Conversational Question Answering Using a Shift of Context

Nadine Steinmetz
TU Ilmenau
Germany
nadine.steinmetz@tu-ilmenau.de

Bhavya Senthil-Kumar[*]
TU Ilmenau
Germany
bhavya.tharayil@gmail.com

Kai-Uwe Sattler
TU Ilmenau
Germany
kus@tu-ilmenau.de

ABSTRACT

Recent developments in conversational AI and Speech recognition have seen an explosion of conversational systems such as Google Assistant and Amazon Alexa which can perform a wide range of tasks such as providing weather information, making appointments etc. and can be accessed from smart phones or smart speakers. Chatbots are also widely used in industry for answering employee FAQs or for providing call center support. Question Answering over Linked Data (QALD) is a field that has been intensively researched in the recent years and QA systems have been successful in implementing a natural language interface to DBpedia. However, these systems expect users to phrase their questions completely in a single shot. Humans on the other hand tend to converse by asking a series of interconnected questions or follow-up questions in order to gather information about a topic. With this paper, we present a conversational speech interface for QA, where users can pose questions in both text and speech to query DBpedia entities and converse in form of a natural dialog by asking follow-up questions. We also contribute a benchmark for contextual question answering over Linked Data consisting of 50 conversations with 115 questions.

1 INTRODUCTION

With the increasing development of speech interfaces and chatbots, conversations with a machine becomes more and more common to us. In this way, we are able to ask for everyday life questions or communicate with a customer service without talking to a real person. But, most of these interfaces are either trained for a very specific domain or limited regarding the type of questions. Question Answering (QA) systems based on knowledge graphs aim to answer questions from the complete knowledge represented by the graph. Of course, these systems might be specific regarding domains, but as they are tuned to graph patterns, they also work on all-purpose knowledge bases, such as DBpedia or Wikidata. Over the last 10 years, the QALD challenge (Question Answering on Linked Data) provided several datasets for the purpose of the evaluation of QA systems. Participating systems showed the ability to understand natural language and transform it to a formal query language, specifically SPARQL, in order to provide the answer to the user. These systems require complete sentences respectively questions to be able to process it further. In contrast to that, conversations often start with a complete sentence/question and further questions are built upon the context of the preceding questions and answers. Follow-up questions might only consist of a fragment of a sentence and must be completed *in mind*. Which means, the conversation partner requires context information. For humans, this also works when the context is slightly changing to a different topic but having minor touch points to the previous course of conversation. With this paper, we present an approach that is able to handle slight shifts of context. Instead of maintaining within a certain node distance within the knowledge graph, we analyze each follow-up question for context shifts.

The remainder of the paper is organized as follows. In Sect. 2 we discuss related approaches from QA and conversational QA. The processing of questions and the resolving of context is described in Sect. 3. We have implemented this approach in a prototype which is evaluated using a newly created benchmark. The results of this evaluation presented in Sect. 4 demonstrate that our approach is capable of holding up several types of conversations. Finally, we conclude the results in Sect. 5 and point out to future work.

[*]work was done during master studies at the institute

2 RELATED WORK

Our proposed system is able to handle speech as incoming questions as well as textual input. But, the focus of our proposed approach is the correct interpretation of conversations within a context. Therefore, we limit the related work discussion to research approaches on conversational QA.

Dhingra et al. proposed KB-InfoBot, a fully neural end-to-end multi-turn dialogue agent for the movie-on-demand task [2]. The agent is trained entirely from user feedback. It consists of a belief tracker module which identifies user intents, extracts associated attributes and tracks the dialogue state in a Soft-KB Lookup component which acts as an interface with the movie database to query for relevant results. Subsequently, the state is summarized into a vector and a dialogue policy selects the next action based on the current dialogue state. The policy is entirely trained on dialogues. The authors proposed a probabilistic framework for computing the posterior distribution of the user target over a knowledge base, termed a Soft-KB lookup. This distribution is constructed from the agent's belief about the attributes of the entity being searched for. The dialogue policy network, which decides the next system action, receives as input this full distribution instead of a handful of retrieved results. They show in their experiments that this framework allows the agent to achieve a higher task success rate in fewer dialogue turns.

Microsoft XiaoIce ("Little Ice" in Chinese) is a social chatbot primarily designed to be an AI companion with which users can form long-term emotional connections [9]. This ability to establish long-term emotional connections with it's human users distinguishes XiaoIce from other social chatbots as well as popular conversational AI personal assistants such as Amazon Alexa, Apple Siri, Microsoft Cortana and Google Assistant. XiaoIce has attracted over 660 million active users since it's launch in May 2014 and has developed more than 230 skills ranging from question answering, recommending movies or restaurants to storytelling and comforting. The most sophisticated skill is Core Chat, where it engages in long and open-domain conversations with users.

In addition to having a sufficiently high IQ required to acquire a range of skills to complete specific tasks, social chatbots also require a high EQ to meet user's emotional needs such as affection and social belonging. The core of XiaoIce's system design is this integration of IQ (content-centric) and EQ(user-centric) to generate contextual and interpersonal responses to form long-term connections with users.

Just recently, Vakulenko et al. presented an approach to rewrite follow-up questions in conversations in order to be able to find the correct answer [8]. Their proposed model employs a unidirectional Transformer decoder. The model predicts output tokens according to an input sequence, references to a context and the context itself. The training is performed via a teacher forcing approach. After the rewriting the input question, the question answering process consists of the retrieval of relevant text passage, the extraction of potential answers to the input question and the ranking of the answers. The authors evaluated their approach on two different datasets (QuAC[1] and TREC CAsT[2]). In contrast to this approach, we are dealing with Semantic Question Answering and our procedure to identify the context of a conversational question is based on the underlying knowledge graph – specifically DBpedia.

Reddy et al. introduced CoQA, a dataset for Conversational Question Answering systems consisting of a total of 127,000 questions along with their answers from conversations about text passages covering seven unique domains [5]. The questions are sourced using Amazon Mechanical Turk (AMT) by pairing two annotators, a questioner and an answerer, on a passage through the ParlAI MTurk API [4]. Every turn in the conversation contains a question and an answer along with its rationale. It can be observed that around 30.5 percent of questions of the CoQA dataset does not rely on coreference with the conversation history and can be answered on their own. Almost half of the questions (49.7 percent) contains explicit coreference markers such as he, she, it etc. They either refer to an entity or an event introduced previously in the conversation. The remaining 19.8 percent does not contain any explicit coreference markers but rather refer to an entity or event implicitly.

All approaches and datasets described above are based on unstructured textual sources. In contrast, our approach is based on a knowledge graph and takes into account the semantic information of the question. Christmann et al. proposed an approach for a conversational question answering system based on knowledge graphs [1]. Their system, CONVEX, examines the context of an input question within the knowledge graph to be able to answer follow-up questions. Based on the edges and nodes around the focus of the first question, the conversation is disambiguated. The authors claim, that topic changes are rare and therefore only investigate the direct context of the focus entity of the first question. The evaluation benchmark has been sourced by Amazon Mechanical turks and consists of around 11,200 conversations based on the Wikidata knowledge graph. In contrast to CONVEX, our system is able to handle topic changes – for instance, when the focus entity changes, but the predicate stays the same.

[1]https://quac.ai/
[2]http://www.treccast.ai/

3 METHOD & IMPLEMENTATION

We introduce our system architecture as well as the processing steps during a conversation in detail in the next sections.

3.1 System Architecture

Figure 1 shows the proposed system architecture. The speech interface is developed using Dialogflow, Google's development suite for creating dialogue systems on websites, mobiles and IoT devices. Dialogflow is an AI-powered tool that uses machine learning models to detect the intents of a conversation. The underlying Google Speech To Text (STT) API is used to transcribe user audio input to text. The Text To Speech (TTS) API returns spoken text as an audio response back to the user. Once an intent is identified, Dialogflow sends a request to our webhook service with the transcribed text. The webhook service then handles the user's query and forwards the request to an external API, i.e. our HYDRA API[3] or the DBpedia SPARQL endpoint[4]. The result of the query is forwarded to the Dialogflow agent which finally responds to the user with a spoken audio response. Our webhook service is the crux of the system that handles user queries. It has a context module that handles follow-up queries by resolving co-references to entities using a dialog memory.

3.2 Processing Steps

The processing steps in answering a query and the detailed steps are explained in the following sections.

3.2.1 Intent Classification. In the first step, we identify the intent of the question. An intent represents the purpose of a user's input. We predefined several intents for each type of user request that we want to handle:

- greeting intent,
- factual intent - for first questions in conversations,
- follow-up intent - to continue a conversation,
- a fall back intent - if something is wrong with the question or it cannot be answered.

For the training of the intent classification, we used a set of different sample questions. We identify follow-up queries based on certain contextual cues such as the presence of explicit coreferences like "he", "she", "his", "her", "it's" etc. or the presence of phrases like "What about", "how about" etc. For each intent we can either respond with a static response as in the case of a greeting or define an action to be triggered. When a user writes or says something, Dialogflow categorizes and matches the intention to a predefined intent which is known as intent classification. For example a query such as "What is the capital of Cameroon?" would be matched to the factual query intent and a follow-up query such as "What about China?" or "What is its population?" would trigger the follow-up query intent.

3.2.2 Webhook Service and fulfillment. Once the intent of the user is identified and necessary parameters are extracted, the Dialogflow agent sends a webhook request with the transcribed raw text, the intent and parameters to the webhook service. Depending on the intent the appropriate function is triggered to handle the query. The end user writes or speaks in the query "What is the capital of Czech Republic?". Based on the identified intent, different actions are triggered:

[3]HYDRA is a QA system that transforms natural language to SPARQL queries for DBpedia based knowledge graphs, cf. [7]
[4]http://dbpedia.org/sparql

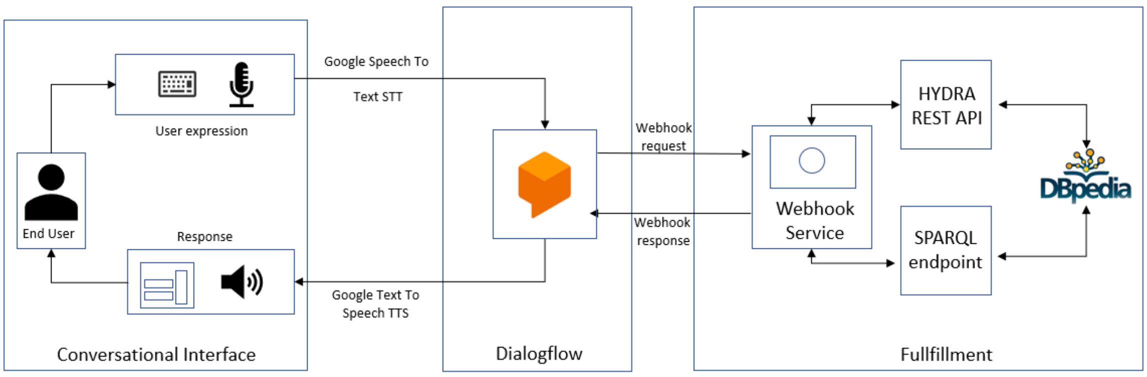


Figure 1: System architecture

- A factual question intent triggers a direct API call for our QA API *HYDRA*.
- In case of a follow-up intent, the question is further analyzed to retrieve the answer.

In the first case, the Dialogflow agent matches the end-user expression to the factual question intent. This triggers the direct call of the *HYDRA* REST interface with the query text. The *HYDRA* system generates the SPARQL using a template based approach (citation removed for anonymization). The *HYDRA* service provides the answer and the respective SPARQL query. The answer is transferred to Dialogflow and provided to the user. The SPARQL query is parsed to extract the subject and predicates and the information is stored in the output context of the payload. For every conversation turn, we store the subject and predicate values to the context memory in order to resolve them in subsequent follow-up queries.

For the case of a follow-up intent, the question is analyzed according to the context of the preceding conversation. The actual analysis and the processing steps are described in the following sections.

3.2.3 Context Detection in Follow-Up Questions. For the handling of a follow-up question, several steps are required. We identify named entities and detect the type of context shift. For our approach, a shift of context can occur in two different ways:

- Either the focus (named entity (NE)) of the previous question changes,
- or the follow-up question asks for a different relationship/predicate.

Consider the factual question *What is the official language in Germany?*. Here, the following types of questions are conceivable:

- Either the follow-up question asks for the official language of a different country,
- or the conversation continues with a question regarding a different fact about Germany.

Any other question would be considered to be a new factual question and the start of a new conversation. Therefore, we identify the shift of context and continue to generate the new SPARQL based on the context that maintains compared to the previous question.

Thus, our approach for context detection in follow-up questions consists of three sub-steps:

- named entity linking (NEL),
- predicate identification (PI), and
- context resolution (CR).

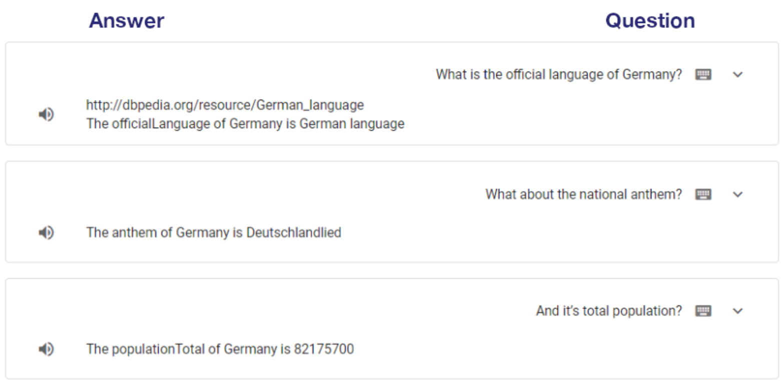
These three sub-steps are described in detail in the next paragraphs.

Named Entity Linking. Following the sample question above, a follow-up question could be "What about China?". Therefore, we have to identify the new NE in the follow-up question. For this, we utilize the DBpedia Spotlight web service[5] to annotate mentions of named entities in natural language. As the disambiguation of ambiguous mentions would require more contextual information, the service most probably returns the NE with the highest popularity.

Predicate Identification. In case, the follow-up question asks for a different fact about the same named entity, we need to identify the new predicate. For the mapping of natural language phrases to their corresponding DBpedia properties, the PATTY [3] dataset is used. The dataset contains the original labels and several alternative phrases for each DBpedia ontology property. For the identification of property phrases in the follow-up question, we use the Dialogflow entity service[6]. The service allows to create custom (so-called) entities to train the identification within natural language. Here, we create a custom entity using the DBpedia properties and the phrases from the PATTY dataset. The Dialogflow agent is trained on this custom entity and we can use this custom entity to recognize the phrases in the user utterance and map them to their corresponding DBpedia properties. For example the phrases “Who was he married to”, “Who is his spouse” etc. are both mapped to the DBpedia property `dbo:spouse` and stored as a parameter in the webhook request payload and can then be accessed in the webhook service handler.

Resolving context. When resolving the context of a follow-up question, the essential task is to identify the shift of context. This means, we need to analyze if the focus entity or the property changed. Consider the example query as shown in Figure 2. Here, the subject is “Germany” and “official language” refers to the property. Now consider the follow-up query “What about China?” is being asked. After the NEL and PI processing steps, the named entity for the phrase “China” is identified so we know that this is the new focus the user is referring to, but no properties are identified. Therefore, we need to resolve this from the predicate mentioned in the previous turn in the conversation. We fetch the property `dbo:language` from the dialog memory and reformulate

[5]https://www.dbpedia-spotlight.org/api
[6]https://cloud.google.com/dialogflow/es/docs/entities-custom



What is the official language of Germany?

http://dbpedia.org/resource/German_language
The officialLanguage of Germany is German language

What about the national anthem?

The anthem of Germany is Deutschlandlied

And it's total population?

The populationTotal of Germany is 82175700

Figure 2: Sample conversation for a change of the property.

the same query by replacing the subject with the new subject `dbr:China`[7]. Now consider the case if the follow-up question would have been "What about its national anthem?". Here, the new property for the phrase “national anthem” is identified, but it is required to keep the focus entity of the previous question. We reformulate the query with the new property `dbo:anthem` and subject `dbr:Germany`.

In this manner, at each turn of the conversation we need to identify the presence of new entities or properties and resolve the missing part from the dialog memory. However, the follow-up question could also refer to the answer of the previous question instead of the focus entity. In order to resolve the coreferences correctly, we rely on some rules based on the question type of the first question and the type of follow-up question as follows:

- **WHO** : If the first question is of type *WHO*, we know that the expected answer type is a *Person*. Therefore if we have a *WHO* question followed by a coreference "he/she" as in *When was she born* , we resolve the subject to the answer of the first question. If the same question is followed by a coreference "it" such as "What is its capital?", we resolve the coreference to the first subject "Germany".
- **WHAT / WHICH** : If the first question is of type *WHAT* or *WHICH*, the expected answer can be a city, currency, number etc. For example, "What is the capital of Czech Republic?". Now if this is followed by a question such as "what about its currency?" the coreference "it" is resolved to the subject of the first question which is "Czech Republic".
- **WHEN** : For the question type *When*, the expected answer is a date or a location, for example "When was Princess Diana born?" or "Where was Bach born?”. So, in the case of a follow-up question with a coreference "she" as in "And when did she die?" shall always be resolved to the subject identified in the first question "Princess Diana".
- **WHERE** : Here the expected answer is a location, for example "Where was Bach born?" and similar to the case of *WHEN* questions coreferences in subsequent follow-up questions are always resolved to the subject of the first question, here in this example "Bach".
- **HOW** : The expected answer type of the question type *How* is a number. Hence, the coreferences in the follow-up queries are resolved to the subject in the first question. For example the question "How many languages are spoken in Turkmenistan?" followed by the coreference "it" in the follow up question "What is it’s official language?" will be resolved to "Turkmenistan".

These rules are based on our inferences from the questions in our evaluation benchmark consisting of 50 sets of conversations each of which referring to a question in the QALD 9 evaluation dataset (cf. Section 4.1).

Asking for Clarifications. There are some cases where the follow-up question may be referring to the answer to the first question rather than the subject in the first question. For example consider the question “Who is the spouse of Barack Obama?” followed by the follow-up question “And at which school did she study?”. In this case, the focus of the first question as well as the answer are of the same type. We need to resolve the ambiguity by asking the user for clarification. For our first prototype, we have implemented this for the entities of type `dbo:Person`. That means, whenever the focus of the question as well as the answer to the question both refer to named entities of type `dbo:Person`, we resolve it by asking the user which entity they actually meant. We have implemented this using a suggestions list where users are presented a list of options containing the two entities mentioned and users can confirm by selecting the appropriate entity.[8] Figure 4 shows the conversation which tries to answer the QALD question “What is the name of the school where Obama’s wife studied?”. The focus of the first query is "Barack Obama" and the system returns the spouse of Barack Obama i. e, “Michelle Obama” as the answer. Both entities are saved in the dialog memory for future reference. When the user asks the follow-up query "Which school did she/he study?" the system asks the user for clarification in order to resolve the ambiguity between the two entities[9]. The user is now presented with two options as a suggestion list and the user confirms by clicking on one of them. Finally, the response for the selected query is presented to the user.

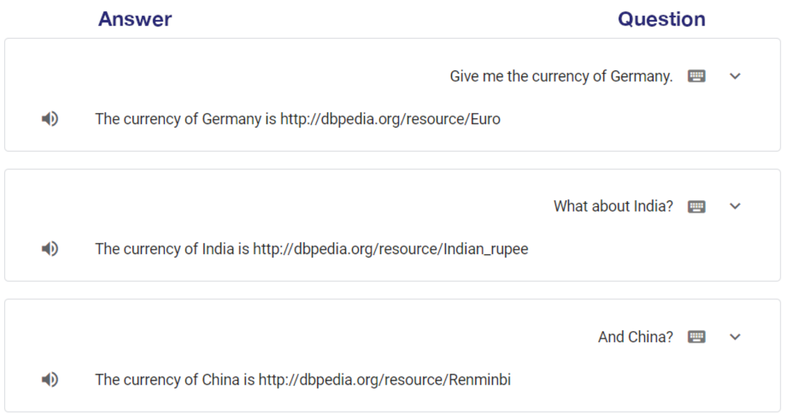
3.2.4 *Query execution and Response.*

Once all entities are resolved, the SPARQL query is formulated by replacing the subject and property in the general SPARQL template. Then the SPARQL query is sent to the SPARQL endpoint for execution. The results from the SPARQL endpoint are usually either a text value such

[7] dbr refers to `http://dbpedia.org/resource/`

[8] Cf. the demo video for a clarification case: https://zenodo.org/record/4091782/files/Barack_Obama.mov

[9] The system cannot resolve the gender-based co-reference of the relevant named entities yet. But, DBpedia provides the property `foaf:gender` which could be used to identify the gender of all entities involved.


Answer

Question

Give me the currency of Germany.

The currency of Germany is http://dbpedia.org/resource/Euro

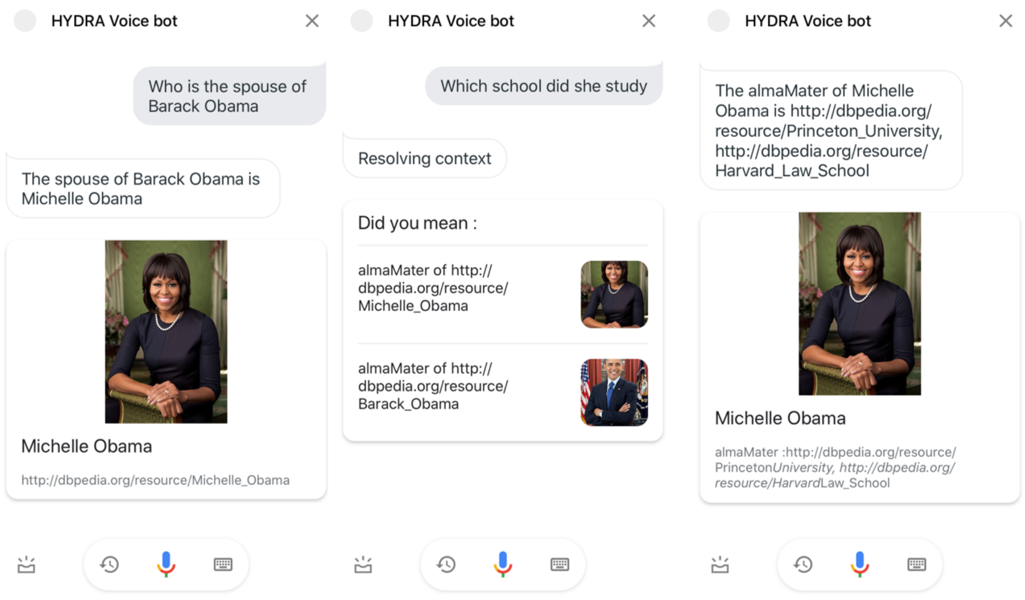
What about India?

The currency of India is http://dbpedia.org/resource/Indian_rupee

And China?

The currency of China is http://dbpedia.org/resource/Renminbi

Figure 3: Sample conversation for a change of the focus entity.


HYDRA Voice bot

Who is the spouse of Barack Obama

The spouse of Barack Obama is Michelle Obama

Michelle Obama

http://dbpedia.org/resource/Michelle_Obama

HYDRA Voice bot

Which school did she study

Resolving context

Did you mean :

almaMater of http://dbpedia.org/resource/Michelle_Obama

almaMater of http://dbpedia.org/resource/Barack_Obama

HYDRA Voice bot

The almaMater of Michelle Obama is http://dbpedia.org/resource/Princeton_University, http://dbpedia.org/resource/Harvard_Law_School

Michelle Obama

almaMater :http://dbpedia.org/resource/PrincetonUniversity, http://dbpedia.org/resource/HarvardLaw_School

Figure 4: Asking for Clarification

as a number or a DBpedia resource URI. We then formulate the response as a complete sentence. For this, we use a template of the format “The `property` of `subject` is `result`”. This text response is converted to speech using Google Text To Speech (TTS) and presented as spoken output to the user. If the result of the SPARQL query returns a DBpedia resource then we execute an additional SPARQL query to obtain the thumbnail associated with that resource if any[10].

Figure 5 shows an example conversation which answers the question "Who was the doctoral supervisor of Albert Einstein?" from the QALD 7 train dataset in a series of follow-up questions and Figure 6 shows another sample conversation that answers the question "What is the currency of Czech Republic" in a series of follow-up questions.

[10]We prepared video demos for some sample conversations: https://doi.org/10.5281/zenodo.4091782

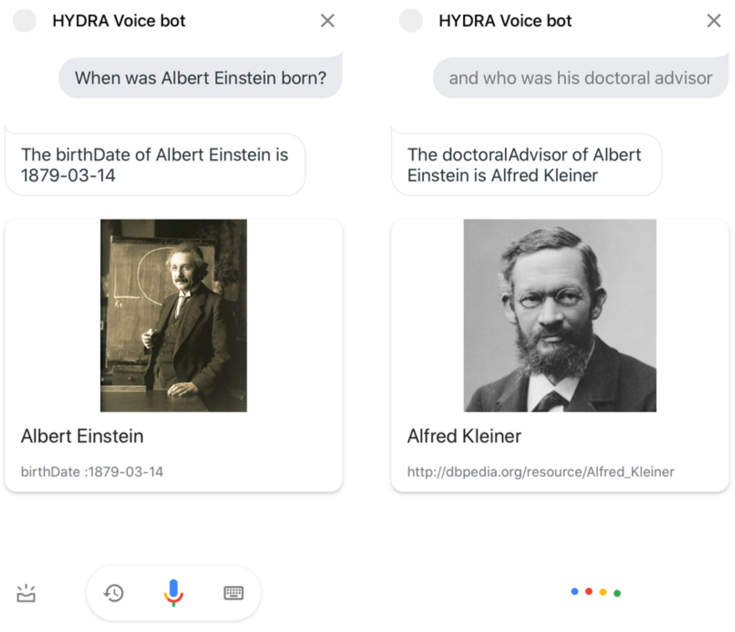
4 EXPERIMENTS

In this section, we describe some experiments to evaluate our system and discuss the results. Section 4.1 introduces our evaluation benchmark. Sections 4.2.1 and 4.2.2 shows the evaluation results of our system against the benchmark. In Section 4.2.3, we take a detailed look at the separate processing steps and their failure rates. Section 4.2.4 describes the results based on different question types and discusses the reasons of failures in each step. We discuss the results of our experiments in Section 4.3.

4.1 Evaluation Benchmark

To the best of our knowledge, benchmarks for conversational QA anchored in DBpedia are not existent[11]. Therefore, we introduce a benchmark of 50 conversations inspired from the QALD datasets. Each conversation consists of a direct question followed by one or two follow-up questions. Overall, the dataset consists of a total

[11]Christmann et al. [1] and Saha et al. [6] just recently published their own benchmarks for conversational QA on knowledge graphs, but they are both based on Wikidata.


HYDRA Voice bot

When was Albert Einstein born?

The birthDate of Albert Einstein is 1879-03-14

Albert Einstein

birthDate :1879-03-14

HYDRA Voice bot

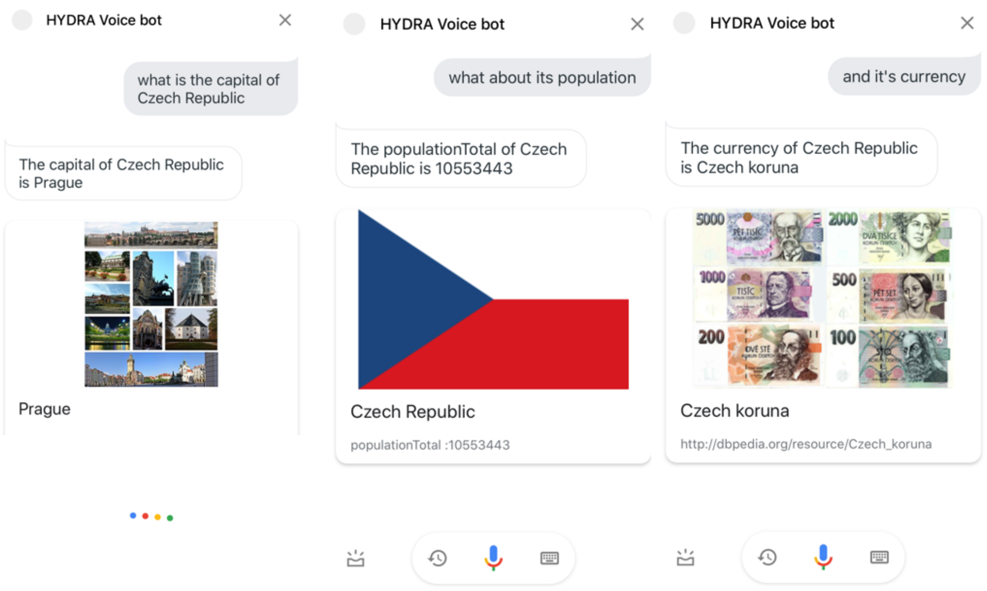
and who was his doctoral advisor

The doctoralAdvisor of Albert Einstein is Alfred Kleiner

Alfred Kleiner

http://dbpedia.org/resource/Alfred_Kleiner

Figure 5: Example query and response (QALD-7-train, question id:167)


HYDRA Voice bot

what is the capital of Czech Republic

The capital of Czech Republic is Prague

Prague

HYDRA Voice bot

what about its population

The populationTotal of Czech Republic is 10553443

Czech Republic

populationTotal :10553443

HYDRA Voice bot

and it's currency

The currency of Czech Republic is Czech koruna

Czech koruna

http://dbpedia.org/resource/Czech_koruna

Figure 6: Example query and response (QALD-7-train, question id:169)

Table 1: Precision of direct and follow-up questions

Type of Question	Number of Questions	Answered	micro-precision
Direct questions	50	42	0.84
Follow-up questions	65	49	0.75
Overall	115	91	0.79

of 115 individual questions[12]. Each conversation corresponds to a question from the QALD-9 dataset.

[12]The dataset is available at https://doi.org/10.5281/zenodo.4091791

4.2 Evaluation Results

Our proposed system performs completely automatically for given questions. Of course, in case of ambiguous follow-up questions a human interaction is required and requested. But the analysis of the questions and the shift of context is performed

Table 2: Percentage of failure by processing step

Processing Step	Percentage of Failure
Speech Recognition	1.74
Response from HYDRA	7.83
Named Entity Recognition	1.74
Mapping of Predicate	5.21
Resolving context	4.35
Total	20.86

Table 3: Percentage of failure by follow-up question type

Follow-up Question Type	Percentage of Failure
WHERE	3.07
WHO/WHOM	7.69
WHICH	4.61
HOW	0
WHEN	0
AND/WHAT + Entity	9.23
AND/WHAT + coreference + Predicate	0
Total	24.61

completely self-sufficient. Hence, the evaluation is executed in this manner. Section 4.2.1 and Section 4.2.2 describe our system's results based on the precision and recall overall and in detail for follow-up questions. In Section 4.2.3 the separate processing steps of our system are evaluated. In addition, the reasons for a failure in each step are discussed. In Section 4.2.4, we evaluate the system based on the different types of follow-up questions and discuss the reasons for failures.

4.2.1 Overall Precision and Recall. We evaluated the overall quality of our system using the measures precision and recall. Recall is defined as the ratio of the number of correct answers provided by the system to the number of gold standard answers with respect to a single question q. In other words, recall is the ratio of intersection of relevant and retrieved answers to the relevant answers. And precision is defined as the ratio of the number of correct answers provided by the system to the number of all answers provided by the system. Precision can also be defined as the ratio of the intersection of relevant and retrieved answers to the retrieved answers.

For the dataset mentioned above, 36 out of 50 conversations were answered completely by our system. i. e. it answered the direct question as well as all follow-up questions correctly. We did not encounter partially correct answers. That means either the system fails to answer a question or the answer returned was 100% accurate. Therefore, recall and precision are the same (= 0.72) for the complete dataset.

4.2.2 Direct and Follow-up queries. We also evaluated the direct and follow-up queries separately and the results are shown in Table 1. The evaluation benchmark consists of 50 direct questions out of which 42 were answered correctly and 49 out of the 65 follow-up queries were answered correctly. As a result the micro-precision of direct queries is 0.84 and that of follow-up queries is 0.75. The overall precision considering 91 out of the total 115 questions answered is 0.79.

4.2.3 Evaluation based on Processing steps. In this section, we present the evaluation results based on the individual processing steps involved and discuss the reasons of failure. Table 2 shows the percentage of failure for each of the processing steps. As discussed in the previous section, the system was able to answer 80% of the questions in overall and the failure rate is 20 %.

Response from HYDRA. The highest percentage of failure is due to an incorrect response to the direct query from HYDRA as the failure of a direct query results in the failure of all of the subsequent follow-up queries. For some answers the HYDRA API took too long to respond or was not able to provide an answer.

Mapping of Predicates. We use PATTY predicates and their synonyms mapped to a custom Dialogflow Entity for identifying predicates. In some cases the system fails to map the correct predicate which results in failure. Consider the example "When was Bach born?" from the conversation id:5 followed by the query "And where?". The system fails to map "where" to "birthPlace" as this is open-ended and could also refer to a location as in "deathPlace". As a result the follow-up query cannot be resolved.

Resolving context. The step involving resolution of co-references or context in follow-up queries is a very crucial step in answering follow-up queries and has a percentage of failure of 4.35. We rely on certain patterns or rules to resolve the context and sometimes the system does not have enough knowledge to resolve them correctly. Consider the example question "What is the capital of French Polynesia?". The follow-up question is "And who is its mayor?" Now, during context resolution it fails to map the coreference "its" to the correct reference (the response of the question) as it does not have the background knowledge that only entities of type city are associated with the predicate "mayor". Hence, it fails to correctly resolve the co-reference in the follow-up query.

Named Entity Recognition. NE mentions in the follow-up queries are annotated using the DBpedia Spotlight API which was successful in resolving most of the entities in our benchmark. However, it fails to resolve certain entities resulting in a failure rate of 1.74. For example, in the follow-up question "And Harry Potter" the entity is resolved to the television show with the same name and not the character Harry Potter which results in an incorrect response.

Speech Recognition. The Google Speech to Text (STT) API was able to correctly transcribe most of the queries correctly and had a very low failure rate of 1.74. However, it fails to transcribe few entities such as the entity "Kurosawa" and "MI6" correctly which could also vary depending upon the accent or the way different users pronounce certain words. It however performed well overall during the evaluation and was able to transcribe most of the input accurately.

4.2.4 Evaluation based on Question Types. In this section we evaluate the system based on the different types of follow-up questions. The system was able to correctly answer 75 % of the follow-up queries and has an overall failure rate of 24. 61 %. Table 2 shows the percentage of failure for each of the question types.

4.3 Discussion

The quality of Conversational QA systems depends on the success of previous questions along the course of the conversation and especially of the first question. As shown in Table 2, the success of a conversation fails when the response of the initial

QA API is incorrect. Therefore, QA in general is required to be the objective of our further research. Secondly, we identified the issue of predicate mapping when the question mentions a relationship that cannot be mapped to a predicate in the knowledge graph. The lack of alternative labels for parts of the ontology is common for most knowledge graphs – properties and ontology classes often have one label whereas the entities mostly have a main label and alternative label. Therefore, further research regarding ontology enrichment is required to provide more information about the ontology properties and classes. Unfortunately, a direct comparison of results to other approaches is not feasible, because of the lack of a common DBpedia-based benchmark and published results respectively.

5 SUMMARY & FUTURE WORK

In this paper, we present our approach to conversational QA using a shift of context. We developed a first prototype that identifies the type of context shift. Thus, a conversation consisting of several request about specific topics / named entities can be conducted. But, in contrast to other recent approaches, our application is also able to handle slight changes of context and the follow-up question are not required to ask for facts in direct proximity of the primary sub-graph within the knowledge graph. Likewise, smart speech interfaces, such as Alexa or Siri, are able to do a conversation in a very limited way, for instance when asked for the weather of different cities in a sequence. Whereas our approach is able to handle conversations about a wide range of domains and with slight changes of topics. In addition, we propose a novel dataset for evaluation purposes of conversational QA based on DBpedia. The dataset consists of 50 conversations and 115 questions and it is publicly available. As discussed in the evaluation section, the improvement of basic QA system is essential for the quality of a conversation. Here, we pursue the further development of our pattern-based approach to map similar natural language patterns to similar graph patterns. Future work will also include the further utilization of DBpedia context information to construct the application in a more intelligent manner: for the identification of the gender of named entities, or the mapping of natural language phrases to DBpedia properties.

6 ACKNOWLEDGEMENTS

This work was partially funded by the German Research Foundation (DFG) under grant no. SA782/26.

REFERENCES

[1] Philipp Christmann, Rishiraj Saha Roy, Abdalghani Abujabal, Jyotsna Singh, and Gerhard Weikum. 2019. Look Before You Hop&#58; Conversational Question Answering over Knowledge Graphs Using Judicious Context Expansion. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*. ACM, New York, NY, USA, 729–738. https://doi.org/10.1145/3357384.3358016

[2] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. End-to-End Reinforcement Learning of Dialogue Agents for Information Access. In *ACL*.

[3] Max Planck Institute for Informatics. 2014. A large resource of relational patterns. https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/patty/.

[4] Alexander Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. ParlAI: A Dialog Research Software Platform. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Copenhagen, Denmark, 79–84. https://doi.org/10.18653/v1/D17-2014

[5] Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics* 7 (March 2019), 249–266. https://www.aclweb.org/anthology/Q19-1016

[6] Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex Sequential Question Answering: Towards Learning to Converse Over Linked Question Answer Pairs with a Knowledge Graph. *CoRR* abs/1801.10314 (2018). arXiv:1801.10314 http://arxiv.org/abs/1801.10314

[7] Nadine Steinmetz, Ann-Katrin Arning, and Kai-Uwe Sattler. 2019. From Natural Language Questions to SPARQL Queries: A Pattern-based Approach. In *Datenbanksysteme für Business, Technologie und Web (BTW 2019), 18. Fachtagung des GI-Fachbereichs „Datenbanken und Informationssysteme" (DBIS), 4.-8. März 2019, Rostock, Germany, Proceedings*. 289–308. https://doi.org/10.18420/btw2019-18

[8] Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2020. Question Rewriting for Conversational Question Answering. arXiv:cs.IR/2004.14652

[9] Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2018. The Design and Implementation of XiaoIce, an Empathetic Social Chatbot. arXiv:cs.HC/1812.08989