the graph of document form those triples.
4. Add the relations from background knowledge graph into the graph of document.
5. Run KBQA model on the graph of document.

However, this structure is not practical after experiments and further analysis. One reason is that the extracted triples are not always accurate as the data extraction pipeline of REBEL still keeps some noise, or excludes some relations that are entailed by the text[14]. Especially, the extracted model can not discriminate some proper nouns like terms that will not appear in general sentences. This means the extracted entities or relations may be incomplete themselves.

Moreover, changing the search domain of a KBQA model is inconvenient. Most high-performance KBQA models are based on public knowledge bases like Freebase or Wikidata, which have their own entity set and relation set, and all the triples in it are made with special syntax. For Freebase, the entities in it must be related to a specific Freebase ID, and there is no easy way to align textural entities to them, as the Freebase API is already closed. Also, the relations (called properties in Freebase) must form the logical structure like /automotive/engine/horsepower,[29] which makes it even more difficult to generate self-made relations. For Wikidata, it is easier, but all entities and relations also have to align to a specific ID, which means many entities that only appear in the specific context can not be discriminated against. So there may need many extra efforts will be a significant loss of information if we build our system in this way. So we give up this approach, and tried another way to realize the purpose, which will discussed in the Methodology section.

3.2 Model Architecture

Our QA system consists of a question decomposition model, a QA model, and a Ranking model. The overall diagram of the architecture can be found in Figure 6.

3.2.1 Question decomposition model

The question decomposition model is modified from the decomposition functions of DecompRC[30]. It read the question from the user, if it is a compositional question,

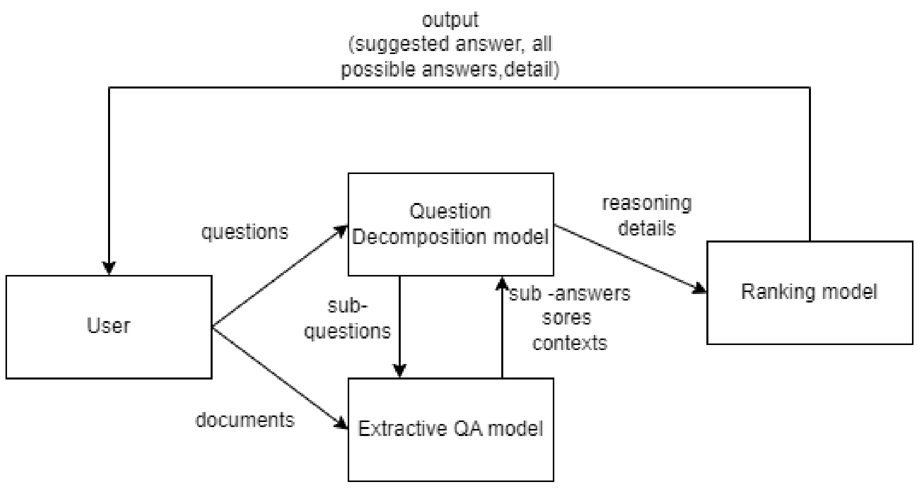


Figure 6: calculate highest joint probability

the model decomposes it into sub-questions. After the extractive QA model found the answer to each sub-question, the model will also aggregate them to get the final answers to the original question.

3.2.2 QA model

The QA model is fine-tuned from roberta-base-squad2[2]). It is an extractive QA model and it is trained on the basic knowledge of a specific field in advance.

The QA model receives the documents before the QA section, which are related to the field and will be the smaller domain of a conversation period (it is equal to a set of question-answering tasks at the current stage). Then, it will receive questions, and search for the answer to each question from the documents, and calculating their score.

The model can generate many possible answer at once, and it have a parameter called top_k to control the amount of answers it output each time. In practice, we set the default top_k = 5, which at most 5 answers with highest score will be kept each time. Then, it will send the answer together with their score and corresponding context, to the decomposition model.

3.2.3 Ranking model

The Ranking model is used to rank the final answers. As there may be more than 1 possible answer for each sub-question, when we decompose by bridging, it will influence the next sub-questions, then lead to different final answers. So we use a ranking function to rank and find the most possible final answer. It receive the final answers and their reasoning details, i.e. the score of all sub-answer in the intermediate steps, and ranks a final answer by the those scores. In practice, We simply calculate the score of the final answer as the product of sub-answers in the same answering sequence, i.e. $S_{final} = S_1^{p_1} \times ... \times S_n^{p_n}$. That S_i denotes the score of $i_t h$ sub-answer, and p_i is the parameter, we use p = 1 so that all sub answers have the same weight.

After that it will generate the suggested final answer, i.e. the most possible one, and other final answers will be treat as possible answers and out put together in case the suggested one is not correct, and for easily evaluation. The reasoning details are also stored, their are an option to also output them.

3.2.4 Running procedure

The whole system will receive documents from the user first. After that, it will receive questions from the user. Their is a flow chart of the overall procedure in Figure 8

For each question asked by the user, using bridging as an example, it will use the question decomposition model to generate single-hop sub-questions. Then the extractive QA model will find the answers to the sub-questions, together with the context and score. Then, those sub-answers will form different next sub-questions and each of which will send to the extractive QA model. It repeats the process until getting the final answers. Then, the answer ranking model will calculate the score of each final answer. After all, the system will output the most possible answer, and all other possible answers, and their reasoning details as an option. Their is a diagram of how the sub-questions and answers generated in Figure 7

The model could also output the processing detail for each of the final answers, e.g. the intermediate sub-answers, their scores, and corresponding contexts, for a better understanding of its working procedure, or for evaluation.

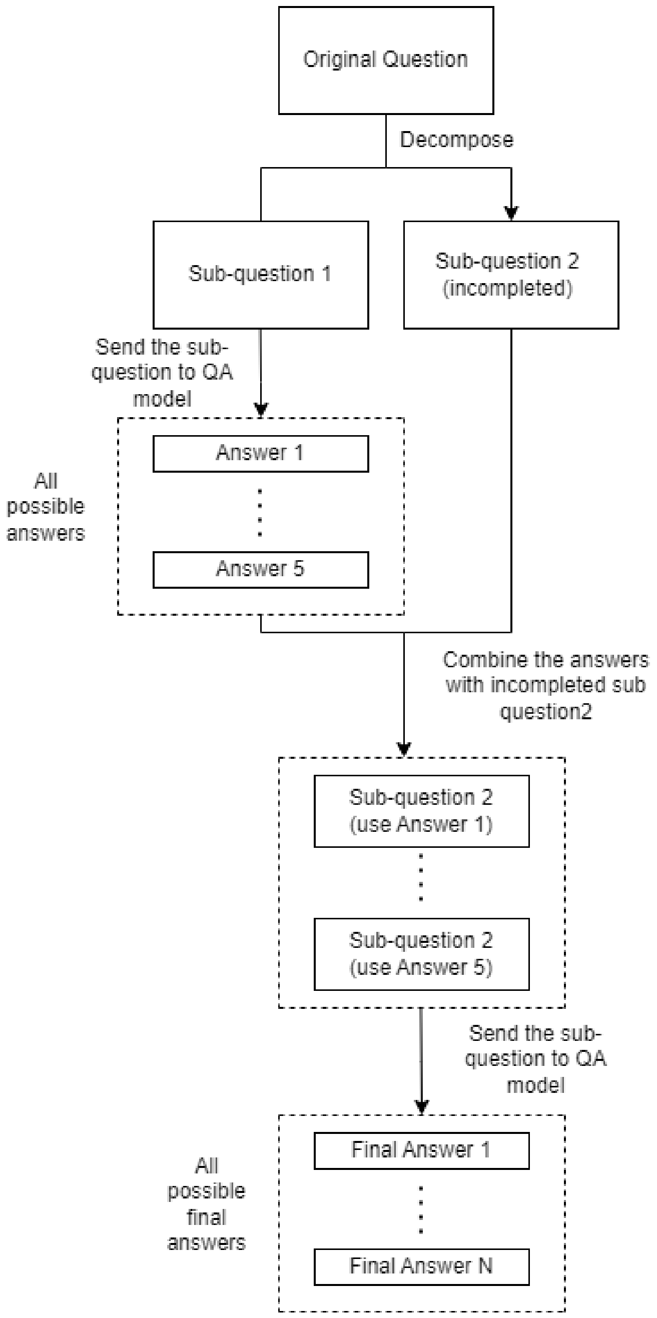

Original Question
Decompose
Sub-question 1
Sub-question 2 (incompleted)
Send the sub-question to QA model
All possible answers
Answer 1
Answer 5
Combine the answers with incompleted sub question2
Sub-question 2 (use Answer 1)
Sub-question 2 (use Answer 5)
Send the sub-question to QA model
All possible final answers
Final Answer 1
Final Answer N

Figure 7: The diagram of how the sub-questions and answers are generated

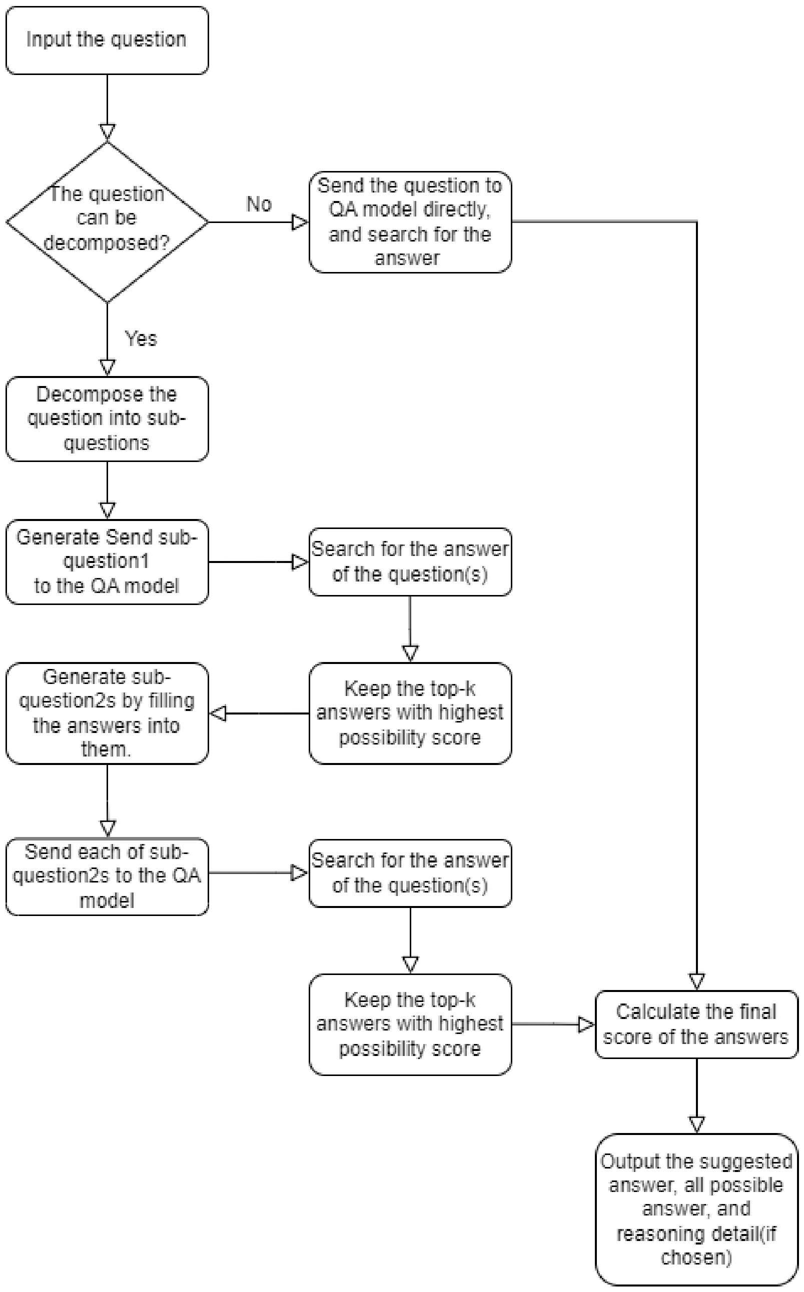
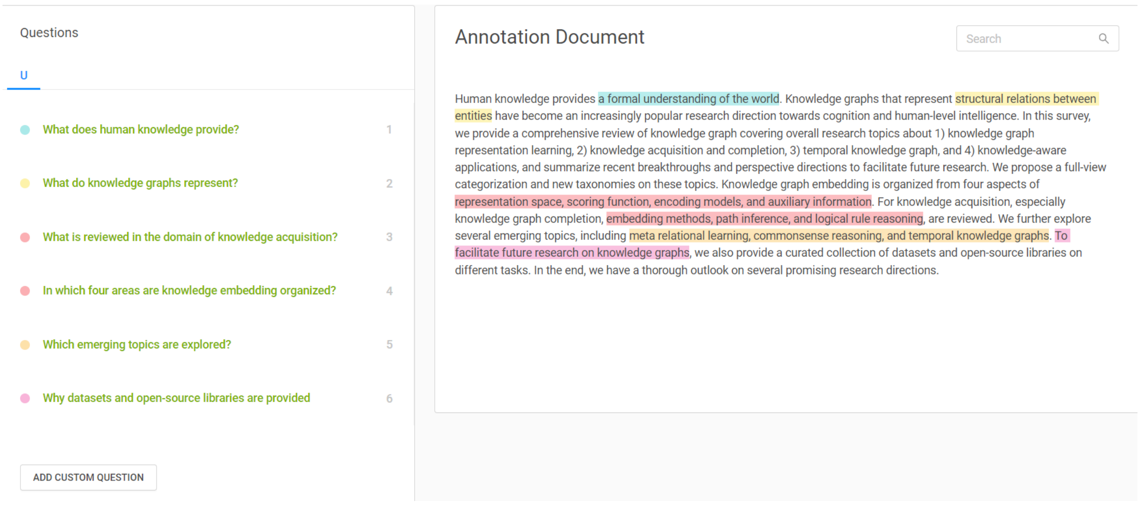

Input the question
The question can be decomposed?
No
Send the question to QA model directly, and search for the answer
Yes
Decompose the question into sub-questions
Generate Send sub-question1 to the QA model
Search for the answer of the question(s)
Keep the top-k answers with highest possibility score
Generate sub-question2s by filling the answers into them.
Send each of sub-question2s to the QA model
Search for the answer of the question(s)
Keep the top-k answers with highest possibility score
Calculate the final score of the answers
Output the suggested answer, all possible answer, and reasoning detail(if chosen)

Figure 8: The flow chart of the running procedure of the QA system.

3.3 Model Training

The extractive QA model (i.e. roberta-base-squad2[2]) can be fine-tuned using the toolkit provided by Haystack[31]. The fine-tuning dataset can be created using Haystack's annotation tool. Users can input a document into Haystack's annotation tool, construct custom questions regarding the document, then highlight a specific section of the document as the answer to the question. The annotated document and question can be output in SQuAD format.

The reason for fine-tuning the extractive QA model is to let the model have training data on a specific domain so that it may better understand some of the vocabularies that appear a lot in that domain. Thus, a survey of knowledge graph[8] is used as the document of the training dataset. An example of a custom question dataset is in Figure 9.



Questions

U

●	What does human knowledge provide?	1
●	What do knowledge graphs represent?	2
●	What is reviewed in the domain of knowledge acquisition?	3
●	In which four areas are knowledge embedding organized?	4
●	Which emerging topics are explored?	5
●	Why datasets and open-source libraries are provided	6

Annotation Document

Human knowledge provides a formal understanding of the world. Knowledge graphs that represent structural relations between entities have become an increasingly popular research direction towards cognition and human-level intelligence. In this survey, we provide a comprehensive review of knowledge graph covering overall research topics about 1) knowledge graph representation learning, 2) knowledge acquisition and completion, 3) temporal knowledge graph, and 4) knowledge-aware applications, and summarize recent breakthroughs and perspective directions to facilitate future research. We propose a full-view categorization and new taxonomies on these topics. Knowledge graph embedding is organized from four aspects of representation space, scoring function, encoding models, and auxiliary information. For knowledge acquisition, especially knowledge graph completion, embedding methods, path inference, and logical rule reasoning, are reviewed. We further explore several emerging topics, including meta relational learning, commonsense reasoning, and temporal knowledge graphs. To facilitate future research on knowledge graphs, we also provide a curated collection of datasets and open-source libraries on different tasks. In the end, we have a thorough outlook on several promising research directions.

Figure 9: An example of creating a custom question dataset using the annotation tool provided by Haystack[31]. The column on the left side is the custom questions created and the answers regarding each question are highlighted in the text under "Annotation Document".

There are a few key points to the custom question design process:

- The questions should contain words that are used especially in this domain and are uncommon outside this domain.

- The questions should be fact-seeking questions that can be answered by an entity or an explanation.
- The question should focus on the material in the provided document and should not require additional information.
- The questions should be a reformulation of the answer and should sound as natural as possible.

The toolkits provided by Haystack take the dataset created by the annotation tool as input and the original model (i.e. roberta-base-squad2) as the base model for fine-tuning. Then, the fine-tuning model will be used for testing and evaluation.

4 Experiment

4.1 Experimental Setup

In this section, we describe how we evaluate our model. We made several datasets based on information from different fields, they consist of a/some document(s) and a set of questions related to those documents, we test the system by asking those questions, and examining whether it can

- Decompose the sentence correctly.
- Find the correct context that contains the answer.
- Output the correct format of the answer, especially for those containing terms or special phrases.

We also test how the background knowledge helps improve the accuracy by testing the model on questions that need those information to get an answer, we run both the trained and untrained model on it, and compare their result, to ensure our QA system not pass the testing because of the help of background knowledge, instead of by coincidence.

4.2 Data

To evaluate our model, we manually build several evaluation datasets. Each of them includes some documents and a set of Questions that ask for information from the document. The documents are from a different domain for each evaluation phase.

To test its general performance, we use 2 manually made document.

One of the dataset is about bank service,the document is made from a list of services. As most of the services are independents, the questions are mostly simple questions. it evaluates our model's ability to find the correct context and output the correct and output the correct format of the answer, as many of its services have a similar name or similar diction. So it needs to discriminate the sentences about different services.

the other dataset is about NLP, whose document is extracted from Wikipedia and about the knowledge of NLP, to build another evaluation dataset. It includes many compositional questions, so we use it to evaluate the ability to decompose the compositional sentences.

The questions in a dataset are manually modified from the original sentences in the related document. We simply convert a sentence or some sentences that introduce something into a question that asks for some information from it. An example is in Table 1

The questions can be classified into simple questions, which can search for the answer directly, and compositional questions, which need to decompose and answer step by step.

To make a compositional question, we combine simple questions together. For example, if we make a compositional question by 2 simple questions, namely question1 and question2. The answer to question1 will be the component of another question2. Then, we combine the questions by replacing the replaceable words in question1 with a clause that deformed from question2. An example of making a compositional question from simple questions is in Table 2.

As the datasets are from different domains, we also use models that are trained on the basic knowledge of the corresponding domain. So that we can evaluate whether the training helps them discriminate the terms and proper nouns. There are 2 datasets used for this evaluation task, the information about them are in Table 3

Table 1: Example of generate question from original text

Context: Statistical methods in NLP research have been largely replaced by neural networks [32]	
Question: Many different classes what algorithms have been applied to natural-language-processing tasks	
Correct answer: machine-learning algorithms	
Incomplete answer: machine-learning **Cumbersome answer:** classed of machine-learning algorithm	

Table 2: Example of making a compositional multi-hop question from simple questions

Context 1: Statistical methods in NLP research have been largely replaced by neural networks[32] **Simple Question 1:** What replaced statistical methods in NLP research? **Answer 1:** neural networks
Context 2: In more practical terms neural networks are non-linear statistical data modeling or decision making tools[32] **Question 2:** Which kind of tools does neural networks belongs to? **Answer 2:** data modeling or decision making tools
Compositional Question: Which kind of tools does the method that replaced statistical methods in NLP belongs to? **sub question 1:** What replaced statistical methods in NLP research? **Answer 1:** neural networks **sub question 2:** Which kind of tools does [ANSWER] belongs to? **Answer 2:** data modeling or decision making tools **Final Answer:** data modeling or decision making tools

To test whether the background knowledge make effects, we build a dataset that also based on the document of NLP, but modifying the original sentences by deleted many description that helps understand it. And the diction of questions will also set to be different from the document. These modification makes it harder to answer by simply match the keywords.Their are 20 questions in this dataset, an example of modified text and questions is in Table 4.

Table 3: Data used to evaluate our QA system. There a 2 datasets build form texts about NLP and Bank Service for each. The length of document and the amount of 2 types of questions are shown in the table

Field	Document	Simple Question	Compositional Question
NLP	9426 words	20	28
Bank Service	4452 words	93	5

Table 4: An example of modified text, and question asked. as the untrained model do not know what is NLU, it just find a task of other application.

Original text: Natural language understanding (NLU) convert chunks of text into more formal representations such as first-order logic structures that are easier for computer programs to manipulate. [32]
Modified text: Natural language understanding convert chunks of text into more formal representations.
Question: What does NLU do?
Answer of model with background information: convert chunks of text into more formal representations.
Answer of untrained model: Natural-language generation

4.3 Evaluation

We then evaluated our model on the those datasets.

For decomposition, we test: Whether the decomposition is correct in syntax and whether the answer asked by sub-questions can be aggregated to the final answer.

Generating the question word-by-word is known to be a difficult task that requires substantial training data and is not straightforward to evaluate.[3] To be easier to perform the evaluation, we just test whether the sub-questions can be answered by the QA model and whether the sub-answers, if the model answer correctly, can lead to the final answer.

For finding context, we test: Whether the context that the QA model detect is including the correct answer. So We compare the context by the model and the ”correct” context, if the former one includes the latter one, or includes the parts that contain the answer, then it is correct, otherwise, it is wrong.

For the output of the answer, we test whether it is complete and in the correct format. i.e. intelligible and including all required descriptions. The examples of correct and incorrect answers are in Table5.

Table 5: Example of answers in correct and incorrect form

Context: Many different classes of machine-learning algorithm have been applied to natural-language-processing tasks[32]
Question: Many different classes what algorithms have been applied to natural-language-processing tasks
Correct answer: machine-learning algorithms
Incomplete answer: machine-learning **Cumbersome answer:** classed of machine-learning algorithm

4.4 Result

Table 6 shows the result of question decomposition and Question answering test, we can see that the model is able to find the correct answer to most of the questions. For those it outputs the incorrect answer, in the end, it also detects the correct answer, but it treats the correct answer as less possible than other answers. We also test some multi-hop questions that need the knowledge of terms and synonym, the models can also solve them.

Table 7 shows the result of background knowledge test, we can see that the untrained model passes some cases by coincidence, but the accuracy is far lower than the trained one.

Table 6: The result of running our trained models on the evaluation datasets. The accuracy in decomposition, finding context, and output answer are shown in the table

Dataset	Decomposition	Context	Answer
NLP	21 of 28	37 of 41	34 of 41
Bank Service	5 of 5	89 of 98	86 of 98

Table 7: The result of trained and untrained model on the background knowledge testing dataset, result shows that trained model behave well most of the cases, but untrained model can only pass few of them.

Model	**Correct context**	**Correct Answer**
trained	18 of 20	15 of 20
untrained	4 of 20	3 of 20

5 Discussion

5.1 Analysis

Our QA system is capable to accurately answer the questions within a specific domain, decompose compositional multi-hop questions, and make good uses of background knowledge to improve the performance.

The result on most of the questions during the experiment is good, but some drawbacks of the system are found. After further analysis of the format of some questions it fails to answer, it shows that the system only performs well under some conditions. That is:

1. The compositional questions must be intuitively compositional, i.e. can decompose into several sub-question from the original question text literally.
2. The answer to each sub-question must appear in the documents literally.
3. All the words in the answer text must appear in the context and must be successive. Both the decomposition model and extractive QA model cannot change the order of the appearance of words without manual annotation.

We test the model with several questions that violate this restriction, and the output proves our assumption: this system has some limitations and thus is not useful for all kinds of questions.

Then, we come up with some methods to avoid or reduce these kinds of errors. The methods are discussed in the improvement section and future work section.

5.2 Limitation

The limitations of the model are mainly from the limitations of the decomposition model, and the limitations of extractive QA. Some limitations are inherited from

DecompRC like some questions are not compositional but require implicit multihop reasoning, hence cannot be decomposed. Second, there are questions that can be decomposed but the answer for each sub-question does not exist explicitly in the text, and must instead be inferred with commonsense reasoning.[3] The cases are shown in Table 8 and Table 9.

For extractive QA, there are also some limitations in finding the context and getting the answer. When finding the context, we found that the current model can not recognize some tones, like negative tones, or some conditions, like time conditions. When Sentences have similar diction but differentiate in these conditions, the current model could not distinguish them and thus may fail to find the correct context. The cases that fail to find the context are shown in Table 10

Also, the answers must appear in the document literally, but some final answers can not be directly extracted from the test, and need to be restricted or represent form other words. The cases are shown in Table 11

Table 8: A typical failure case of our model, which is because the failure of decomposition. The required multi-hop reasoning is not compositional intuitively. The correct decomposition need some extra information.

Context1: Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so.[33] **Context2:** Decision trees is a earliest-used machine learning algorithm that produced systems of hard if-then rules similar to existing hand-written rules.[32]
Question: Decision trees build a model based on what?
subQ1: decision trees build which model? **subQ2:** [ANSWER]based on what?
Correct subQ1: What kind of algorithm does decision trees belongs to? **Correct subQ2:** [ANSWER] build a model based on what?
Final answer: sample data

Table 9: A typical failure case of our model, which is because the failure of decomposition. The required multi-hop reasoning is compositional intuitively, but the sub-question Q1 do not have answers, this lead to the failure of answering the original question. To answer this question correctly, the model need to treat it as a non-compositional question.

Context: Most higher-level NLP applications involve aspects that emulate intelligent behaviour and apparent comprehension of natural language.[32]
Question: higher-level NLP applications involve aspects that emulate what?
subQ1: higher-level NLP applications involve what aspect? **subQ2:** [ANSWER]emulate what?

Table 10: The extractive QA model can not discriminate the sentences when they have similar diction but different tones or conditions. In this case, the model ignore the words "before" and "after", and select the former context as it have more matching keywords

Question: How can I apply for a Credit Card in Hong Kong before leaving my home country?
Context found: After you leave your home country and move to Hong Kong, you can apply for a Credit Card by visiting one of our branches... **Correct context:** You can apply for a Credit Card even before you leave your home country. Fill out an application and provide all the necessary documents and we will review your application...

5.3 Improvement

After the analysis of the failure cases, there are some plans to improve the behavior of the system.

Firstly, we will make a threshold of output, that judge whether the suggested answer is appropriate, and thus prevents the model from outputting low-possibility answers. When all the answers' final score are lower the threshold, the question are treated as "no answers". The value of the threshold could be decided after further experiments.

After that, we could train the model to understand some implicit relations, like inclusion or opposition. So that, when it could not find an acceptable answer by original question text, it could change the subject or scope, to solve questions similar to the case in Table 8.

Table 11: The cases that questions can not answer by extracted QA model, although it is decomposed and have an answer.

Case1: The reasoning logic is implicit.
Context: Many words have more than one meaning; we have to select the meaning which makes the most sense in context.[32]
Question: What kind of words we need to select the meaning of them?
Answer: Many words **Correct answer:** words that have more than one meaning
Case2: Answers directly extracted from context are not complete or miss some restrictions
Context: Machine translation Automatically translate text from one human language to another.[32]
Question: Machine translation translate text into what?
Answer: another **Correct answer:** another human language

To solve similar cases like those shown in Table 9, it may need a structure to restore the intermediate abstract concept, that is not shown in the document literally, For example, "The aspect" in that case.

To improve the accuracy of finding the corresponding context and output the correct format of the answer. There need to be a model that detects the condition, restriction, or tone of sentences, to help match the corresponding context, and describe the entities accurately.

Also, the training on background information could be improved. Current training on includes discriminate proper nouns or expressions like abbreviations. We could train the model by understanding the variants of them, and relations between them. Also, current training still rely on manual work, it is time-costing, and make it hard to put our models on different fields, as they need to be trained on different basic knowledge for each field. The decomposition model should also be fine-turned to perform more accurately.

5.4 Future Work

Besides improving the QA system, we will also expand the scope of the application of our project. The current model is able to answer questions, but a complete chatbot needs more than just answering a question, it should also be able to reply to different kinds of sentences. For example, it should be able to do greetings, some simple chatting, or even discuss some topics. In a sense of that, we will add more methods to the current system in order to make the chatbot more complete.

Moreover, we are also exploring the practical usage of our system, like customer service, that is, there could be an AI customer service that introduces a company's services and answer customers' question based on this model, or educational usage. We will try hard to put the chatbot into usage in reality in the future.

6 Conclusion

We studied different kinds of QA methods, and build a QA system that could answer questions from a/some specific document(s), with some background knowledge that helps discriminate proper nouns and understand special relations in a certain domain in this term. Our first approach does not succeed but helps us better understand the structure of QA systems. Our final system, which consists of a question decomposition model, and an extractive QA model, successfully fulfills the requirements. It can answer complicated questions related to a specific domain with high accuracy and acceptable waiting time and is fine-tuned by tool kits from Haystack, to have some "background knowledge" of the domain. We also made several datasets to test it and evaluate its behavior. During the evaluation, it performs well on the majority of data, but it also revealed some limitations of the QA system: it is not appropriate for some question types. We further analyzed the failure cases and come up with some improvement plans to make it more reliable. After all, we made the plan for our future work, which is mainly about building a full chatbot that can have conversations with users based on given documents and could have some practical usage.

References

[1] A. S. Lokman and M. A. Ameedeen, “Modern chatbot systems: A technical review,” in *Proceedings of the Future Technologies Conference (FTC) 2018*, K. Arai, R. Bhatia, and S. Kapoor, Eds. Cham: Springer International Publishing, 2019, pp. 1012–1023.

[2] B. Chan, T. Soni, M. Pietsch, and T. Möller, “Roberta-base-squad2,” 2020. [Online]. Available: https://huggingface.co/deepset/roberta-base-squad2

[3] S. Min, V. Zhong, L. Zettlemoyer, and H. Hajishirzi, “Multi-hop reading comprehension through question decomposition and rescoring,” *arXiv preprint arXiv:1906.02916*, 2019.

[4] G. Caldarini, S. Jaf, and K. McGarry, “A literature survey of recent advances in chatbots,” *Information*, vol. 13, no. 1, p. 41, 2022.

[5] P. Suta, X. Lan, B. Wu, P. Mongkolnam, and J. Chan, “An overview of machine learning in chatbots,” *Int J Mech Engineer Robotics Res*, vol. 9, no. 4, pp. 502–510, 2020.

[6] S. Badugu and R. Manivannan, “A study on different closed domain question answering approaches,” *International Journal of Speech Technology*, vol. 23, pp. 315–325, 2020.

[7] J. Hirschberg and C. D. Manning, “Advances in natural language processing,” *Science*, vol. 349, no. 6245, pp. 261–266, 2015.

[8] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, “A survey on knowledge graphs: Representation, acquisition, and applications,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 494–514, 2021.

[9] L. Ehrlinger and W. Wöß, "Towards a definition of knowledge graphs." *SEMANTiCS (Posters, Demos, SuCCESS)*, vol. 48, no. 1-4, p. 2, 2016.

[10] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[11] A. Clementeena and P. Sripriya, "A literature survey on question answering system in natural language processing," *International Journal of Engineering and Technology(UAE)*, vol. 7, pp. 452–455, 06 2018.

[12] S. Lende and M. Raghuwanshi, "Closed domain question answering system using nlp techniques," *IJESRT*, 01 2016.

[13] M. Aydar, O. BOZAL, and F. ÖZBAY, "Neural relation extraction: a review," *TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES*, vol. 29, pp. 1029–1043, 03 2021.

[14] P.-L. Huguet Cabot and R. Navigli, "REBEL: Relation extraction by end-to-end language generation," in *Findings of the Association for Computational Linguistics: EMNLP 2021.* Punta Cana, Dominican Republic: Association for Computational Linguistics, November 2021, pp. 2370–2381. [Online]. Available: https://aclanthology.org/2021.findings-emnlp.204

[15] Y. Lan, G. He, J. Jiang, J. Jiang, W. X. Zhao, and J.-R. Wen, "Complex knowledge base question answering: A survey," *arXiv preprint arXiv:2108.06688*, 2021.

[16] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1247–1250.

[17] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer *et al.*, "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic web*, vol. 6, no. 2, pp. 167–195, 2015.

[18] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[19] X. Ye, S. Yavuz, K. Hashimoto, Y. Zhou, and C. Xiong, "Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering," *arXiv preprint arXiv:2109.08678*, 2021.

[20] Y. Gu, S. Kase, M. Vanni, B. Sadler, P. Liang, X. Yan, and Y. Su, "Beyond iid: three levels of generalization for question answering on knowledge bases," in *Proceedings of the Web Conference 2021*, 2021, pp. 3477–3488.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[22] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer." *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.

[23] M. Luo, K. Hashimoto, S. Yavuz, Z. Liu, C. Baral, and Y. Zhou, "Choose your qa model wisely: A systematic study of generative and extractive readers for question answering," *arXiv preprint arXiv:2203.07522*, 2022.

[24] A. Fisch, A. Talmor, R. Jia, M. Seo, E. Choi, and D. Chen, "MRQA 2019 shared task: Evaluating generalization in reading comprehension," in *Proceedings of the 2nd Workshop on Machine Reading for Question Answering.* Hong Kong, China: Association for Computational Linguistics, November 2019, pp. 1–13. [Online]. Available: https://aclanthology.org/D19-5801

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).* Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[26] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019. [Online]. Available: https://arxiv.org/abs/1907.11692

[27] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text," *arXiv e-prints*, p. arXiv:1606.05250, 2016.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).* Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[29] “Freebase api (deprecated),” 2022. [Online]. Available: https://developers.google.com/freebase

[30] S. Min, V. Zhong, L. Zettlemoyer, and H. Hajishirzi, “Multi-hop reading comprehension through question decomposition and rescoring,” in *ACL*, 2019. [Online]. Available: https://github.com/shmsw25/DecompRC

[31] M. Pietsch, T. Soni, B. Chan, T. Möller, and B. Kostić, “Haystack,” 2019. [Online]. Available: https://github.com/deepset-ai/haystack/

[32] Wikipedia contributors, “Natural language processing, wikipedia,” 2022, [Online; accessed 29-November-2022]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Natural_language_processing\&oldid=1122895538

[33] ——, “Machine learning, wikipedia,” 2022, [Online; accessed 29-November-2022]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Machine_learning\&oldid=1124340806