UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Jorão Gomes Jr.

A Hereditary Attentive Template-based Approach for Complex Knowledge Base Question Answering Systems

Juiz de Fora

2021

Jorão Gomes Jr.

A Hereditary Attentive Template-based Approach for Complex Knowledge Base Question Answering Systems

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Ciência da Computação. Área de concentração: Sistemas e Tecnologias da Computação

Orientador: Prof. Dr. Jairo Francisco de Souza

Coorientador: Prof. Dr. Victor Ströele de Andrade Menezes

Juiz de Fora

2021

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

Gomes Jr., Jorão.
A Hereditary Attentive Template-based Approach for Complex Knowledge Base Question Answering Systems / Jorão Gomes Jr.. – 2021.
89 f. : il.

Orientador: Jairo Francisco de Souza
Coorientador: Victor Ströele de Andrade Menezes
Dissertação (Mestrado) – Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação, 2021.

1. Question Answering. 2. Knowledge Base. 3. Complex Question. 4. Semantic Parsing. 5. Neural Networks. I. Francisco de Souza, Jairo, orient. II. Ströele de Andrade Menezes, Victor, coorient. III. Título.

Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Programa de Pós-Graduação em
Ciência da Computação
www.pgcc.ufjf.br

Jorão Gomes Júnior

"A hereditary attentive template-based approach for Complex Knowledge Base Question Answering systems"

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Ciência da Computação.

Aprovada em 18 de outubro de 2021.

BANCA EXAMINADORA

Prof. Dr. Jairo Francisco de Souza – Orientador
Universidade Federal de Juiz de Fora

Prof. Dr. Victor Ströele de Andrade Menezes - Coorientador
Universidade Federal de Juiz de Fora

Prof. Dr. Heder Soares Bernardino
Universidade Federal de Juiz de Fora

Prof. Dr. Anderson Almeida Ferreira
Universidade Federal de Ouro Preto

AGRADECIMENTOS

A minha mãe, Fátima, e irmã, Yasmin, pelo encorajamento e apoio que foram essenciais para a conclusão deste trabalho.

Ao meu orienteador Jairo, por além da amizade, paciência, orientação e liberdade para aplicar minhas ideias malucas, ter me proporcianado diversas expericências acadêmicas desde a graduação.

Ao meu coorienteador Victor, pela parceria, paciência e orientação, sempre vendo o lado positivo dos nossos esforços.

Ao Rômulo, que colaborou ativamente para a realização deste trabalho.

Aos meus amigos, em especial Celow e Nicolas, que estiveram presente durante todo o desenvolvimento deste trabalho, me motivando nos altos e baixos.

A todos os funcionários do ICE que, de alguma forma, contribuiram para finalização deste trabalho.

Por útlimo, a CAPES pelo financiamento desta pesquisa.

“Yesterday is history, tomorrow is a mystery, but today is a gift. That is why it is called present.”

- Master Oogway, Kung Fu Panda

RESUMO

Os sistemas de perguntas e respostas (QA) recuperam a resposta mais relevante para uma pergunta de linguagem natural. Os sistemas de perguntas e respostas sobre Bases de Conhecimento (KBQA) exploram entidades e relações das Bases de Conhecimento (KB) para gerar respostas. Os sistemas KBQA precisam lidar com perguntas que podem ser divididas em dois grupos: perguntas simples e complexas. Perguntas simples são aquelas que contêm respostas diretas que precisam ser detectadas para responder a um pergunta. As perguntas complexas precisam de mais informações do que as explícitas que podem ser extraídas de perguntas simples. É necessário utilizar operações de consulta avançada para coletar a resposta das KB, como exploração de relações indiretas entre entidades, multi-relações, restrições qualitativas e quantitativas, entre outras. Atualmente, os sistemas KBQA alcançam melhores resultados ao responder a perguntas simples, e os sistemas de perguntas e respostas complexas sobre Bases de Conhecimento (C-KBQA) tornaram-se o objetivo para a pesquisa recente. No entanto, faltam estudos que abordem questões complexas na área de KBQA. Este trabalho visa preencher essa lacuna, apresentando um estudo sobre sistemas C-KBQA. A contribuição desta dissertação de mestrado é dividida em dois grupos: um mapeamento sistemático da literatura C-KBQA e uma nova abordagem de correspondência de modelos para sistemas C-KBQA. Primeiro, o mapeamento sistemático mostrou que os sistemas C-KBQA precisam lidar com dois tipos de perguntas: múltiplos saltos e perguntas com restrições. Além disso, foi possível identificar três etapas principais para a construção de um sistema C-KBQA e a utilização de duas abordagens principais neste processo. Em segundo lugar, a abordagem de C-KBQA proposta realiza um casamento entre modelos usando a combinação de análises semânticas e técnicas de redes neurais para prever o modelo de resposta apropriado para uma questão de linguagem natural. A chamada Atenção Hereditária foi criada para auxiliar a Tree-LSTM, e demonstramos a eficácia de nossa solução comparando-a com o estado da arte do conjunto de dados LC-QuAD. Os resultados mostram que nossa abordagem supera os sistemas de última geração.

Palavras-chave: Perguntas e Respostas. Pergunta Complexa. Bases de Conhecimento. Análise Semântica. Redes Neurais.

ABSTRACT

Question Answering (QA) systems retrieve the most relevant answer to a natural language question. Knowledge Base Question Answering (KBQA) systems explore entities and relations from Knowledge Bases (KB) to generate answers. KBQA systems need to deal with questions that can be divided into two groups: simple and complex questions. Simple questions are those that contain direct answers that need to be detected to answer a question. Complex questions need more information than the explicit features that can be extracted from simple questions. It is necessary to use advanced query operations to collect the answer from the KB, such as exploiting indirect relations among entities, multi-relations, qualitative and quantitative constraints, and others. Currently, KBQA systems achieve better results when answering simple questions, and Complex Knowledge Base Question Answering (C-KBQA) systems turned the goal to the recent research. However, there is a lack of studies that address complex questions in the KBQA field. This work aims to fill this gap by presenting a study on C-KBQA systems. The contribution of this master thesis is twofold: a systematic mapping of the C-KBQA literature and a novel template matching approach for C-KBQA systems. First, the systematic mapping showed that C-KBQA systems need to handle with two question types: multi-hop and constraint questions. Also, it was possible to identify three main steps to construct a C-KBQA system and the use of two main approaches in this process. Secondly, our proposed C-KBQA approach performs a template matching using the combination of Semantic Parsing and Neural Networks techniques to predict the appropriate answer template to a natural language question. The so-called Hereditary Attention was created to assist the Tree-LSTM, and we demonstrate the effectiveness of our solution by comparing it to the state-of-the-art in the LC-QuAD dataset. The results show that our approach outperforms the state-of-the-art systems.

Keywords: Question Answering. Complex Question. Knowledge Base. Semantic Parsing. Neural Networks.

LIST OF FIGURES

Figure 1 – Master thesis reading flowchart. 15
Figure 2 – Example of simple question and complex question. 20
Figure 3 – Systematic mapping process. 27
Figure 4 – Complex question by subtype. Figure 4a and 4b presents an example of multi-hop questions and constraint questions, respectively. 29
Figure 5 – C-KBQA pipeline example for the question "When was the director of Titanic born?" as the NLQ input. In the Question parsing step (1) the type of the question and the important subjects are identified. In the Question representation (2) the map and connection of the entities and relations into a KB structure are performed. The Candidate ranking (3) is performed to select the most appropriate KB triple. Finally, the entity of the KB is the answer output. 30
Figure 6 – Example of the question parsing step for the question "When was the director of Titanic born?". 31
Figure 7 – Online and offline phases. 32
Figure 8 – Semantic parsing approaches. 34
Figure 9 – Example of question representation and candidate generation. Figure 9a and 9b present the Semantic Parsing and the Neural Networks pipeline, respectively. 35
Figure 10 – Most used datasets over the years 39
Figure 11 – Confusion Matrix regarding the class C. 40
Figure 12 – Number of publications by year 42
Figure 13 – Difference of time execution between the Tree-LSTM and the standard LSTM architectures. 54
Figure 14 – Hereditary Tree-LSTM. 56
Figure 15 – Question length before sentence length normalization. 59
Figure 16 – Question length after sentence length normalization. 59
Figure 17 – Creation of Dummy SPARQL templates. 60
Figure 18 – Creation of group Dummy SPARQL templates by SPARQL aproximation. 60

LIST OF TABLES

Table 1 – The defined research questions. 24
Table 2 – PICOC definition. 24
Table 3 – Generated search terms from the PICOC definition. 24
Table 4 – Exclusion criteria to eliminate unrelated papers 25
Table 5 – Databases used to execute the search string 26
Table 6 – Papers by subtype of complex question 28
Table 7 – List of papers by approaches for question representation and candidate generation. 33
Table 8 – Databases used for C-KBQA. Read 'NP' as Number of papers using the dataset, 'L' as Logic Form available or not, and 'H' as max-hops in the dataset. . 37
Table 9 – Most cited articles 44
Table 10 – Publishing Venues Types 44
Table 11 – Most Popular Journal and Conference 45
Table 12 – Questions distribution by dummy template and dummy group ids for the Wikidata KB part 1. 61
Table 13 – Questions distribution by dummy template and dummy group ids for the Wikidata KB part 2. 62
Table 14 – Model setup. 63
Table 15 – Results of template matching against the baselines. Read "LC" as LC-QuAD, "Acc" as Accuracy, "MA" as Macro and "W" as Weighted. 65
Table 16 – Results of HTL w/paraph divided for each question type. Read TID as dummy template id (see the Tables 12 and 13 to see the SPARQL dummy template). 67
Table 17 – Full list of papers 84
Table 18 – Questions distribution by dummy template and dummy group ids for the DBpedia KB. 86
Table 19 – Questions distribution by dummy template and dummy group ids for the DBpedia KB part 2. 87
Table 20 – Questions distribution by dummy template and dummy group ids for the DBpedia KB part 3. 88
Table 21 – Dummy SPARQL Template co-occurrences mapping. 89

ACRONYMS

QA	Question Answering
KB	Knowledge Base
KBQA	Knowledge Base Question Answering
C-KBQA	Complex Knowledge Base Question Answering
NLQ	Natural Language Question
S-NLQ	Simple Natural Language Question
C-NLQ	Complex Natural Language Question
HTL	Hereditary Tree-LSTM
RNN	Recurrent Neural Network

LIST OF SYMBOLS

$\forall$	For all
$\in$	Inside
$\xrightarrow{predicate}$	Predicate edge

CONTENTS

1 INTRODUCTION . . . 13
1.1 OBJECTIVES . . . 14
1.2 CHAPTERS OVERVIEWS . . . 14
1.3 CONTRIBUTIONS . . . 15
2 MANUSCRIPT 1: LITERATURE REVIEW . . . 17
2.1 INTRODUCTION . . . 17
2.2 BACKGROUND AND RELATED WORK . . . 19
2.2.1 RELATED WORK . . . 21
2.3 SYSTEMATIC MAPPING . . . 22
2.3.1 MAPPING PROTOCOL . . . 23
2.3.2 MAPPING CONDUCTION . . . 25
2.4 SYSTEMATIC MAPPING REPORT . . . 27
2.4.1 RQ1 — WHAT KINDS OF COMPLEX QUESTIONS ARE DEFINED AS COMPLEX QUESTIONS IN THE QA LITERATURE? . . . 27
2.4.2 RQ2 — WHAT ARE THE MOST COMMONLY USED FEATURES AND METHODS IN COMPLEX QUESTION ANSWERING? . . . 29
2.4.2.1 QUESTION REPRESENTATION AND CANDIDATE GENERATION 32
2.4.3 RQ3 — WHAT ARE THE MOST USED DATASETS AND HOW ARE THEY EVALUATED? . . . 37
2.4.4 RQ4 — HOW THE WORKS HAVE BEEN PUBLISHED OVER THE YEARS? 41
2.5 THREATS TO VALIDITY . . . 46
2.6 TRENDS AND CHALLENGES . . . 47
2.7 CONCLUDING REMARKS . . . 48
3 MANUSCRIPT 2: PROPOSED APPROACH . . . 49
3.1 INTRODUCTION . . . 49
3.2 RELATED WORK . . . 51
3.3 C-KBQA APPROACH . . . 53
3.3.1 TREE-LSTM . . . 53
3.3.1.1 TREE-LSTM WITH HEREDITARY ATTENTION . . . 55
3.4 TRAINING AND EVALUATION METHODOLOGY . . . 58
3.4.1 DATASETS AND PRE-PROCESSING . . . 58
3.4.2 TRAINING AND EVALUATION SETUP . . . 63
3.5 RESULTS . . . 64
3.6 FINAL REMARKS . . . 66
4 CONCLUSIONS . . . 69
4.1 LIMITATIONS . . . 70
4.2 CONTRIBUTIONS . . . 71

4.3 FUTURE WORK 72
REFERENCES 73
APPENDIX A – SYSTEMATIC REVIEW PAPER LIST . . 84
APPENDIX B – ANSWER TEMPLATES ADDITIONAL DATA 85

1 INTRODUCTION

Question Answering (QA) systems have the purpose to retrieve the most relevant information (answer) to a search question made by a user (CROFT; METZLER; STROHMAN, 2010). Unlike search engines, QA systems aim to find the exact answers to a natural language question (YIN; GE; WANG, 2014; RODRIGO; PENAS, 2017). To do so, QA systems need to recognize the information inside a natural language question. This task implies the identification of relevant objects and their connections, extracting the main descriptions or ideas that are contained in a question. The mapping from a natural language question to its main subjects (concepts, organizations, people, etc) is a task that has been explored for QA systems (ABUJABAL et al., 2017; JIA et al., 2018; HAO et al., 2019; LU et al., 2019; BAKHSHI et al., 2020). QA systems make use of semantic structures known as knowledge bases (POPPING, 2003) to perform the extraction of this information and its relations.

Knowledge base (KB) is a data model based on a semantic network, which uses a triple format (subject, predicate, object) to represent and relate the information contained within a data domain (POPPING, 2003; JI et al., 2020). QA systems that made use of KB are also usually called Knowledge Base Question Answering (KBQA). KBQA systems use those semantic structures, for example, Freebase (BOLLACKER; COOK; TUFTS, 2007), Wikipedia (LEHMANN et al., 2015) or Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014) to directly answer the question. So, KBQA systems extract the main features from the text and map them into a knowledge base to answer a question. The use of KB gives a more accurate and concise result, once that a natural language question can be understood and mapped precisely to structured queries over the KB (CUI et al., 2019).

KBQA systems need to deal with different kinds of questions. We can divide them into two groups: simple questions and complex questions. Simple questions are those that contain direct answers and only direct entities that need to be detected to answer a question (BORDES et al., 2015). Complex questions need more information than the explicit features that can be extracted from simple questions. It is necessary to use advanced query operations to collect the answer from the KB, such as exploiting indirect relations among entities, multi-relations, qualitative and quantitative constraints, and others (BAO et al., 2016; QIU et al., 2020a). However, it is hard to extract and map the features of a complex question into a KB since the questions have indirect relations, qualitative information, and many entities/predicates.

Currently, KBQA systems achieve better results when answering simple questions, and Complex Knowledge Base Question Answering (C-KBQA) systems turned the goal to the recent researches in the QA field (QIU et al., 2020b; HUA et al., 2020b; HUA et al., 2020a).

1.1 OBJECTIVES

The problem of answering complex questions was already recognized as a challenge to the KBQA systems (HÖFFNER et al., 2017; RODRIGO; PENAS, 2017). The C-KBQA area is receiving huge attention from researchers, and great advances are expected. The objectives of this research are to analyze the challenges and limitations of C-KBQA systems and to present a KBQA approach capable to classify complex natural language questions into answer templates. In this master thesis, these objectives were divided into threefold: (i) provide an overview of the C-KBQA area including solutions and evaluation methods; (ii) present limitations of C-KBQA datasets and release a new version of the C-KBQA dataset to fill this gap for other researchers; (iii) present a novel C-KBQA approach that extracts the semantic of a complex question and maps it into an intermediate format to further request the answer to the question on the KB.

1.2 CHAPTERS OVERVIEWS

This master thesis is divided into four chapters. Chapters 2 and 3 are full research papers and Chapter 4 connects the previous two chapters, explaining how each chapter is connected to the others, their findings, limitations, and the general context of the research. The two research papers were submitted to peer-reviewed journals, and each chapter of this master thesis is structured as follows:

- **Chapter 2 Literature Review[1]:** To analyze the C-KBQA area, a systematic mapping of the literature was carried out. This chapter presents an overview of the C-KBQA, and the main objective of this mapping is to understand the state of scientific research in C-KBQA. This chapter wa submitted to the Journal “Knowledge and Information Systems”.
- **Chapter 3 Proposed Approach[2]:** This chapter presents a template matching-based approach for C-KBQA systems using a novel hereditary attention mechanism for Tree-LSTM. The approach was created using the combination of Semantic Parsing and Neural Networks techniques to determine the answer template that a complex question matches. An attention mechanism was created to assist a Tree-LSTM in selecting the most important information of a natural language question Also, a new cleaned version of a C-KBQA dataset containing answer templates was released. This chapter presents the approach for C-KBQA systems as well the results and

[1] Authors: Jorão Gomes Jr., Rômulo Chrispim de Mello, Victor Ströele, and Jairo Francisco de Souza

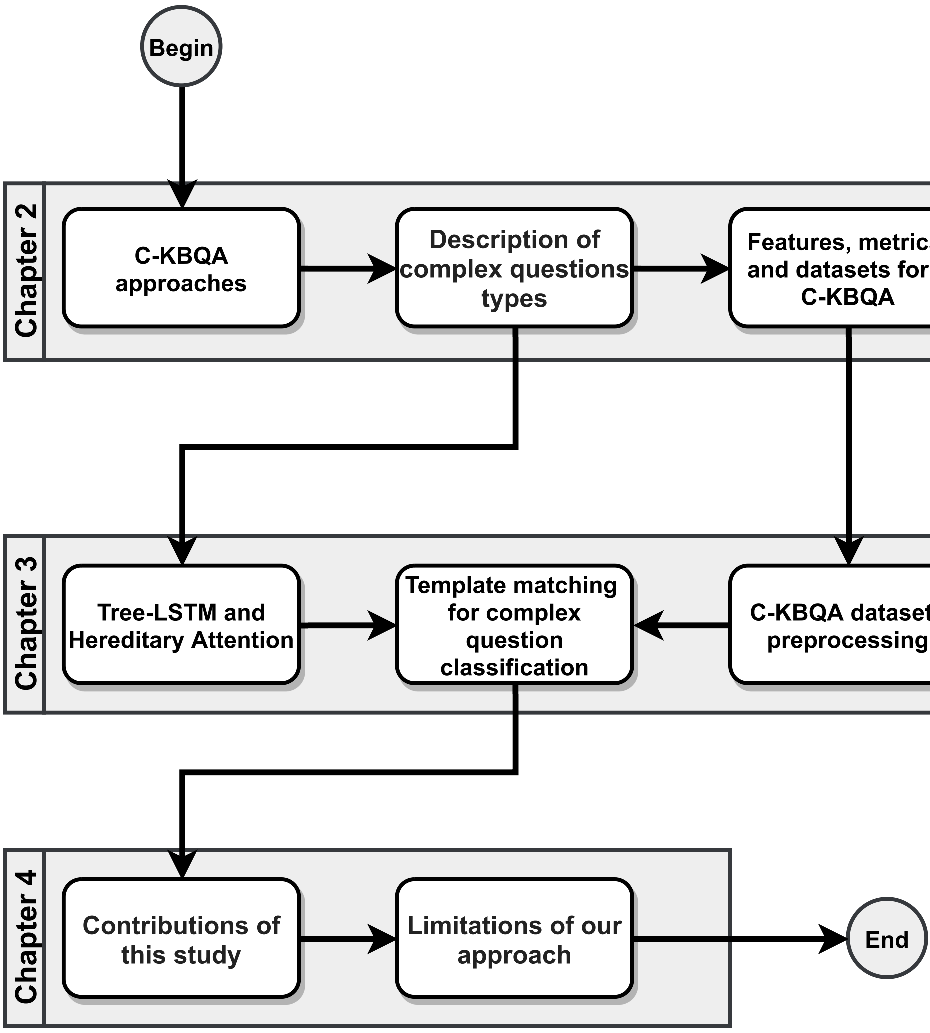
[2] Authors: Jorão Gomes Jr., Rômulo Chrispim de Mello, Victor Ströele, and Jairo Francisco de Souza

limitations for complex question classification. This chapter was submitted to the "Expert Systems with Applications" Journal.

- **Chapter 4 Conclusion and future works:** Finally, this chapter discusses the methodologies and findings of each of the previous chapters, presenting the main contributions, limitations, and perspectives for future work.

Figure 1 shows the workflow of the steps taken in the research conduction and in which Chapter the reader can find details of each step. Through Figure 1 readers can go through the document and find the subject that interests them most.

Figure 1 – Master thesis reading flowchart.



Source: Created by the author (2021)

1.3 CONTRIBUTIONS

The main contributions of this work are presented below:

1. An overview of the C-KBQA field through four research questions (RQ): RQ1 — "What kinds of complex questions are defined as complex questions in the QA literature?", RQ2 — "What are the most commonly used features and methods in C-KBQA?", RQ3 — "What are the most used datasets and how are they evaluated?", and RQ4 — "How has the research been published over the years?". The study presents a map of a collection of 54 papers systematically selected from 894 papers and the identification of the most frequent methods, venues, knowledge bases, datasets, and metrics used in the literature.

2. An attention mechanism was created to assist a Tree-LSTM in selecting the most important information of a natural language question. In the so-called Hereditary Attention, each neural network cell inherits the attention from another neural network cell in a bottom-up way.

3. A template matching approach using the combination of Semantic Parsing and Neural Networks techniques to determine the answer templates that a complex question match was created. The Hereditary Attention is used to perform the template matching.

4. A new cleaned version of a C-KBQA dataset. The original dataset was preprocessed and a set of dummy answer templates were created.

2 MANUSCRIPT 1: LITERATURE REVIEW

Question Answering (QA) systems retrieve the most relevant answer to a natural language question. Knowledge Base Question Answering (KBQA) systems explore entities and relations from Knowledge Bases to generate answers. Currently, QA systems achieve better results when answering simple questions, but Complex QA systems are receiving great attention nowadays. However, there is a lack of studies that analyze complex questions inside KBQA field and how it has been addressed. In this master's thesis, the main objective of this chapter is to provide an overview of Complex Knowledge Base Question Answering (C-KBQA). This chapter is a full article[1] that presents a systematic mapping of the works published in KBQA area with an emphasis on C-KBQA approaches. The main contribution of this chapter is the identification of methods, datasets, metrics, gaps and future directions for C-KBQA literature.

2.1 INTRODUCTION

Question Answering (QA) systems have the purpose to retrieve the most relevant information (answer) to a search question made by a user (CROFT; METZLER; STROHMAN, 2010). Different from search engines, the QA system aims to find the exact answers to a natural language question (YIN; GE; WANG, 2014; RODRIGO; PENAS, 2017). To do so, QA systems need to recognize the information inside a natural language question. This task implies the identification of relevant objects and their connections, extracting the main descriptions or ideas that are contained in a question. The mapping from a natural language question to its main subjects (concepts, organizations, people, etc) is a task that has been explored for QA systems (ABUJABAL et al., 2017; JIA et al., 2018; HAO et al., 2019; LU et al., 2019; BAKHSHI et al., 2020). QA systems make use of semantic structures known as knowledge bases (POPPING, 2003) to perform the extraction of this information and its relations.

A Knowledge base (KB) is a data model based on a semantic network, which usually uses a triple format (subject, predicate, object) to represent and relate the information contained within a data domain (POPPING, 2003; JI et al., 2020). QA systems that make use of KB are called Knowledge Base Question Answering (KBQA). KBQA systems use those semantic structures, for example, Freebase (BOLLACKER; COOK; TUFTS, 2007), Wikipedia (LEHMANN et al., 2015) or Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014) to directly answer the question. So, KBQA systems extract the main features from the text and map them into a knowledge base to answer a question. The use of KB gives a more accurate and concise result, once a natural language question can be understood and mapped precisely to structured queries over the KB (CUI et al., 2019).

[1] **Title**: A Study of Approaches to Answering Complex Question over Knowledge Bases

Even with the use of KB, QA systems still need to deal with different kinds of questions. We can divide them into two groups: simple questions and complex questions. Simple questions have direct answers, that is, the answer requires few facts from the KB, such as a single subject-predicate-object triple (BORDES et al., 2015). On the other hand, answering complex questions needs more information than explicit ones extracted from simple questions. In this case, it is necessary to use more advanced query operations to collect the facts from the KB, such as exploiting indirect relations among entities, multi-relations, qualitative and quantitative constraints, and others (BAO et al., 2016; QIU et al., 2020a). In the simple questions, the main objects can be directly extracted by Natural Language Processing (NLP) tools and mapped into a knowledge base, however, it is hard to accomplish this task when the questions have indirect relations, qualitative information, and many objects/facts. Currently, QA systems achieve better results when answering simple questions, which contain direct links and only a few facts related to the question. Due to this, Complex Question Answering systems are receiving great attention (VAKULENKO et al., 2019; DING et al., 2019; MAHESHWARI et al., 2019; REDDY; MADHAVI, 2020; HUA et al., 2020c).

The problem of answering complex questions was recognized as a challenge to this scenario (HÖFFNER et al., 2017; RODRIGO; PENAS, 2017). However, there is a lack of studies that analyze complex natural language questions inside the KBQA field and how it has been addressed. Thus, this work aims to fill this gap, presenting a systematic mapping on the Complex Knowledge Base Question Answering (as put forward in this paper we will use C-KBQA to refer to this term) systems. Understanding the solutions for C-KBQA includes the investigation of techniques that are most used, the main current solutions, where these solutions are applied, and, therefore, the identification of the main challenges of this research field. The systematic mapping of the literature was performed following the principles presented by (KITCHENHAM, 2004). Papers that present QA systems to answer complex questions using knowledge bases were selected. These articles were extracted from well-known scientific databases and analyzed according to the technique performed, the attributes employed, the context (domain) of the work, the evaluation method, and the benchmarks used by the authors.

The main contributions of this paper are: (i) the use of a systematic method to provide an overview of the state of the art in Complex Knowledge Base Question Answering; (ii) a collection of 54 papers systematically selected from 898 papers; (iii) the identification of the most frequent venues, domains, and knowledge bases used in the literature; (iv) a mapping of methods, datasets, and metrics used in the complex question answering scenario; (v) future directions and the main gaps in the C-KBQA area. We show that the C-KBQA system tries to solve two types of complex questions: Multi-hop questions and Constraint questions. Also, we identified three main steps to construct C-KBQA systems and the use of two main approaches in this process. We also notice that datasets for C-KBQA are still an open challenge, and the most used datasets are composed of only a few kinds of complex questions. At last, we saw that the C-KBQA area is still rising and it is expected to see a new C-KBQA system or new modules trying to improve the current C-KBQA system over the next years.

The remainder of the paper is structured as follows: Section 2.2 presents the background foundation and compares this work with other reviews in the KBQA field. Section 2.3 describes the systematic mapping protocol that has been followed. Section 2.4 presents the mapping report and the discussion about it. Section 2.5 presents some threats to the validity of this study. Section 2.6 presents the trends, challenges, and next steps for C-KBQA. Finally, Section 2.7 presents our concluding remarks and future directions of the C-KBQA area.

2.2 BACKGROUND AND RELATED WORK

The QA field has a solid foundation, being studied for many years. Despite that, sometimes QA systems are confused with other search engines. Different from search engines that are able to return a list of relevant items as a response from a Natural Language Question (NLQ), QA systems aim to find exact answers to an NLQ made by the users (YIN; GE; WANG, 2014; RODRIGO; PENAS, 2017). All the processing made by QA systems consists of automatically transforming the input question, using a more sophisticated natural language processing technique to get the answer. Therefore, sometimes QA systems can be considered as the next step beyond current search engines (IMAM et al., 2011).

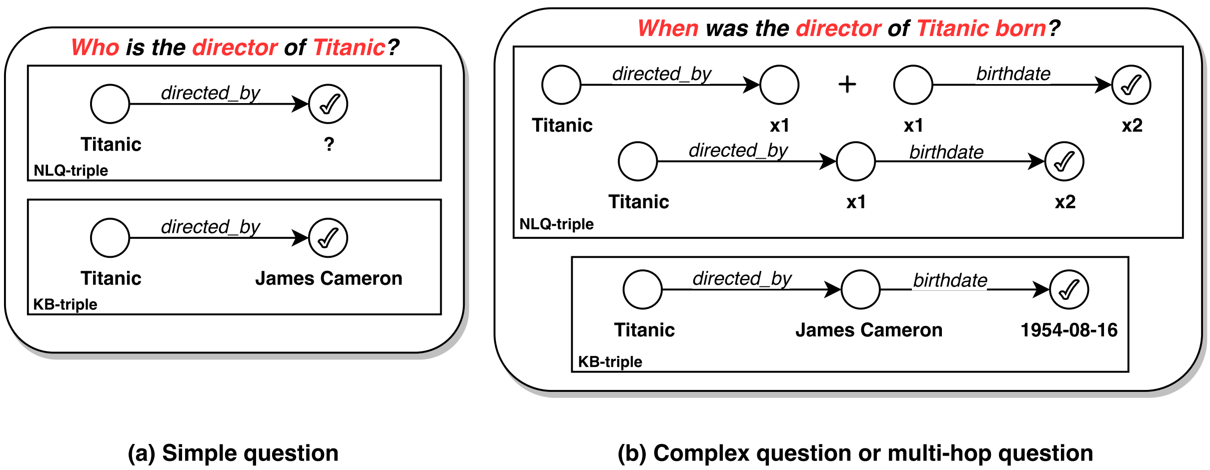
QA systems need to recognize NLQ subjects to answer it. To do this, it is necessary to get the relevant subjects and their connections by extracting the main information contained in an NLQ. The mapping from an NLQ to their main subjects (concepts, organizations, people, etc) is a task that has been explored for QA systems and is sometimes called Semantic Parsing. Semantic Parsing is the process that transforms an NLQ into an intermediate representation that can be further represented in logic form (TRIVEDI et al., 2017). QA systems can make use of Semantic Parsing to map the subjects of an NLQ into structured data and then answer the question concisely with the advance of Web metadata patterns.

Several metadata patterns have been proposed to help systems that use information retrieval techniques to improve the quality of the information produced, founding the concept called Semantic Web (BERNERS-LEE; HENDLER; LASSILA, 2001). The term Linked Data is used to describe a set of practices for publishing, connecting, sharing, and disseminating structured data across multiple domains (BIZER et al., 2011). KB is one form of representation of the knowledge created by the Linked Data (e.g., Freebase (BOLLACKER et al., 2008) and Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014)).

The creation of QA systems over the Semantic Web has received strong attention due to the increase of structured knowledge databases in different domains using Linked Data (RAJABI; SANCHEZ-ALONSO, 2019). For the QA scenario, the KB is used as a source of information of the QA system, where the system needs to identify entities in the NLQ and correctly process the KB structure to match the entities with objects/triples from the KB (this type of QA is usually known as Knowledge Base Question Answering — KBQA). In contrast to web search, KBQA gives out accurate and concise results, provided that an NLQ can be precisely mapped to structured queries over the KB (CUI et al., 2019).

Despite advances in KBQA systems, simple and direct questions are not always requested by users. Simple natural language questions (S-NLQ) can be defined as the problem of answering questions that only pertain to one fact from the knowledge base (BORDES et al., 2015). In an example presented in Qiu et al (QIU et al., 2020a), to answer the question "Who is the director of Titanic?", it is necessary to extract the object "Who", "director", and "Titanic", map it a KB triple format like ($Titanic$, $\xrightarrow{directed\_by}$, $James\ Cameron$) to get "James Cameron" as an answer. However, in a real scenario, the users usually tend to ask more complex questions, which need a greater number of facts, indirect relationships, quantitative and qualitative information, among others, to be answered. If the question "When was the director of Titanic born?" was requested, first the detection and mapping of the main subject are needed ($Titanic$, $\xrightarrow{directed\_by}$, $James\ Cameron$) and finally make a hop in the KB-triple to answer this question ($James\ Cameron$, $\xrightarrow{birthdate}$, 1954/08/16) (QIU et al., 2020a). Figure 2 illustrates the example.

Figure 2 – Example of simple question and complex question.



Source: Created by the author (2021)

This makes the problem of answering these complex natural language questions

(C-NLQ) costly for conventional question answering systems. So, C-KBQA systems turned the goal to the recent researches in the QA field (QIU et al., 2020b; HUA et al., 2020b; HUA et al., 2020a).

2.2.1 RELATED WORK

According to Dimitrakis, Sgontzos e Tzitzikas (2019), the problem of answering complex questions was recognized as a challenge for QA over Linked Data systems in (HÖFFNER et al., 2017; RODRIGO; PENAS, 2017). However, there is a lack of works that present a survey or a mapping of this problem in the Knowledge Base Question Answering (KBQA) field. Previous works presented mappings about QA, but focused on simple question answering systems and presented a brief introduction to the complex questions subject even in recent QA reviews (HÖFFNER et al., 2017; WU; ZHANG; FENG, 2019; SILVA et al., 2020). These works did not make clear the C-KBQA problem and how it has been solved in the literature.

In Höffner et al. (2017), it is performed a survey on challenges of question-answering in the semantic web. The authors gather a total of 62 systems developed from 2010 to 2015. Nonetheless, the complex question problem is not discussed in detail. The authors have only one section that presents a high-level overview of the complex question problem. That was expected because the papers analyzed by the authors were published until the beginning of 2015 and research on complex question answering was incipient (see Section 2.4). The same occurs in Rodrigo e Penas (2017) where the authors made an explanation about the evaluation of QA systems. In contrast to Höffner et al. (2017), Rodrigo e Penas (2017) had the goal to present an overview of all the metrics used to evaluate QA systems. During their evaluation, they argue that QA systems are not yet prepared to perform well inside the complex question answering scenario as they already do in the simple question scenario. The same is present in another recent review (DIMITRAKIS; SGONTZOS; TZITZIKAS, 2019), which shows that the current state-of-the-art QAs cannot solve such complex problems, mainly due to the difficulty of extracting more than one fact to answer a question, looking for related information on three or more edges of distance in the KB graph, and understanding the user's intention. However, with the growth of research in the complex question area, it is evident that this is not true at all. We will show the advance in this challenge in the following Sections.

An overview of the KBQA problem is presented in Diefenbach et al. (2018). The authors presented the main methods and approaches inside this scenario. However, the review is focused only on the QA systems that participated in the "Question Answering over Linked Data" (QALD[2]) challenge. Although the QALD is a well-known challenge in the QA scenario, it contains only between 100 and 450 questions (DIEFENBACH et al.,

[2] <http://qald.aksw.org/>

2020). Evaluating only the participants of this conference becomes a limited map of the scenario with the advancement of QA systems. This can be confirmed by QALD challenge having its own evaluation dataset and most of the complex question datasets are not included in this conference, for example, ComplexWebQuestions (TALMOR; BERANT, 2018) or LC-QUAD (TRIVEDI et al., 2017) that we will discuss in detail in Section 2.4.

Finally, in Wu, Zhang e Feng (2019) the authors present a general mapping of the current KBQA scenario. Like in Diefenbach et al. (2018), Wu, Zhang e Feng (2019) only discusses small gaps in the complex question scenario and also shows that the researches in complex issues need improvements. This confirms the need to study and understand the complex question problem. The main goal of our work is to help other researchers in this field and allow them to understand the main concepts and challenges of C-KBQA.

After this brief explanation, our work differs from the others by performing an analysis of the methods proposed for the complex question answering systems over knowledge bases. Furthermore, we performed a systematic approach aiming to find the works in a more accurate and impartial way, giving other researchers the option to reproduce by themselves all the processes that are made here. To the best of our knowledge, there is no systematic mapping that analyzes the existing solutions that can provide support to C-KBQA. So, this work presents a quantitative and qualitative mapping of several aspects of C-KBQA and allows an overview of what has been done in the area. Therefore, this study makes the first contact of new researchers with this topic easier.

2.3 SYSTEMATIC MAPPING

Systematic reviews and systematic mapping are two main methods for synthesizing scientific evidence in Software Engineering (PETERSEN; VAKKALANKA; KUZNIARZ, 2015). Systematic reviews focus on gathering and synthesizing evidence (PETERSEN; VAKKALANKA; KUZNIARZ, 2015) while systematic mapping studies are used to structure a research area. Systematic mapping is a second empirical study that provides an overview of state of the art in a field, identifying venues and topics addressed in the literature (FERNANDEZ-SOTOS et al., 2019). Furthermore, this type of method is a systematic approach to understanding the “map” of a field of knowledge, research question, or practice (PERRYMAN, 2016), by identifying connections rather than results (COOPER, 2016). A systematic map is a defined method to build a classification scheme and to structure a field of interest. The analysis of results focuses on frequencies of publications for categories within the scheme (PETERSEN et al., 2008). Rather than providing answers to specific questions of impact, these maps can be proven as highly useful for research, policy, and practice communities by providing assessments of knowledge gaps and patterns across the research literature that promote best practice (HADDAWAY et al., 2016). Systematic Mappings are focused on a visual synthesis of the data and are best designed for: (1) when there is an abundance and a diversity of research; (2) as the first step to a systematic review; (3) to identify gaps in a topic area.

The objective of the present work is to find the main approaches developed in the area of complex question answering over knowledge base systems. Systematic mapping is suitable for this area of knowledge due to the growth of new papers during the recent years, the diversity of approaches in this research field, the lack of systematic reviews of complex question answering systems over knowledge bases, and the difficulty of gathering information on datasets, features, and methods used in literature. To the best of our knowledge, this is the first systematic mapping in the field of complex questions over knowledge bases.

This section shows the protocol used to create an overview of existing research on a particular subject so that the least possible interference of the researcher's bias occurs. Thus, the final product can be used by new researchers interested in the area. This systematic mapping's main goal is to help other researchers that are starting in the complex question answering field to know where to start and with what to start their research. With this, they can spend more time researching new solutions or improving existing solutions as it is proposed by some researchers (SINGH et al., 2018).

2.3.1 MAPPING PROTOCOL

To reduce the bias and make the study reproducible for other researchers, we had to take some precautions. In this way, a protocol was adopted for the execution of systematic mapping. The process used in this paper was based on the same protocol presented by Neiva et al. (2016). In this section, we will detail all steps taken to elaborate the research protocol. The planning process consisted of the following steps: (i) definition of the research questions, (ii) selection of the relevant search terms, (iii) definition of the exclusion criteria, and (iv) selection of the research repositories.

This research aims to identify and understand what is being developed for the complex question answering over knowledge bases field. Thus, the research objectives were defined according to research questions. As discussed in (PETERSEN; VAKKALANKA; KUZNIARZ, 2015), research questions aim to categorize and create an overview of the literature, discovering covered topics in the research area. We defined the research questions that should be answered, as can be seen in Table 1.

Based on the research objectives, the scope was defined using the PICOC method (Population, Intervention, Comparison, Outcome, and Context) (PETTICREW; ROBERTS, 2006). The PICOC method helps to identify relevant keywords from the objectives associated with each of its entries. It is possible to define the search terms, keywords, and synonyms that must be used to find the relevant papers for this research. Table 2 describes the PICOC elements for this paper and Table 3 defines the search terms defined for each

Table 1 – The defined research questions.

	Questions
RQ1	What kinds of complex questions are defined as complex questions in the QA literature?
RQ2	What are the most commonly used features and methods in C-KBQA?
RQ3	What are the most used datasets and how are they evaluated?
RQ4	How has the research been published over the years?

Source: Created by the author (2021)

PICOC element.

Table 2 – PICOC definition.

Element	Description
Population (P)	Articles that present question answering systems
Intervention (I)	Approaches to address complex questions
Comparison (C)	-
Outcome (O)	The solutions to answer complex questions
Context (O)	QA systems which make use of knowledge bases

Source: Created by the author (2021)

Table 3 – Generated search terms from the PICOC definition.

Element	Search term and synonyms
Population (P)	question answering, qa, semantic search, search engine, answering engine
Intervention (I)	complex question, complex information, complex queries, complex query, complex answer
Comparison (C)	-
Outcome (O)	method, technique, algorithm, approach, application, system
Context (O)	knowledge base, knowledge graph, kb, kg, linked data, linked open data, lod, semantic web, semantic data

Source: Created by the author (2021)

A logical query string was created using the terms generated (Table 3). Each PICOC element was separated by an *AND*, and each synonym term in the same PICOC element was separated by *OR*. From the search terms found, we created a search string that can be represented as:

("question answering" OR qa OR "semantic search" OR "answering engine" OR "search engine")AND("complex question" OR "complex information" OR "complex queries" OR "complex query" OR "complex answer")AND(method OR

technique OR algorithm OR approach OR application OR system)AND("knowledge base" OR "knowledge graph" OR kb OR kg OR "linked data" OR "linked open data" OR lod OR "semantic web" OR "semantic data")

Although the search terms used for this study were extracted from the PICOC field analysis, works unrelated to the research's purpose can still be found. Thus, some criteria were chosen so that these works were excluded during the process. These criteria are listed in Table 4.

Table 4 – Exclusion criteria to eliminate unrelated papers

	Exclusion Criteria
EC1	Duplicates
EC2	Articles that do not present a system for Question Answering using Knowledge Bases
EC3	Articles that do not try to solve the Complex Question problem
EC4	Article not written in English
EC5	Grey literature[a]
EC6	Articles that are not available in full text.

[a] We consider as grey literature the manuscripts published without a peer review process, such as pre-prints, technical reports, patents, and others.

Source: Created by the author (2021)

After defining the research questions and the exclusion criteria, the following steps were performed to determine the databases based on which the research would be carried out, according to (COSTA; MURTA, 2013). The requirements adopted were: (i) Databases can perform searches using logical expressions or similar mechanisms; (ii) They allow searches to be made to encompass all text or just specific fields (e.g., title, abstract)[3]; (iii) They must be available at the researcher's institution. Based on this, the databases chosen to execute the search string are listed in Table 5.

After the protocol definition, we performed our mapping and collected the papers related to the research field. As the authors did not find any other mapping in the complex question answering field to compare them, the mapping did not have any search restriction, like year restriction (e.g., only papers after 2014) or field restriction (e.g., only papers in Computer Science).

2.3.2 MAPPING CONDUCTION

The mapping was conducted by executing the search string in each scientific repository presented in Table 5. Once the articles list of each database was recovered,

[3] As Google Scholar does not have native metadata feature filtering, we created a script that does it using the HTML of the pages and Regular Expressions. The script is freely available and can be accessed at <https://github.com/lapic-ufjf/gscholar-review-filter>

Table 5 – Databases used to execute the search string

Scientific papers databases	Access link
Scopus	<http://www.scopus.com>
Google Scholar	<https://scholar.google.com.br/>
ISI Web of Science	<http://www.isiknowledge.com>
IEEE Digital Library	<http://ieeexplore.ieee.org>

Source: Created by the author (2021)

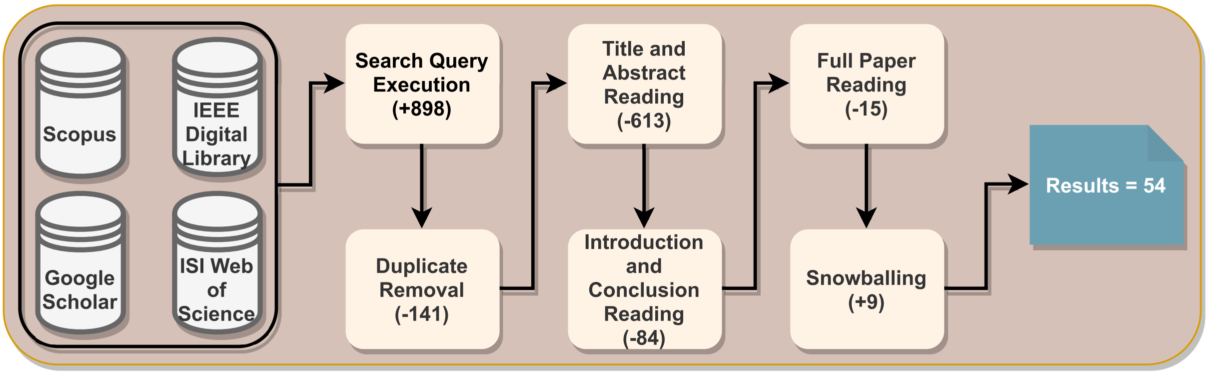
we first removed all the duplicate papers (EC1 — see Table 4). To organize the results retrieved and remove the duplicates, we used a tool called *Parsif.al*[4], which helped during the protocol steps. For the remaining steps, all exclusion of papers was done considering the remaining exclusion criteria (EC2 — EC6).

In the second stage, the exclusion was based on reading the title and abstract of the papers. Papers that did not have relevance for this mapping were removed. The next stage was reading the introduction and conclusion of each paper in the previous step, filtering the papers again out of the scope of this research (Table 2). The remaining articles were completely read and analyzed according to the mapping questions (Table 1). Finally, we performed the Snowballing step. This step aims to find relevant articles that were not returned by the search string by looking at the works cited in the references of the accepted papers. In the last two stages, besides the exclusion criteria, the quality of the paper was also taken into account before the questions were raised to exclude the articles that did not have answers to the mapping questions. The protocol pipeline and the results obtained can be seen in Figure 3. The steps were performed by multiple persons to solve possible divergences of evaluations found between the results, obtaining better suitability of the research results.

In the first step of the protocol, 898 papers were obtained through the set of the four scientific repositories, where 725 papers were given by Google Scholar, 142 papers by Scopus, 25 papers by Web of Science, and 6 papers by IEEE Digital Library. This search was performed on November 17, 2020, out of which 141 (15.70%) duplicated articles were removed. The remaining 757 articles were analyzed through title and abstract reading, where 613 (80.98%) articles were excluded and 144 (19.02%) were maintained for further analysis. The 144 papers selected in the second phase had their introduction and conclusion read. Based on the exclusion criteria, at the end of this stage, 60 (41.67%) articles remained. These selected papers correspond to 6.68% of the articles initially selected. After reading the full text and applying the exclusion criteria, 45 papers (5.01% of the initial papers) were selected. Finally, 9 papers were added after the snowballing

[4] <http://parsif.al>

Figure 3 – Systematic mapping process.



Source: Created by the author (2021)

step was performed resulting in 54 papers to be mapped in this paper. We made all data and spreadsheets generated for this work available for further research here[5].

2.4 SYSTEMATIC MAPPING REPORT

The following sections discuss the collected data to answer the research questions. The accepted papers list and each paper ID can be found in Table 17 (Appendix). We will refer to the paper IDs in the following mapping tables.

2.4.1 RQ1 — WHAT KINDS OF COMPLEX QUESTIONS ARE DEFINED AS COMPLEX QUESTIONS IN THE QA LITERATURE?

Simple questions already have a clear definition and applications presented in the literature. However, the same is not true for complex questions. The complex question term is related to different sub-problems in C-KBQA: multi-hop questions and constraint questions. We found 52 works addressing multi-hop questions and 17 works addressing constraint questions. Table 6 presents the papers classified by the type of complex questions.

To address multi-hop questions, a C-KBQA system has to handle several subjects and predicates from the KB. The entities detected in those questions need to be linked to the KB entries. It is necessary to deal with indirect relations, unlike simple questions that can be answered directly. The triple connections (s,p,o) inside a KB are explored, and systems make hops between the objects detected in the C-NLQ and the KB relations to get the target entity (see Figure 2b).

To address constraint questions, C-NLQ also includes some restrictions that limit the answering options for a given question (SHIN; LEE, 2020). Those restrictions can be

[5] https://github.com/lapic-ufjf/CKBQA-systematic-mapping-2021

Table 6 – Papers by subtype of complex question

Complex questions	Paper ID	# Papers
Multi-hop	1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54	52
Constraint	1, 3, 6, 11, 18, 19, 20, 21, 27, 28, 35, 39, 43, 44, 51, 53, 54	17

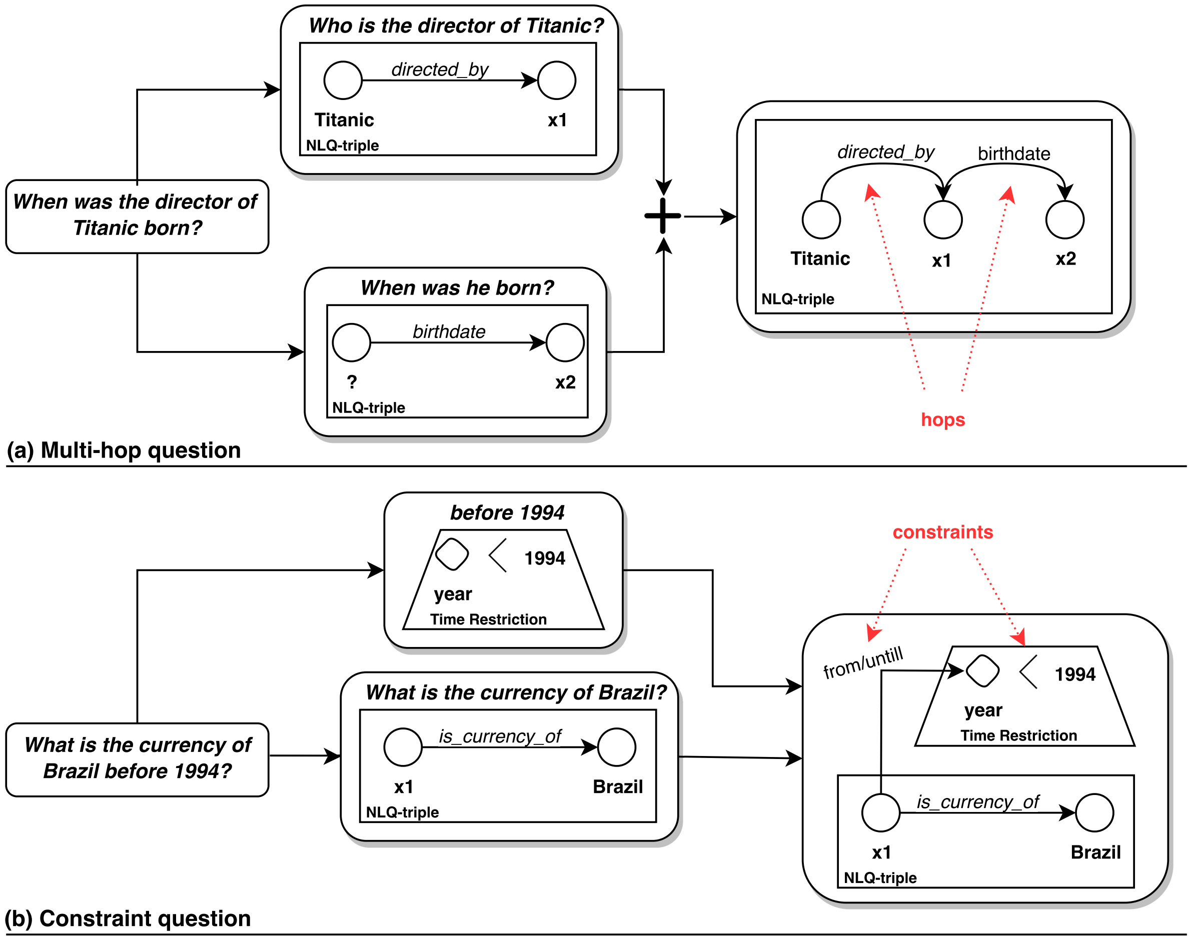
Source: Created by the author (2021)

of several types, for example, temporal *("... before 2000")*, ordinal *("The first person who ...")*, quantitative *("... having more than 5 ...")*, and others. These constraints can modify the main subjects of an NLQ and consequently change the answer. As simple QA systems target only the direct entities detected, they can not handle these slight modifications caused by these restrictions.

Articles in the multi-hop category try to solve this problem by detecting the root entities and relations and creating a list of possible candidates to hop to other relations and predicates. For the Constraint questions, question templates, constraints rules, and question decomposition are more common. Figure 4 presents an example of how the process to solve each type of complex question is done. To answer a multi-hop question first the question candidates are created, and in the next step, the join between the candidates is performed to create the hop between the entities detected (Figure 4a). Figure 4b presents an example of constraint questions where a time restriction is identified and mapped to a constraints rule added in the final NLQ-triple to restrict the query. Finally, as it is possible to see in Table 6, some works try to solve both complex types simultaneously and we observed that "multi-constraints" is another term to refer to the union of the categories listed above.

The computational cost is one of the main problems in the complex question approaches. QA systems need to handle too many triples and hops to answer some complex questions. For example, ComplexWebQuestions (TALMOR; BERANT, 2018) dataset contains questions that need at least 6-hops to generate the answer (Section **2.4.3** present more information). The high number of entity candidates (KB resources) is another problem since some algorithms take exponential processing time to process several triple connections. A module to prune and rank the best candidates is used to solve this problem.

Figure 4 – Complex question by subtype. Figure 4a and 4b presents an example of multi-hop questions and constraint questions, respectively.



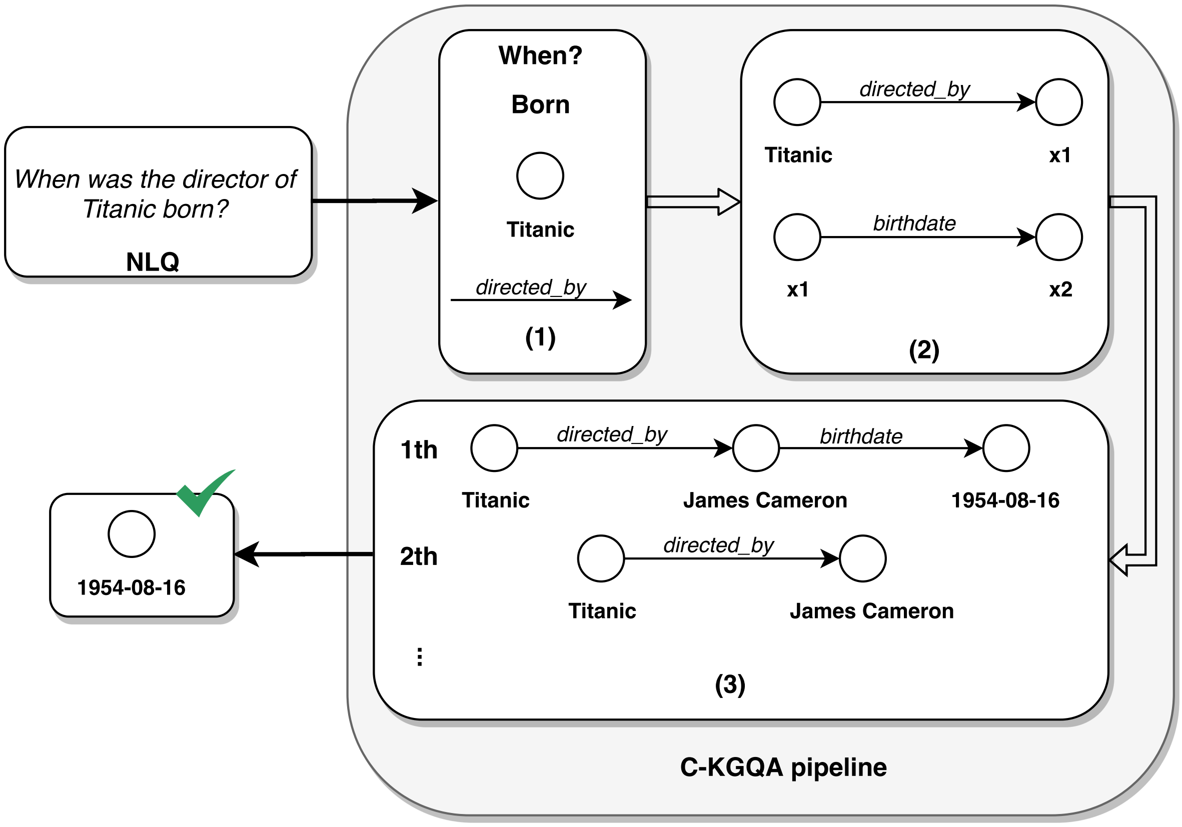
Source: Created by the author (2021)

2.4.2 RQ2 — WHAT ARE THE MOST COMMONLY USED FEATURES AND METHODS IN COMPLEX QUESTION ANSWERING?

We can generally divide the C-KBQA pipeline into three steps: question parsing, question representation, and candidate ranking. Figure 5 illustrates the steps for the question "When was the director of Titanic born?".

In the Question parsing step, the selection of the question type and identification of the main subjects are performed. First, the system finds the question types that match the NLQ, such as "when", "what", "how", among others. These question types are called *wh*-questions. Part-of-speech (POS) tagging and dependency trees are usually used to extract the sentence's grammatical structure and understand which *wh*-question the sentence represents. Also, Named Entity Recognition methods are used to locate and classify named entities in text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages.

Figure 5 – C-KBQA pipeline example for the question “When was the director of Titanic born?” as the NLQ input. In the Question parsing step (1) the type of the question and the important subjects are identified. In the Question representation (2) the map and connection of the entities and relations into a KB structure are performed. The Candidate ranking (3) is performed to select the most appropriate KB triple. Finally, the entity of the KB is the answer output.



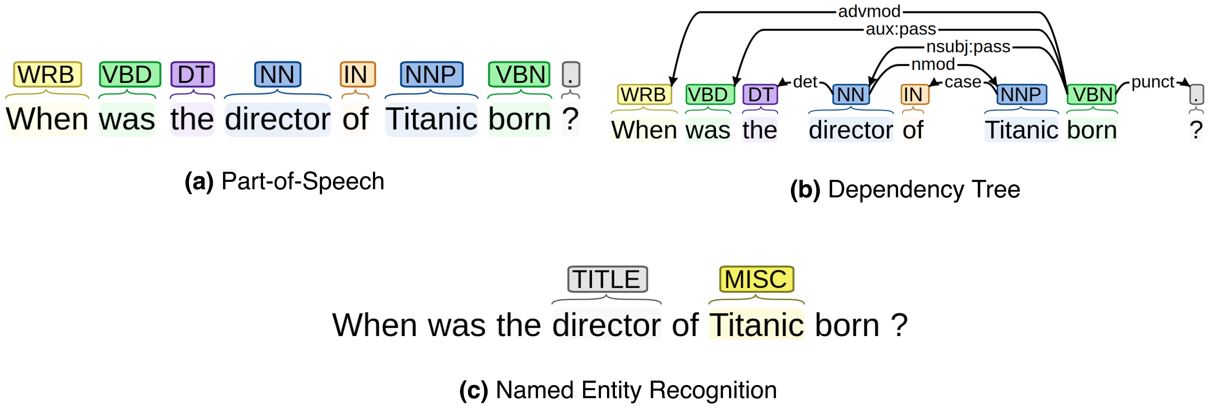
Source: Created by the author (2021)

Figure 6 shows an example of how this step is performed using the Stanford CoreNLP package[6]. Figure 6a presents the results of the POS-tagging process. We can see the identification of each word tag, e.g. NN that represents a Noun, and WRB that represents a Wh-Adverb. Besides that, in Figure 6b the dependency tree indicates the relations between every linguistic unit, or word, of the sentence, using directed arcs in a typed dependency structure. The relationship between any two words is marked by a dependency tag where one acts as the head and the other is the dependent. In the example, there is a dependency from “born” to “When” where “born” is the head and “When” is the child or dependent. It is denoted by advmod which represents a adverbial modifier[7]. Finally, Figure 6c shows the tags resulting from the NER process, where the words “director” and “Titanic” are classified as TITLE and MISC, respectively.

[6] We used the web API of Stanford CoreNLP package in version 4.0.0 (updated 2020-04-16). It can be accessed at <https://corenlp.run/>

[7] <https://universaldependencies.org/u/dep/advmod.html>

Figure 6 – Example of the question parsing step for the question “When was the director of Titanic born?”.



(a) Part-of-Speech

(b) Dependency Tree

(c) Named Entity Recognition

Source: Created by the author (2021)

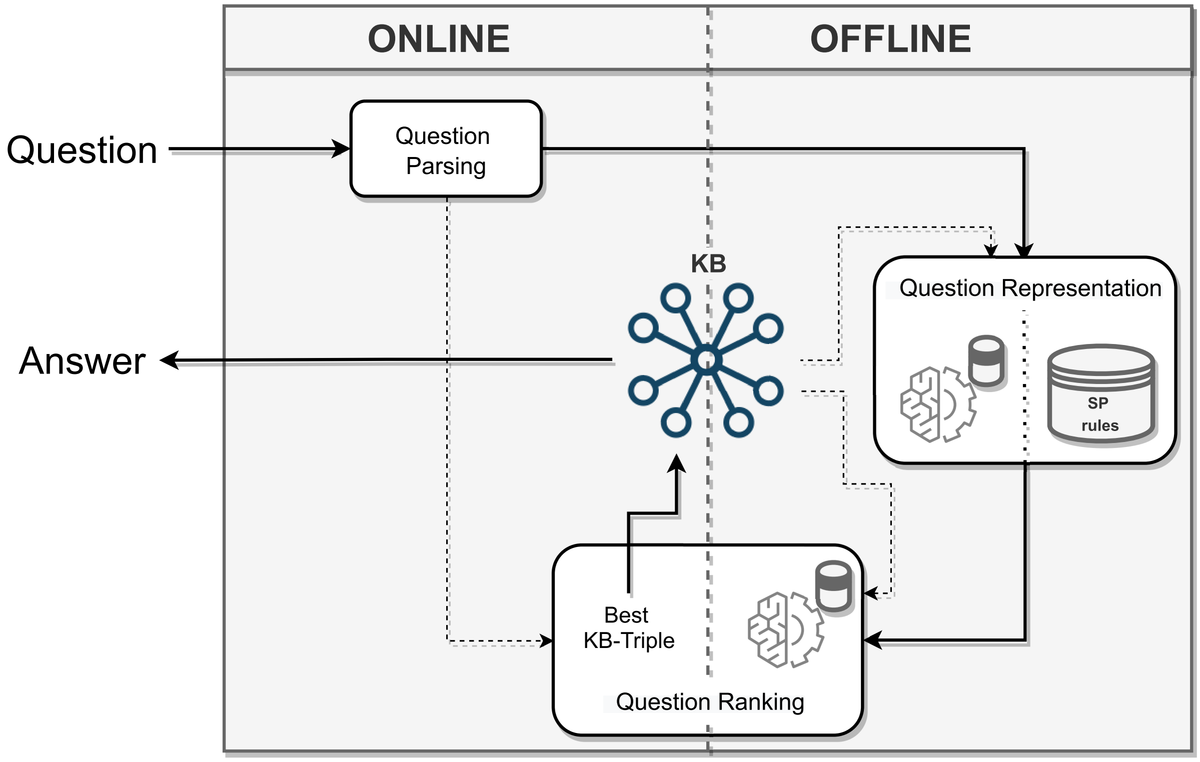
In the Question representation step, semantic mapping is performed. After question parsing, the C-KBQA system has the entities and structure extracted from a C-NLQ. Next, it is necessary to map and connect the entities and relations identified to match the KB structure. C-KBQA research follows two paths: Semantic Parsing-based and Neural Network-based approaches. Semantic parsing-based (or rule-based) approaches map the questions and the extracted information into a set of logical forms to further be transformed into KB triple query templates. Neural network-based (or rule-free) approaches use neural networks to automatically identify the types of questions and what are the most appropriate query patterns to get the answer. Both approaches can create a set of candidates that can be considered as the final answer. Section *2.4.2.1* shows each of these approaches in detail. In this step, Named Entity Recognition and Disambiguation (NERD) and relation extraction methods are performed to link a reference within a unit of text to its corresponding entity in some knowledge base, such as a node in a knowledge graph. DBpedia Spotlight (MENDES et al., 2011), S-MART (YANG; CHANG, 2015), FALCON (SAKOR; SINGH; VIDAL, 2019; SAKOR et al., 2020), and Stanford Named Entity Recognizer[8] are examples of tools used in most works.

After semantic representation and candidate generation, the candidate ranking is performed. The goal of the candidate ranking step is to remove the incorrect answers based on the type and semantics of the original C-NLQ. The best candidate is selected based on an evaluation function. For this, some works use similarity metrics such as cosine similarity or a log-likelihood function. In other cases, a machine learning model is trained, e.g., Logistic Regression or a Support Vector Machine, to collect the patterns and classify the answer candidates into a list of the best results (DIEFENBACH et al., 2020; HU et al., 2017).

[8] <https://nlp.stanford.edu/software/CRF-NER.html>

The steps above are performed in two phases. In Figure 7 it is presented the execution of each C-KBQA pipeline divided into online and offline phases. The solid arrows represent the main flow, and the dotted arrow represents the interaction between the steps and the KB. The offline phase focuses on generating the materials to be used at the online phase and does not interact with users. The question representation and candidate ranking steps are performed offline, where the rules set are defined, or machine learning models are trained. In the online phase, the C-NLQ is submitted by users, and question parsing, question representation, and candidate ranking are executed to retrieve the answer from the KB, using the models generated in the offline step.

Figure 7 – Online and offline phases.



Source: Created by the author (2021)

2.4.2.1 QUESTION REPRESENTATION AND CANDIDATE GENERATION

The question representation and candidate generation approaches for C-KBQA can be divided into two types: semantic parsing-based and neural network-based approaches. We found 47 works using semantic parsing and 31 works using neural networks. Table 7 presents the papers in each category.

Semantic parsing is the mapping of NLQ to a meaning representation or equivalent semantic structure that represents the semantics of a question (WU; WU; ZHANG, 2019; TONG; ZHANG; YAO, 2019). In this process, the NLQ is transformed into an intermediate

Table 7 – List of papers by approaches for question representation and candidate generation.

Approaches	Paper ID	# Papers
Semantic Parsing	1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 41, 43, 45, 46, 47, 48, 49, 50, 51, 52	47
Neural Networks	2, 5, 7, 10, 11, 14, 23, 24, 25, 27, 29, 30, 32, 33, 35, 36, 38, 39, 40, 42, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54	31

Source: Created by the author (2021)

representation that can be further represented as a logical form (e.g., SPARQL) (TRIVEDI et al., 2017). After identifying the main subjects in the question parsing step, the question is broken into pieces of information. This information is used to create intermediate representations, and divide-and-conquer solutions are generally used. Divide and conquer aims to decompose a given problem into a set of simpler subproblems to solve them easier than the composed problem. Usually, C-NLQ can be seen as a composition of multiple sub-questions. Thus the question can be decomposed into a set of simple questions to solve the simple question first and then recompose the simple representation into the complex question intermediate representation (SHIN; LEE, 2020).

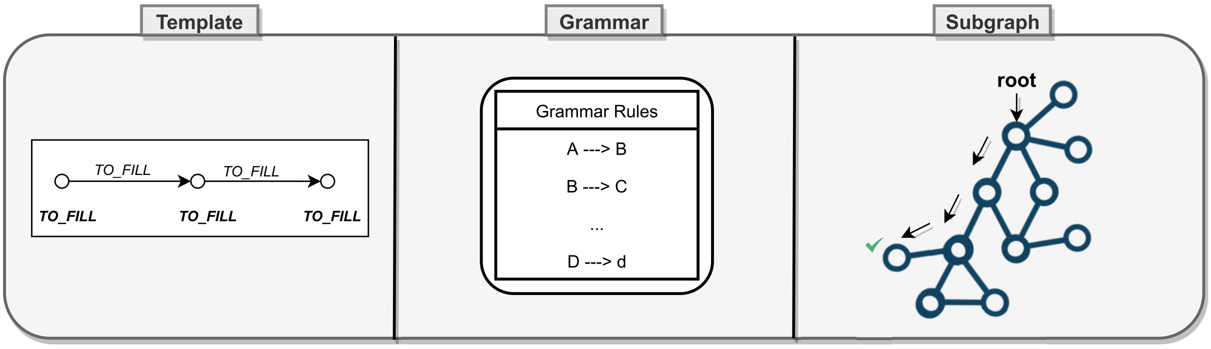
Three approaches for semantic parsing were identified (Figure 8):

- **Template-based approaches:** The answer is represented using a KB query language. A set of pre-defined query templates with slots (e.g., subject, predicate, and constraints) are available. This set of query templates considers the KB structure and covers query operations to solve multi-hop and constraint cases. The slots are filled by the system using entities and relations from C-NLQ.

- **Grammar rules and logical expressions:** It consists of a set of production rules (transition states) in the format A $\rightarrow$ b (A products b). An argument can be an entity (detected with a NERD process) or a constraint (time, ordinal, etc.), for example. Logical expressions are executed in a chain (e.g. "A $\rightarrow$ b, b $\rightarrow$ c, c $\rightarrow$ d") until no more production rule could be consumed and the answer path be reached.

- **Subgraph approximation:** Searching for paths to answer a question in the KB is usually performed using a seed entity of a KB. The seed entity is usually identified in C-NLQ. NERD approaches are used to link a reference within a unit of text to its corresponding entity of the KB. All the relations connected to the root entity in the KB are mapped and set as candidate paths to construct the subgraph. The additional information extracted in the question parsing step (e.g., relations and other entities) is used to guide the subgraph path construction and pruning. The path to answering a question is reached when there are no more candidates to be consumed. Breadth-first and Depth-first searches are commonly used to perform path candidate validation, and the exclusion of inconsistent KB paths can be used to confirm whether a subgraph path is consistent or not.

Figure 8 – Semantic parsing approaches.



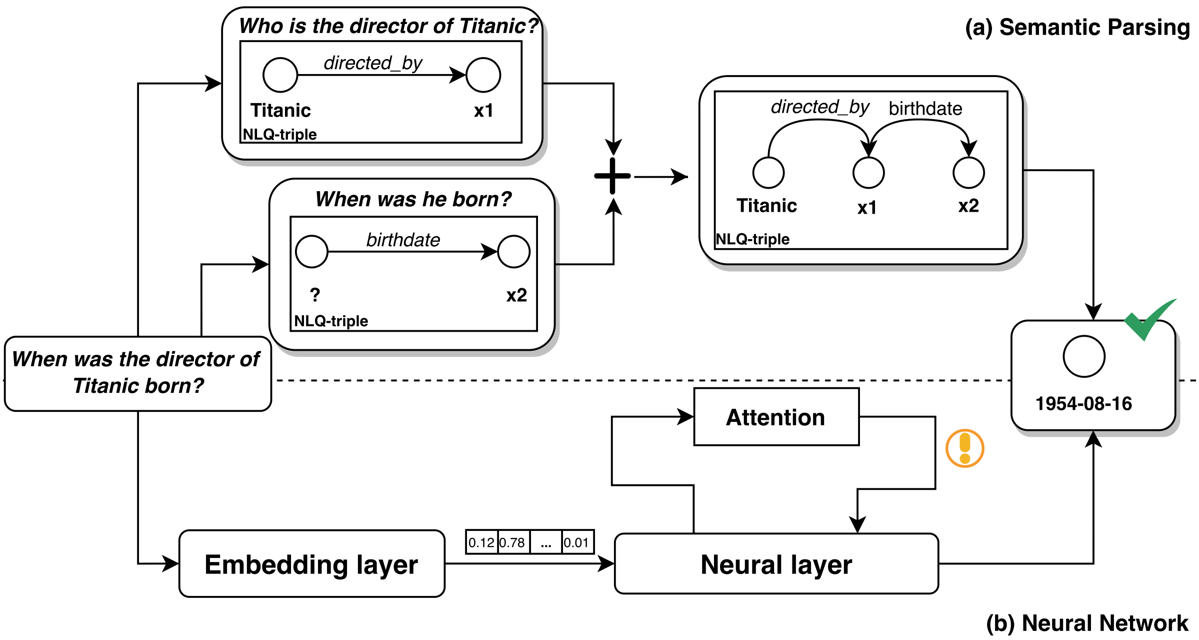
Source: Created by the author (2021)

When the NLQ matches these patterns, it is easier to query it further inside a KB. C-KBQA makes use of Semantic Parsing to map the subjects of the NLQ into semantically structured data, map the structured data inside the KB triple representation, and finally answer the question concisely. Figure 9a presents an example of the process. Initially, the question is broken into sub-questions. After the extraction of the main subject of the NLQ, it creates the intermediate representation of each sub-question. The logical form is converted to a representation consistent to retrieve information from the KB (the KB query format) in the next step. Finally, the KB triple is created, and it can be further queried in the KB schema.

This semantic mapping makes it easier to extract the relevant terms of a question and link them to KB triples. However, as it is a rule-dependent process, these approaches achieve good results if an NLQ does not match a decomposition and re-composition rule in the set of semantic rules. Problems with query scalability, when a huge number of entities and relationships is detected, increase the number of candidate rules due to the number of relations (e.g, try every possible logic form or subgraph path) (AGARWAL; RAMANATH; SHROFF, 2019).

Neural Networks-based (rule-free) approaches use neural networks architectures to encode both questions and answers into a vector space model and select the most appropriate query patterns to get the answer (LUO et al., 2018). In this way, it is possible to identify question types and common patterns to answer a certain type of question (LUO et al., 2018). The works in this category usually are composed of a word embedding layer

Figure 9 – Example of question representation and candidate generation. Figure 9a and 9b present the Semantic Parsing and the Neural Networks pipeline, respectively.



Source: Created by the author (2021)

and a neural network layer. Figure 9b presents an overview of the Neural Networks-based approach.

First, the word embedding layer is used to transform the sentence into a sequence of word vectors or sentence vectors. Word embeddings reduce the computational complexity since the matrix operations through these word vectors are fast to compute. Word2vec (MIKOLOV et al., 2013), GloVe (PENNINGTON; SOCHER; MANNING, 2014), and FastText (BOJANOWSKI et al., 2017) are examples of pre-trained word embedding used in C-KBQA literature. Second, a deep neural network is used. In this step, Recurrent Neural Networks (RNN) are the most common architecture. RNN can disseminate historical information through a sequence of neural network units. An RNN works as a chain network architecture and analyzes the present input and the previous output at each time step when processing sequential data. Thus, RNN can extract the information context propagation of a C-NLQ. RNN have also been used as encode-decode architectures (sequence to sequence). In this process, an RNN unit encodes the C-NLQ, and another RNN collects the historical information from the C-NLQ and decodes it into the answer sequence. Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) (and its variants such as bidirectional architectures) are the models most used to perform this step, since these RNN can deal better with the vanishing gradient problem.

One of the main problems of NN is the decrease in performance for longer and complex sequences. To solve this, recent works have used an attention mechanism (BHUTANI; ZHENG; JAGADISH, 2019; DING et al., 2019; TONG; ZHANG; YAO, 2019; BHUTANI et al., 2020). Attention is used to emphasize the more relevant parts of a question and to preserve the context of the sentences (BHUTANI; ZHENG; JAGADISH, 2019; BHUTANI et al., 2020). Three components are usually used when creating attention mechanisms: query, key, and value. The query is the information that you are looking for. The key is the set of assets that can be used when making a query. The value is the matched result for your query given the set of keys used. These three components are used as weight matrices, that will be learned during the training step. Each attention matrix can be constructed with linear layers and the weights of each matrix are learned during the training. Thus, the attention module learns to highlight the important information of a question.

Neural Networks-based Semantic Parsing approaches try to solve complex questions by using a combination of Semantic Parsing and Neural Networks techniques and are becoming state of the art (LUO et al., 2018; DING et al., 2019). This approach consists in training a neural network to match a set of semantic parsing rules instead of only the final answer. Thus, the model learns the semantics behind a C-NLQ instead of only learning the most appropriate query patterns to retrieve the answer. These approaches are more generalist, as they learn step-by-step how to answer a question. Usually, an encode-and-compare pipeline is applied, where both the NLQ and the subgraph sequence used to answer the NLQ are, for example, encoded as semantic vectors in common embedding space (LUO et al., 2018). The semantic similarity in the embedding space is calculated by a score between vectors (e.g. cosine similarity). Although RNN are widely used, for Neural Networks-based Semantic Parsing approaches some works also try the use of Convolutional Neural Networks (HU; ZOU; ZHANG, 2018; BAO et al., 2016), Memory Neural Networks (MILLER et al., 2016; HAO et al., 2019; SAHA et al., 2018; HUA et al., 2020b; HUA et al., 2020a), or their variants in this step. Conventional Networks (CNN) are used to perform semantic matching and calculate the similarity score between NLQ and the subgraph path between the NLQ and the expected answer exploiting both syntactic and sentential information correlations (BAO et al., 2016; HU; ZOU; ZHANG, 2018). Memory Neural Networks (MNN) are used to learn the ways to answer an NLQ with a long-term memory component and various inference components (XU et al., 2019). The most common MNN for C-KBQA systems is the Key-Value Memory Neural Network and it is able to learn a chain of logical operations in a key-value structure (MILLER et al., 2016).

2.4.3 RQ3 — WHAT ARE THE MOST USED DATASETS AND HOW ARE THEY EVALUATED?

Complex question datasets are challenging to find. To help in this process, we mapped the most-used datasets and metrics used to evaluate them. We selected the freely available datasets used for more than two works and removed all the datasets that focus on simple questions from this list. Table 8 presents the datasets and their information.

Table 8 – Databases used for C-KBQA. Read 'NP' as Number of papers using the dataset, 'L' as Logic Form available or not, and 'H' as max-hops in the dataset.

Dataset	#NP	L	KB[a]	#H[b]	#Size	Year	Metric[c]
WebQuestions (BERANT et al., 2013)	17	N	FB	-	5.8K	2013	Ac,P,R,F1
QALD[d]	12	Y	DB	3[e]	500	2011-	P,R,F1,T
ComplexQuestions (BAO et al., 2016)	11	N	FB	-	2.1K	2016	Ac,P,R,F1
WebQuestionsSP (YIH et al., 2016)	7	Y	FB	2	4.7K	2016	Ac,F1
LC-QuAD-1 (TRIVEDI et al., 2017)	7	Y	DB	2	5K	2017	P,R,F1,T
CSQA/CQA (SAHA et al., 2018)	6	N	WD	-	200K	2018	R,P,F1
ComplexWebQuestions (TALMOR; BERANT, 2018)	3	Y	FB	6	34K	2018	P,F1
PathQuestion (ZHOU; HUANG; ZHU, 2018)	2	Y	FB	2/3	1K∼5K	2018	Ac,P
WordCup2014 (ZHANG; WINN; TOMIOKA, 2016)	2	Y	SD	1/2	1.4K∼6K	2016	Ac,P

[a] The KB are: Freebase (FB) (BOLLACKER et al., 2008), DBpedia (DB) (AUER et al., 2007), Wikidata (WD) (VRANDEČIĆ; KRÖTZSCH, 2014), and Specific domain (SD).
[b] Information collected through the dataset description or looking at the logical representation of each dataset.
[c] The metrics are: Ac = Accuracy, P = Precision, R = Recall, F1 = F1-score, and T = Time.
[d] <http://qald.aksw.org/>
[e] Extracted from the last available version, QALD-9.

Source: Created by the author (2021)

WebQuestions was made available by (BERANT et al., 2013). The questions were collected by Google Suggest API. The question "Where was Barack Obama born?" was used as seed, and all the following *wh*-questions were collected using breadth-first searching. Amazon Mechanical Turk (AMT), a crowdsourcing platform, was used to evaluate the question retrieved. However, the first version of Web Questions does not have any logic form to be matched, and the WebQuestions Semantic Parsing (WebQuestionsSP) was released to fill this gap (YIH et al., 2016). WebQuestionsSP contains questions in SPARQL format. However, this dataset contains fewer available questions than the original WebQuestions because some questions were removed to avoid ambiguity. Even so, WebQuestions is the most used dataset for C-KBQA, but only 15% of the dataset is composed of complex questions (DING et al., 2019). Both WebQuestions and WebQuestionsSP use Freebase as Knowledge Base.

Question Answering over Linked Data (QALD) is an annual challenge available in the 9th version. The QALD versions contain only between 100 and 500 questions each; however, most questions require multiple entities and predicates to be answered. The questions in QALD are also presented in SPARQL format, and the datasets use DBpedia as Knowledge Base.

ComplexQuestions (BAO et al., 2016) was released to fill the lack of constraint questions. The dataset contains 597 questions from WebQuestions and the 300 questions used by (YIN et al., 2015) as part of it. The authors filled the rest of the dataset with 878 constraint questions manually labeled. ComplexWebQuestions (TALMOR; BERANT, 2018) applies a process similar to the creation of ComplexQuestions but using the WebQuestionsSP. The authors modified the logic form of WebQuestionsSP and added more constraint types to the original questions. The AMT platform was also used to aid in the dataset evaluation. These new types are composed of time, conjunctions, superlatives, and comparative constraints. ComplexQuestions do not have a logical form and ComplexWebQuestions uses SPARQL as the logic form. Both datasets use Freebase as Knowledge Base.

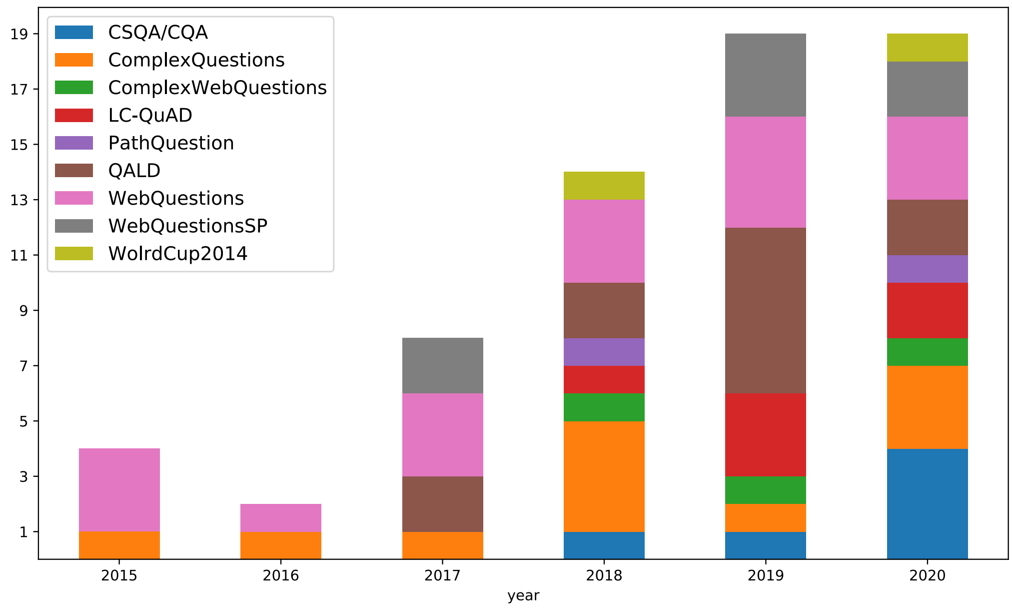
The Large-Scale Complex Question Answering Dataset version 1 (LC-QuAD-1) (TRIVEDI et al., 2017) contains 5000 questions composed of 5042 entities and 615 predicates. Only 18% of the questions are simple questions. The questions were created from a list of template questions. The authors used 38 unique templates and separated the dataset into three major types: entity (a KB object), boolean (true or false), and count (aggregated) questions. The questions in LC-QuAD are presented in the SPARQL format, and the dataset uses DBpedia as Knowledge Base. Besides that, a new version of LC-QuAD was released in 2019 (LC-QuAD-2 (DUBEY et al., 2019)) with 30K questions. LC-QuAD-2 included more types of complex questions, especially constraint questions. LC-QuAD-2 uses both DBpedia and Wikidata as Knowledge Bases.

Complex Sequential Question Answering (CSQA) was created to be used in dialog systems, where it is necessary to look at the historical information chain to get the final answer. The authors also released a new version of the dataset where the questions can be answered without needing previous dialog context, called Complex Question Answering (CQA). The questions on this dataset are composed of seven groups: Simple, Logical, Quantitative, Comparative, Verification, Quantitative count, and Comparative count (HUA et al., 2020b; HUA et al., 2020a). CSQA and CQA do not have a logical form.

PathQuestion (ZHOU; HUANG; ZHU, 2018) is composed of two datasets: PathQuestion (PQ) and the PathQuestion-Large (PQL). PQL is more challenging than PQ. These two are created from subsets of Freebase. PathQuestion contains two sub-types of dataset: "2H" present 2-hop questions and "3H" that present 3-hop questions (QIU et al., 2020a). The questions in PathQuestion are present in the form of paths between entities, e.g. ($e_s \xrightarrow{r1} e_s \xrightarrow{r2} answer$) denotes a 2H example. Finally, WorldCup2014 (ZHANG; WINN; TOMIOKA, 2016) is a specific domain dataset based in a football domain of the participants of FIFA World Cup 2014. The dataset is composed of three question subtypes: single-relation questions (1H), two-hop questions (2H), and conjunctive questions (constraint questions) (ZHOU; HUANG; ZHU, 2018). The questions in WorldCup2014 are also represented as a path between entities.

It is worth noting that some works can use more than one dataset to evaluate their systems. Furthermore, it is common to add simple question datasets (e.g. SimpleQuestions (BORDES et al., 2015), Free917 (CAI; YATES, 2013) or WikiMovies (MILLER et al., 2016)) to evaluate their systems in order to show that their works are also able to answer simple questions. Finally, Figure 10 presents the distribution of use of each dataset from Table 8 over the years.

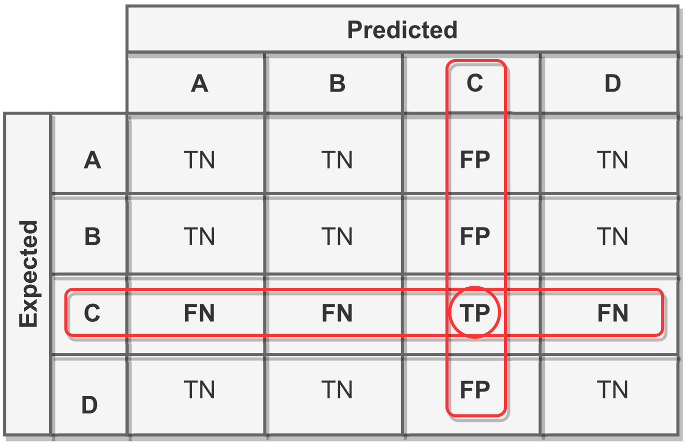
Figure 10 – Most used datasets over the years



Source: Created by the author (2021)

The template-based matching problem is treated as a classification problem where the classes are the desired templates. For logical expressions and subgraph approximation, the evaluation metrics are not directly related to the multiclass classification, but rather to the assessment of the relevance of the recovered entities and the order that they appears. The evaluation metrics are based on the confusion matrix (excluding the processing time). The confusion matrix computes the successes and errors in a classification problem and is divided into expected and predicted values. The expected values are the instances in the evaluation dataset for the QA systems, and the predicted values are the candidate answers selected by a C-KBQA system. True Positive (TP) indicates the instances that were predicted corrected (it was predicted positive and it is true). True Negative (TN) indicates the instances that do not have to be predicted as expected and it is correct (it was predicted negative and it is true). False Positive (FP) indicates the instances that were predicted wrongly as corrected (it was predicted positive and it is false). False Negative (FN) indicates the instances that do not have to be predicted as expected and it is wrong (it was predicted negative and it is false). In Figure 11 it is presented an example of a confusion matrix for a multiclass classification problem.

Figure 11 – Confusion Matrix regarding the class C.



Source: Created by the author (2021)

The most used evaluation metrics are Accuracy, Precision, Recall, F1-score, and Processing Time, as shown in Table 8. Accuracy is used to calculate the percentage of questions that were correctly answered by a C-KBQA system. Precision is the ratio of the correctly predicted answers and the number of predicted answers. Recall is the ratio of the correctly predicted answers and the number of expected answers. F1-score is the harmonic mean between precision and recall. Finally, Processing Time is performed to compare the time performance among systems and can be performed in two ways: only the evaluation of the new approaches/module (e.g. a new way to perform the searching paths to answer a question) or an end-to-end evaluation where all the C-KBQA system steps are evaluated from the input question to the output answer.

Macro and micro average are the two common ways to average the precision, recall, and F1-score metrics. The macro average computes the evaluation metric independent for each class, considering all classes equal (e.g, without taking into account the imbalanced scenario). The micro average computes the average for each class, using the frequency and the proportion of each class as variables to compute the average (better to imbalanced scenarios).

With N being the number of questions in a dataset and C the possible answer for a question in the evaluation dataset, the equation of each evaluation metric for is:

$$Accuracy = \frac{1}{N}\sum_{c \in C} TP_c \tag{2.1}$$

$$MacroPrecision = \frac{1}{C}\sum_{c \in C}(\frac{TP_c}{TP_c + FP_c}) \tag{2.2}$$

$$MacroRecall = \frac{1}{C}\sum_{c \in C}(\frac{TP_c}{TP_c + FN_c}) \tag{2.3}$$

$$MacroF1 = 2 * (\frac{MacroPrecision * MacroRecall}{MacroPrecision + MacroRecall}) \tag{2.4}$$

$$MicroPrecision = \frac{\sum_{c \in C} TP_c}{\sum_{c \in C}(TP_c + FP_c)} \tag{2.5}$$

$$MicroRecall = \frac{\sum_{c \in C} TP_c}{\sum_{c \in C}(TP_c + FN_c)} \tag{2.6}$$

$$MicroF1 = 2 * (\frac{MicroPrecision * MicroRecall}{MicroPrecision + MicroRecall}) \tag{2.7}$$

Most used metrics for KBQA end up being Information Retrieval metrics instead of the metrics usually used in other QA subareas such as BLEU, ROUGE, etc (CHEN et al., 2019). It is expected that the systems start to use these metrics to evaluate their approaches with the advancement of research in C-KBQA systems and Natural Answer Generation (LI; HU; ZOU, 2020).

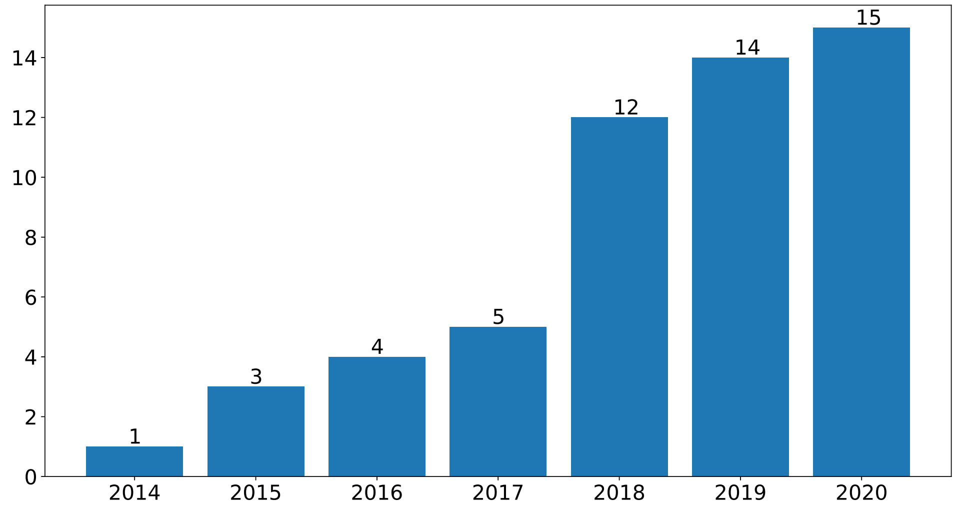
2.4.4 RQ4 — HOW THE WORKS HAVE BEEN PUBLISHED OVER THE YEARS?

The C-KBQA problem started to be addressed from 2014. There was an increase in the number of works that were published, showing the interest in this field. Figure 12 presents the number of publications by year. Also, we listed the most cited papers to show the most important works in the area. Table 9 presents the most cited papers. We discuss the main approaches in this section, but detailed information on all works is available in the Github repository[9].

Several approaches have been explored for C-KBQA systems over the years. From 2014 to 2015, the first mapped work presented an ontology-based QA system for Chinese delicacy (YIN; GE; WANG, 2014). The authors proposed a novel query triple representation

[9] TSV files available at https://github.com/lapic-ufjf/CKBQA-systematic-mapping-2021

Figure 12 – Number of publications by year



Source: Created by the author (2021)

to model complex questions with multiple hops and constraints. A method for transforming questions into query triple representation was presented and a set of semantic parsing logical rules was created to map a complex question to a KB triple. Yin et al. (2015) presented a study on complex semantic constraints (e.g. prepositional or adverbial phrases) and the use of n-tuple assertions. The n-tuple assertions are assertions with an arbitrary number of arguments and work as a set of grammar rules to create an intermediate format of KB tuple queries. Bast e Haussmann (2015) presented a template-based system, called AQQU that maps the questions to a set of predefined templates. In their approach, the entities from the KB mentioned in the question are identified and, thus, a set of candidate query templates are matched to the question. A ranking model was used to select the best template. However, AQQU has a limited cover of templates for complex questions: only three templates were presented. Yih et al. (2015) propose a novel semantic parsing framework called Staged Query Graph Generation (STAGG). STAGG presents a solution for the subgraph approximation problem. First, the authors defined a query graph that can be directly mapped to a logical form and, next, the logical form is reduced to query graph generation to retrieve the answer from the KB. A CNN is used to score all the possible paths and a log-linear model is used to reward the system (i.e. calculates the likelihood that a query graph was correctly parsed and rank it).

From 2016 to 2017, most of the new approaches to focused in improve the limitations to answer multi-hop questions. Bao et al. (2016) presented that STAGG has a limitation to cover questions with multiple constraints. The authors improved the basic constraint question covered for STAGG presenting a Multi-Constraint Query Graph (MulCG). MulCG uses the same basic query graph defined for STAGG but iteratively adds new subgraphs to cover more constraints (i.e new logical rules for constraints questions were added to the system). Abujabal et al. (2017) present QUINT, a system that automatically generates templates from user questions paired with their answers. QUINT came as a novel solution for removing predefined handcraft templates. A complex question can be answered even without having any templates for the entire question exploring the natural language composition (using dependency parse tree) and templates learned from simple questions. Finally, Miller et al. (2016) presented the first use o Memory Networks for KBQA systems and proposed the Key Value-Memory network (KV-MemNN). The KV-MemNN is used to learn a chain of logical operations in a key-value structure.

From 2018 ahead, the use of Semantic Parsing-based and Neural Network-based approaches are more present. Jia et al. (2018) presented a novel module capable to deal with constraint questions. The work proposed a new module, called TEQUILA, that addresses complex temporal questions. TEQUILA reused and improved other KBQA systems like AQQU (BAST; HAUSSMANN, 2015) and QUINT (ABUJABAL et al., 2017). Both AQQU and QUINT handle simple questions using a template-based approach, however, the systems have limited coverage for complex questions. TEQUILA came out as a side module to improve the answer to temporal questions using decomposition and re-composition rules. Talmor e Berant (2018) proposed a novel framework that decomposes complex questions into a sequence of simple questions. The final KB triple is computed from the sequence of simple question answers using a set of semantic parsing rules. The authors identified the need for better datasets for C-KBQA and released a novel dataset with complex questions.

In its turn, Bhutani, Zheng e Jagadish (2019) also used a decomposition technique in a system called TextRay, which searches for sub-graphs through a combination of Semantic Parsing and Neural Networks methods. Two LSTM and an attention mechanism were used in the semantic matching model to extract the semantic relation between the NLQ and the KB entities and relations. A new version of TextRay is presented in Bhutani et al. (2020), where the system is able to search for answers in multiple KB with an additional module that computes the similarity of entities and relations among multiples KB. In the same way, the work of Diefenbach et al. (2020) also explored the difficulties of Semantic Parsing approaches to NLQs for multiples KB and multiple languages KB. Zafar, Napolitano e Lehmann (2018) presented a semantic parsing module for query generation. The so-called SPARQL Query Generator (SQG) detects subgraphs in the KB using Named Entity Disambiguation and Relation Extraction methods. However, the SQG has a limited cover of complex question types (questions with ordinal and filter restrictions). To improve the limitations of SQG, Abdelkawi et al. (2019) extended the SQG and created two extra modules to better handle questions with ordinal and filter restrictions (constraint questions). Finally, with the advancement of transference learning and deep neural architectures, Transformers were first used in a template-based approach in Evseev e Arkhipov (2020). A BERT for classification was used to perform template matching and to determine the template type of a complex question.

Table 9 – Most cited articles

Article	#Cited by[a]	#Papers
(MILLER et al., 2016; YIH et al., 2015)	>400	2
(LIANG et al., 2017)	200 - 400	1
(XU et al., 2016; BAST; HAUSSMANN, 2015)	150 - 199	2
(ZHANG et al., 2018; CUI et al., 2017; ABUJABAL et al., 2017; YU et al., 2017; TALMOR; BERANT, 2018)	100 - 149	5
(HU et al., 2017; BAO et al., 2016; SAHA et al., 2018; YIN et al., 2015)	50 - 99	4
(ZHENG et al., 2018; DIEFENBACH et al., 2020; LUO et al., 2018; ZHOU; HUANG; ZHU, 2018; ZAFAR; NAPOLITANO; LEHMANN, 2018; HU; ZOU; ZHANG, 2018)	20 - 49	6
(MAHESHWARI et al., 2019; JIA et al., 2018; LU et al., 2019; XU et al., 2019; SAHA et al., 2019; VAKULENKO et al., 2019)	10 - 19	6
(BHUTANI; ZHENG; JAGADISH, 2019; JIN et al., 2019; TONG; ZHANG; YAO, 2019; QIU et al., 2020a; BAKHSHI et al., 2020; DING et al., 2019)	5 - 9	6
(RADOEV et al., 2018; HONG et al., 2016; ABDELKAWI et al., 2019; YIN; GE; WANG, 2014; HAO et al., 2019; WANG et al., 2019; AGARWAL; RAMANATH; SHROFF, 2019; BHUTANI et al., 2020; HUA et al., 2020c)	1 - 4	9
Others	0	13

[a] Extract from Google Scholar 'Cited by' score on November 24, 2020.

Source: Created by the author (2021)

The works discussed previously were published in important venues. We ranked the publication venues individually separated by the number of articles present in this mapping. Table 10 presents the venue types and Table 11 presents the ranked list of publication venues (we selected the venues where two or more papers were published).

Table 10 – Publishing Venues Types

Venue	#Papers[a]
Conference (C)	42
Journal (J)	13
Workshop (W)	1

[a] Some conferences occur at the same time. Due to this fact, the total of venues is bigger than the number of papers mapped.

Source: Created by the author (2021)

In the Table 10 it is presented the number of articles published in journals, conferences, and workshops ranging from 2014 to 2020. The vast majority of the articles

Table 11 – Most Popular Journal and Conference

Name	Type	#Papers	H5-index[a]
ACM International Conference on Information and Knowledge Management (CIKM)	C	6	54
Conference on Empirical Methods in Natural Language Processing (EMNLP)[b]	C	5	112
Meeting of the Association for Computational Linguistics (ACL)[c]	C	4	135
International Conference on Computational Linguistics (COLING)	C	2	49
International Joint Conference on Artificial Intelligence (IJCAI)	C	2	95
The VLDB Endowment	J	2	116[d]
Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)	C	2	90
AAAI Conference on Artificial Intelligence (AAAI)	C	2	126
Asia-Pacific Web and Web-Age Information Management Joint International Conference on Web and Big Data (APWeb-WAIM)	C	2	11
International Joint Conference on Natural Language Processing (IJCNLP)[b,c]	C	2	24

[a] Extract from Google Scholar h5-index score at <https://scholar.google.com.br/citations?view_op=top_venues> on November 24, 2020.
[b,c] IJCNLP is a biennial flagship conference of AFNLP and took place in conjunction with the ACL or EMNLP.
[d] This is the only score not mapped by Google Scholar. So, we extracted the score from Scimago Journal & Country Rank (SJR) at <https://www.scimagojr.com/>.

Source: Created by the author (2021)

were published in Conferences (approximately 75% of the articles). As stated in Meyer et al (MEYER et al., 2009), conferences are of great importance in Computer Science and with high acceptance rates. As shown in Table 11, the ACM International Conference on Information and Knowledge Management is the conference with the largest number of publications in our mapping, corresponding to 10% of the total articles. Soon afterwards, the Empirical Methods in Natural Language Processing took place, the conference had six publications in our mapping, all between 2016 and 2020, the time of greatest growth of publications in the area of complex questions. Different techniques were presented in these conferences such as template-based approaches, candidate entity ranking, and path generation and ranking on knowledge graphs. Finally, Neural Networks-based approaches were very frequent in ACL articles.

2.5 THREATS TO VALIDITY

This systematic literature mapping aimed to present an overview of Complex Knowledge Base Question Answering systems. However, there are threats to its validity and limitations, like any research method. Certain uncontrollable limitations may have influenced the results of this study. There might be bias regarding the number of researchers selecting the papers. Despite reviewing the overall process and aiming to mitigate this threat to validity, two researchers (first and second authors) were able to reproduce this process to reduce the possibility of bias.

Removing articles not written in English and those in gray literature, for example, can diminish the accuracy of the conclusions, even though the mapping covered 54 articles. Also, some exclusion criteria could be more flexible. However, this mapping focused on articles that explained the main process of C-KBQA systems and was discussed in detail, even though some influential works in the area may have been lost during the selection process. Besides that, all the process was created to be reproducible. So it is easier for any new researcher who wants to extend it if needed.

Additionally, errors can be inserted in the protocol definition and the search string may not contain all the relevant keywords, causing the loss of some valuable studies. To mitigate this, other researchers reviewed the mapping planning presented in Section 2.3. The search string was evaluated using articles to control the results. We also performed the snowballing step to ensure that some works that were not covered by the search string could be collected. The papers appeared in the results generating evidence about the search string correctness. We performed one-level forward snowballing (citation tracking) and backward snowballing (reference tracking). It is expected that an in-depth searching with more levels may add new references to our final list, however, this process is time-consuming and the one-level forward snowballing is more effective (WOHLIN, 2014; GREENHALGH; PEACOCK, 2005).

Furthermore, we did not consider all the relevant electronic databases, e.g., ACM Digital Library. So, it is possible that relevant studies were not indexed in the selection of this mapping. However, this research relies on the representativeness of the repositories selected to answer the research questions. Besides the relevant electronic databases as SCOPUS e IEEE, we also considered Google Scholar to mitigate that relevant studies that were not indexed in our selection. Google Scholar presents a good coverage of papers; however, it is not the best option to be used alone to systematic mapping (GUSENBAUER; HADDAWAY, 2020). We believe that the selected electronic databases and Google Scholar were enough to obtain a big picture of the C-KBQA research area.

2.6 TRENDS AND CHALLENGES

We show an overview of the C-KBQA area and how it has been addressed. C-KBQA is a challenging area and there is a number of issues to be addressed. In this section, we present some gaps and future steps to future researchers in this field.

Recurrent Neural Networks have become well-used in this field. However, recent works show that pre-trained models and transfer learning are new options to faster train new solutions. In 2019, (LUKOVNIKOV; FISCHER; LEHMANN, 2019) showed that pre-trained models like BERT achieve good results to answer simple questions. The authors state that those models could have a bigger impact on C-KBQA. However, only one paper tried to explore those deep neural architectures for KBQA (EVSEEV; ARKHIPOV, 2020). Advances in C-KBQA and Deep learning can create architectures that demand less memory and have a lower training cost, allowing a popularization of the KBQA models for specific domains. However, deep learning approaches need a lot of data to generalize the training model. The creation of a big C-KBQA dataset is costly since it needs to first collect the data, create the logic form for each question, and validate it on KB.

However, how to ensure that the information contained in the KB is updated and the system can deal with actual questions? Since the evaluation dataset is created at a fixed KB version, the system learns to answer the question related to the train and test dataset. It is necessary to put these systems against real users and also evaluate them in real scenarios. One option to do this is to test these systems using a crowdsourcing platform and evaluate their usability. QA systems can make use of the last advances in cross-language approaches applied to KB (SCHUMACHER; MAYFIELD; DREDZE, 2020), disambiguation (KARTSAKLIS; PILEHVAR; COLLIER, 2018), faster or better methods for graph knowledge searching (Yang et al., 2016; NAMAKI et al., 2017), embedding (JOULIN et al., 2017; DETTMERS et al., 2018), and reasoning (TRAN et al., 2020; CHEN; JIA; XIANG, 2020). Moreover, new natural language understanding techniques can improve question-answering systems, such as advances in handling noisy questions (ZHANG et al., 2018), new dependency parsers and semantic role labeling methods (SHI; LIN, 2019; YANG et al., 2020; ANDERSON; GóMEZ-RODRíGUEZ, 2020; HAN et al., 2020; ZHENG; KORDJAMSHIDI, 2020), and common sense reasoning (LIN et al., 2019; BOSSELUT; BRAS; CHOI, 2020; LIU et al., 2020).

Question-answering systems and conversational agents are closely related. The context of the search and the use of historical data is important to achieve more accurate answers and a better user experience. There is a large number of papers discussing advances in research on conversational agents (RAMESH et al., 2017; NAGARHALLI; VAZE; RANA, 2020; PINXTEREN; PLUYMAEKERS; LEMMINK, 2020). Also, the popularity of semantic web standards, such as RDFa[10], will substantially increase the

[10] https://www.w3.org/TR/rdfa-primer/

amount of data present in the knowledge graph, thus allowing systems to be less KB-dependent and outdated.

2.7 CONCLUDING REMARKS

This paper presented a systematic mapping of Complex Knowledge Base Question Answering (C-KBQA) systems. A protocol was adopted for the execution of systematic mapping to reduce the bias and make the study reproducible for other researchers. A collection of 54 papers were systematically selected from a total of 894 papers. The identification of the most used methods, datasets, knowledge bases, metrics, and domains in the complex question answering scenario in the literature was presented and the main gaps for C-KBQA systems were discussed. Finally, we presented the future directions and the main gaps for C-KBQA systems.

We show that C-KBQA systems try to handle two types of complex questions: Multi-hop and Constraint questions. Also, we present an overview of the process to construct C-KBQA systems and how the main approaches are performed. In this mapping, we notice that the papers try to use two main approaches: Semantic Parsing and Neural Networks-based. However, in recent years, the combination of these two approaches has become state of the art (called Neural Network-based Semantic Parsing).

Good datasets for C-KBQA are still an open challenge. We show that the dataset most used is the WebQuestions, however, this dataset has few complex questions. Other datasets came to fill this gap (e.g. CSQA/CQA, LC-QuAD, and ComplexQuestions), but few C-KBQA systems are using them. We also noted that the evaluation metrics for KBQA are still using information retrieval metrics, like F1 and Accuracy. However, it is expected that the authors change their evaluation approaches with the advancement of research in C-KBQA systems and Natural Answer Generation to metrics like BLEU and ROUGE.

Finally, the overview of the publication and venues shows that the C-KBQA area is receiving huge attention from researchers, and great advances are expected in a short time. Also, we made all the data used in this review available for use in future mappings.

3 MANUSCRIPT 2: PROPOSED APPROACH

Currently, KBQA systems achieve better results when answering simple questions, and Complex Knowledge Base Question Answering (C-KBQA) systems turned the goal to the recent researches. In this master's thesis, the main objective of this chapter is to provide a novel model solution for C-KBQA systems. This chapter is a full article[1] and presents a template matching approach for C-KBQA systems using the combination of Semantic Parsing and Neural Networks techniques to determine the answer templates that a complex question fits. Moreover, an attention mechanism was created to assist the neural network in selecting the most important information. In the so-called Hereditary Attention, each neural network cell inherits the attention from another neural network cell, in a bottom-up way. Furthermore, we presented the problems found in C-KBQA datasets and released a new cleaned version of an LC-QuAD 2.0 dataset containing answer templates. We call this new version LC-QuAD 2.1.

3.1 INTRODUCTION

Question Answering (QA) systems have the purpose to retrieve the most relevant information (answer) to a search question made by a user (CROFT; METZLER; STROHMAN, 2010). Knowledge base (KB) is a data model based on a semantic network, which uses a triple format (subject, predicate, object) to represent and relate the information contained within a data domain (POPPING, 2003; JI et al., 2020). QA systems that made use of KB are called Knowledge Base Question Answering (KBQA). The KBQA systems use those semantic structures, for example, Freebase (BOLLACKER; COOK; TUFTS, 2007), Wikipedia (LEHMANN et al., 2015) or Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014) to answer a question.

KBQA systems need to deal with different kinds of questions. We can divide them into two groups: simple and complex questions. Simple questions are those that contain direct answers and only direct entities that need to be detected to answer a question (BORDES et al., 2015). Complex questions need more information than the explicit features that can be extracted from simple questions. It is necessary to use advanced query operations to collect the answer from the KB, such as exploiting indirect relations among entities, multi-relations, qualitative and quantitative constraints, and others (BAO et al., 2016; QIU et al., 2020a). However, it is hard to extract and map the features of a complex question into a KB since the questions have indirect relations, qualitative information, and many entities/predicates. Currently, KBQA systems achieve better results when answering simple questions, and Complex Knowledge Base Question Answering (C-KBQA) systems

[1] **Title**: A Hereditary Attentive Template-based Approach for Complex Knowledge Base Question Answering Systems

turned the goal to the recent researches in the QA field (QIU et al., 2020b; HUA et al., 2020b; HUA et al., 2020a).

The extraction of the features of a question and the mapping into a KB is called semantic parsing. Semantic parsing is the mapping of Natural Language Question (NLQ) to a meaning representation that can be further represented as a logic form (TONG; ZHANG; YAO, 2019). In this process, the NLQ is transformed into an intermediate format that can represent the structure of the question (TRIVEDI et al., 2017; WU; WU; ZHANG, 2019). Similar to the divide and conquer problem, it is possible to break an NLQ into a list of intermediate representations. However, the use of semantic parsing alone on a complex question can be computationally expensive due to the number of operations that must be performed to find the structure that answers a question (HÖFFNER et al., 2017).

Template matching can perform semantic parsing process for C-KBQA. A set of pre-defined answer templates are defined and slots to be filled (e.g., subject, predicate, and constraints) are created. These answer templates are related to the KB and have different formats to deal with multi-hops and constraints. The slots are filled with features of the complex question to answer the question. The combination of Semantic Parsing and Neural Networks came as the next step to solve the C-KBQA problem (LUO et al., 2018; DING et al., 2019). A Neural Network can classify the templates given the feature extracted from a complex question, to perform the template matching. This approach consists of training a neural network to match a set of semantic parsing rules, in this paper, a set of answer templates. This can reduce the computationally cost of using Semantic parsing alone.

This work addresses the problem of answering complex questions with a semantic template matching approach for C-KBQA systems. The C-KBQA approach uses the combination of Semantic Parsing and Neural Networks techniques to determine the answer templates that a complex question fits. Moreover, an attention mechanism was created to assist the neural network in selecting the most important information. In the so-called Hereditary Attention, each neural network cell inherits the attention from another neural network cell, in a bottom-up way. However, good datasets for C-KBQA are an open challenge and it is a limitation to perform and evaluate a template matching approach for C-KBQA. To mitigate this problem, we released a new version of a C-KBQA dataset containing answer templates.

The main contributions of this paper are threefold: (i) a new C-KBQA approach for template matching; (ii) a hereditary attention mechanism to assist in question semantic extraction that achieved promising results and better accuracy than related work; (iii) a new preprocessed version of a dataset for complex question answering that other researchers can use.

The remainder of the paper is structured as follows: in Section 3.2 the background and related work are presented. In Section 3.3 the details of each step our C-KBQA approach are presented. In Section 3.4 the evaluation setup and dataset pre-processing methodology are described. Section 3.5 our results and challenges are presented. Finally, in Section 3.6 presents our conclusions remarks and future directions are presented.

3.2 RELATED WORK

Some works started to address the C-KBQA problem in different ways. For C-KBQA systems, the complex questions can be divide into two subgroups: multi-hop questions and constraint questions. Multi-hop questions, a C-KBQA system has to handle several subjects and predicates that can be found in the question (LI; HU; ZOU, 2020). The entities detected in those questions need to be linked, and it is necessary to deal with indirect relations, unlike simple questions that can be answered directly. The triple connections (subject $\xrightarrow{predicate}$ object) inside a KB are explored, and the systems make hops between the objects detected in the NLQ and the KB relations to get the target entity. In constraint questions, the NLQ often includes some restrictions that limit the answering options for a given question (SHIN; LEE, 2020). Those restrictions can be of several types, for example, temporal, ordinal, quantitative, and others. These constraints can modify the main subjects of an NLQ and consequently change the answer.

In Yin, Ge e Wang (2014) the authors dealt with the multi-hops questions with a Semantic Parsing approach. The author tries to directly map a complex question, creating a set of rules to define the type of complex question and match them into a logical format. Jia et al. (2018) present advances to solve constraint questions with a module capable to deal with constraint questions. This work creates a new module, called TEQUILA, that addresses part of the problem related to complex temporal questions, re-using and improving other KBQA systems like AQQU (BAST; HAUSSMANN, 2015), and QUINT (ABUJABAL et al., 2017). Both AQQU and QUINT creates template question to answer simple questions, however, the systems have limited coverage for complex questions. TEQUILA came as a side module to improve the answer to temporal questions using decomposition and re-composition rules. We address the problem of answering complex questions with a template matching approach as AQQU and QUINT, however, our approach is capable to deal with a complex question based on the semantic structure of an NLQ.

Talmor e Berant (2018) propose a system that decomposes complex questions into a sequence of simple questions. The final answer is computed from the sequence of simple question answers, using a set o semantic parsing rules. The authors also present that datasets for C-KBQA are a limitation and released a novel dataset for complex questions. Bhutani, Zheng e Jagadish (2019) also addresses the complex question with decomposition technique in a system called TextRay, that searches for sub-graphs through a combination of Semantic Parsing and Neural Networks. Two LSTM are used in the semantic matching model to extract the semantic relation between the NLQ and the KB entities and relations. A new version of TextRay is present in Bhutani et al. (2020) where the system handles the search of answers in multiple KB with an additional module that computes the similarity among multiples KB. The combination of Semantic parsing and Neural network is also used in this paper. We created a Hereditary Tree-LSTM that is capable to classify the template that an NLQ matches given the semantic feature extracted from a complex question. Also, we present the problems with C-KBQA datasets and released a new cleaned dataset to help in the C-KBQA dataset limitation.

Zafar, Napolitano e Lehmann (2018) present a semantic parsing module for query generator. The so-called SQG detects subgraphs paths in KB using Named Entity Disambiguation and Relation Extraction tasks. However, the SQG has a limited cover of complex question types. In Abdelkawi et al. (2019) the authors reused the SQG and created two extra modules to better handle ordinal questions (constraint questions). Dileep et al. (2021) presents a C-KBQA system over LC-QuAD 2.0. The authors presented that the XGBoost achieve good results in template matching. Athreya et al. (2021) also uses the Tree-LSTM for question answering on LC-QuAD 1.0 and shows that Tree-LSTM can have good performance in questing answering. Finally, Diomedi e Hogan (2021) adopt a neural machine translation approach to translate an NLQ into a structured query language (templates) and also present some limitations of C-KBQA datasets. Our so-called Hereditary Attention is used to assist our Tree-LSTM architecture in deal with complex questions. The attention mechanism is used to emphasize the relevant parts of the question and so detect the target template. So, it is possible to relate the constraint in the NLQ with the respective templates instead of subgraphs paths.

Given this overview of works in C-KBQA, we highlight the following points, which make our proposed approach distinct from the previous ones:

1. A novel architecture for question answering using template matching. It is the so-called Hereditary Tree-LSTM. The architecture uses a Hereditary Attention, where each neural network cell inherits the semantic attention from another neural network cell, in a bottom-up way.

2. Datasets for C-KBQA are an open challenge and it is a limitation to perform and evaluate approaches for C-KBQA. Here, we present some limitations of one of the largest datasets for C-KBQA, the LC-QuAD 2.0 (DUBEY et al., 2019). These issues can cause systems to answer questions wrongly or mask errors when evaluating question answering systems. To mitigate this problem, a new version called LC-QuAD 2.1 containing answer templates was released. LC-QuAD 2.1 is a cleaned version of the original dataset, whiteout duplicated questions, malformed questions, and other problems present in the Section 3.4 Also, the training, development, and

testing sets used in this paper were released so future work in C-KBQA can compare their approach with the approach presented here.

3.3 C-KBQA APPROACH

A C-KBQA system can be divided into three steps: question parsing, question representation, and candidate ranking. In a nutshell, in the question representation step, semantic mapping is performed. The question representation step structure the semantic mapping in a KB intermediate semantic format representation. Finally, after the semantic representation, the candidate ranking step performs the removal of incorrect answers based on the type, entities, predicates, and semantics detected on the original question. In this paper, our goal is to present a different solution for the question representation step.

We create a Neural Networks-based Semantic Parsing system to deal with complex questions by using a combination of Semantic Parsing and Neural Network techniques. Semantic parsing is the mapping of NLQ to a meaning representation that can be further represented as a logic form (TONG; ZHANG; YAO, 2019). In this process, the NLQ is transformed into an intermediate representation (TRIVEDI et al., 2017). We are using a template matching approach where a question can be mapped into an intermediate representation (our template) of a KB.

A Recurrent Neural Network (RNN) is used to select the best-appropriated template based on the semantic of the question. Our template-based C-KBQA was created using a Tree-LSTM architecture. Usually, the Tree-LSTM is used for sentiment classification and semantic relatedness of sentence (TAI; SOCHER; MANNING, 2015). However, in this paper, we implemented a Tree-LSTM to create a C-KBQA system that will be able to extract the semantic of a question and decide the answer template that a question belongs to.

An attention mechanism was implemented to assist in selecting the most important information. Attention is used to emphasize the more relevant parts of a question and to preserve the context of the sentences (VASWANI et al., 2017; BHUTANI; ZHENG; JAGADISH, 2019; BHUTANI et al., 2020). In the so-called Hereditary Attention mechanism, the attention layer inherits the attention of the children of each sub-tree of the Tree-LSTM, passing this information on a bottom-up way. The math behind the Tree-LSTM and the Tree-LSTM with hereditary attention architectures is detailed in the next subsections. The combination of Tree-LSTM and hereditary attention is called Hereditary Tree-LSTM.

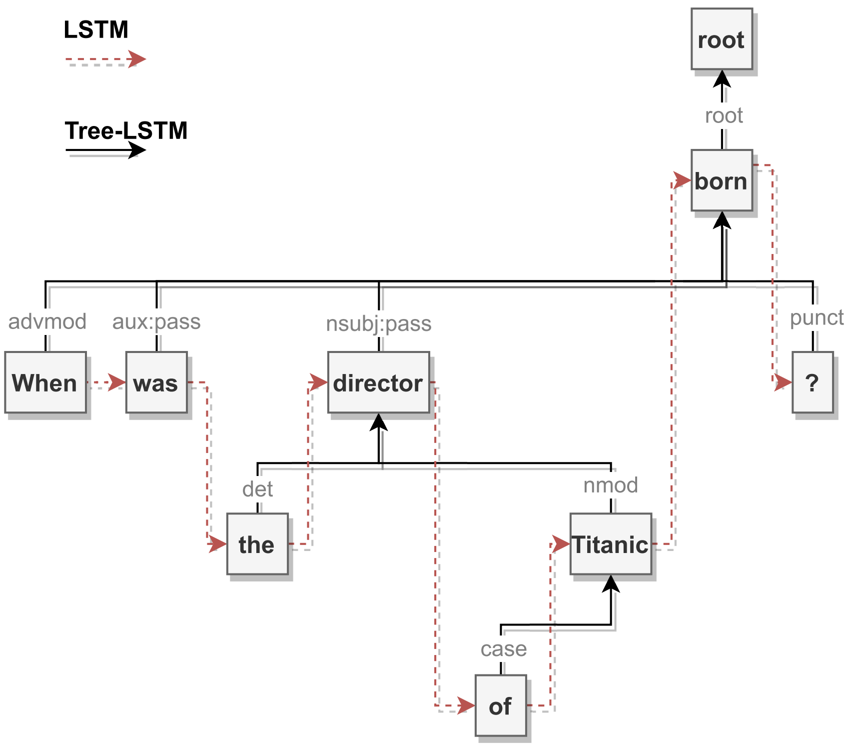
3.3.1 TREE-LSTM

Tree-LSTM is a variation of the standard Long Short Term Memory (LSTM) where the LSTM unit cells are structured in a tree format, passing information from the leaves nodes to the root node. The Tree-LSTM unit hierarchically incorporates information from each child node, while the standard LSTM disseminates historical information through a sequence of neural network units.

The Tree-LSTM can correctly analyze the structured semantic information of a sentence in contrast to the standard linear chain of the LSTM and the bidirectional LSTM (TAI; SOCHER; MANNING, 2015). For this reason, a Tree-LSTM model can learn to emphasize semantic heads in a semantic relatedness task, or it can learn to preserve the representation of sentiment-rich children for sentiment classification (TAI; SOCHER; MANNING, 2015; MIYAZAKI; KOMACHI, 2018; AHMED; SAMEE; MERCER, 2019). In the Complex Question Answering task, where all the information of the sentence has to be understood, a Tree-LSTM can be valuable architecture to solve this problem.

Figure 13 presents the difference of time execution between the Tree-LSTM and the standard LSTM. The tree structure was created using the semantic dependency tree structure of the question "When was the director of Titanic born?". Each rectangle represents an LSTM or a Tree-LSTM cell. The red arrow presents the linear sequential chain of the LSTM, from the word "When" until the character "?". The black arrow presents the hierarchical execution of the Tree-LSTM from the word "of" until the root node, following the semantic dependecy structure of the question, in a bottom-up way fashion.

Figure 13 – Difference of time execution between the Tree-LSTM and the standard LSTM architectures.



Source: Created by the author (2021)

There are two types of Tree-LSTM: Child-Sum and N-Array. In the Child-Sum, the Tree-LSTM can have many children as the tree structure that was selected to be used, and in the N-Array the Tree-LSTM can have only N children per level (e.g., a binary tree). We used the Child-Sum Tree-LSTM version since we want to capture all the semantics behind a complex question composition.

Similar to the standard LSTM, the Tree-LSTM uses input gate, forget gate, output gate, memory cell, and hidden state. As present in Tai, Socher e Manning (2015), given a tree, let C(j) denote the set of children of node j, x_j denotes the input, i_j denotes the input gate, f_j denotes the forget gate, o_j denotes the output gate, c_j denotes the memory cell, and h_j denotes the hidden state. The Child-Sum Tree-LSTM transition equations are the following:

$$\tilde{h}_j = \sum_{n \epsilon C(j)} h_n \tag{3.1}$$

$$i_j = \sigma(\mathrm{W}^{(i)} x_j + \mathrm{U}^{(i)} \tilde{h}_j + b^{(i)}) \tag{3.2}$$

$$f_{jk} = \sigma(\mathrm{W}^{(f)} x_j + \mathrm{U}^{(f)} \tilde{h}_k + b^{(f)}) \tag{3.3}$$

$$o_j = \sigma(\mathrm{W}^{(o)} x_j + \mathrm{U}^{(o)} \tilde{h}_j + b^{(o)}) \tag{3.4}$$

$$u_j = \tanh(\mathrm{W}^{(u)} x_j + \mathrm{U}^{(u)} \tilde{h}_j + b^{(u)}) \tag{3.5}$$

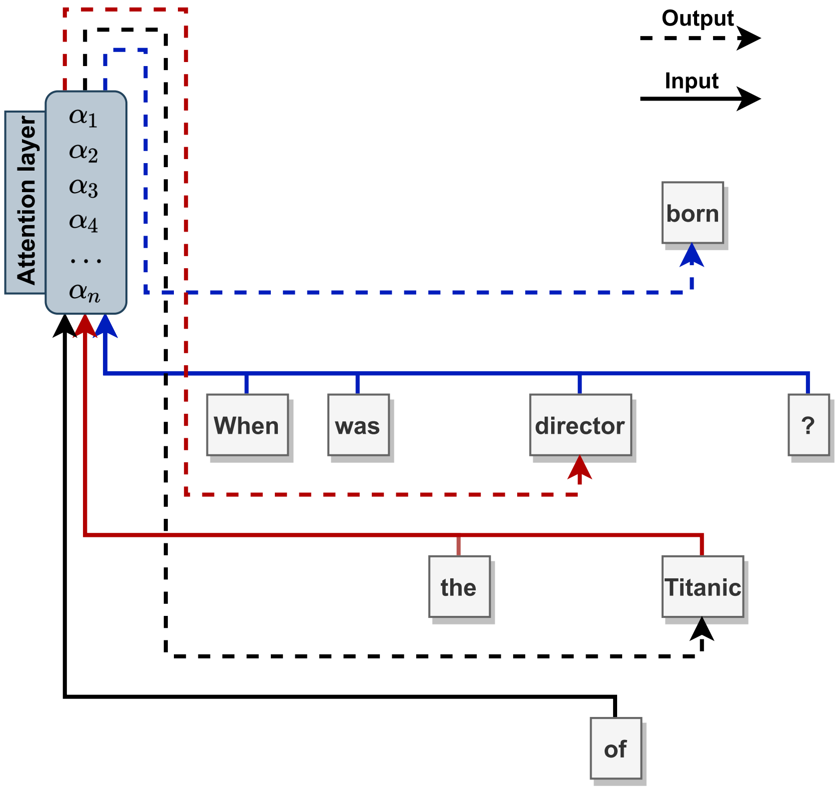
$$c_j = i_j \odot u_j + \sum_{k \epsilon C(j)} f_{jk} \odot c_k \tag{3.6}$$

$$h_j = o_j \odot \tanh(c_j) \tag{3.7}$$

3.3.1.1 TREE-LSTM WITH HEREDITARY ATTENTION

Miyazaki e Komachi (2018), Ahmed, Samee e Mercer (2019) presented that the attention mechanism for Tree-LSTM can achieve good results for the semantic relatedness and sentiment classification of texts tasks. In this paper, a hereditary attention mechanism was created to assist the Tree-LSTM in focus only on the relevant information of a complex natural language question. The attention is applied successively in the set of children of each sub-tree and decides the features more relevant that have to be emphasized to build the new hidden state of this sub-tree. In this way, the children's hidden states of each sub-tree are weighted with a "factor of importance", until the root node be achieved (bottom-up). Figure 14 presents an example of the hierarchical attention version of the Tree-LSTM. The solid arrow presents the input of the attention layer and the dotted arrow presents the output after the weighted process. Each arrow color represents the flow of a level (children set of each sub-tree) in the semantic structure of the question presented in Figure 13.

Figure 14 – Hereditary Tree-LSTM.



Source: Created by the author (2021)

The hereditary attention is based on the self dot product attention (VASWANI et al., 2017) with a scale factor. Three components are used when creating attention mechanisms: query, key, and value. The query is the information that you are looking for. The key is the set of assets that can be used when making a query. The value is the matched result for your query given the set of keys used. These three components are used here as weight matrices, that will be learned during the training step. Each matrix is constructed using a linear layer. The attention created in this paper is similar to the presented in Zhang et al. (2019), where the authors used the self dot attention for image generation tasks. However, the authors used convolution layers instead of the linear layers and do not have a normalization factor to weigh the information that will be preserved of each set o children.

Let H denote all children's hidden states of a given sub-tree concatenated ($H =$

$[h_1; h_1; h_1...h_n]$), d denote memory dimension of the Tree-LSTM. $W^{(q)}, W^{(k)}, W^{(v)}$ denote, respectively, the query, key, and value weight matrices, and $b^{(q)}, b^{(k)}, b^{(v)}$ denote, respectively, the query, key, and value bias. The query, key, and value matrix are constructed using the following equations:

$$query = \mathrm{W}^{(q)}H + b^{(q)} \tag{3.8}$$

$$key = \mathrm{W}^{(k)}H + b^{(k)} \tag{3.9}$$

$$value = \mathrm{W}^{(v)}H + b^{(v)} \tag{3.10}$$

The query, key, and value matrices are of dimension n x d, where n represents the number of children concatenated in H (Equations 3.8, 3.9, and 3.10). The relation between the information that we are searching for (query matrix) and the set of components that can be used when searching (key matrix) is now calculated. This relation is called energy matrix and it is calculated performing a matrix multiplication between the query and key matrix.

$$scale = \frac{1}{\sqrt{d}} \tag{3.11}$$

$$energy = mm(query^T, key) * scale \tag{3.12}$$

The new energy matrix (Equation 3.12) is a matrix d x d that represents the energy that the hidden states have to this set of children (H). Also, a re-scale factor is used as a normalization factor (Equation 3.11). In this step, our approach defers from Ahmed, Samee e Mercer (2019) as we do not want to diminish the information of one child concerning the other child, but extract the most important information from both children. So, the energy matrix is of size d x d instead of n x n, as suggest for Ahmed, Samee e Mercer (2019).

To calculate the attention matrix, a softmax function is applied (Equation 3.13). The energy matrix is re-scale into the [0,1] scale. This matrix represents how much the information has to be re-weighted, highlighting the important information on this set of children. The attentive hidden states (h_{att}) are calculated using a matrix multiplication between the value matrix and the attention weights (Equation 3.14).

$$attention = softmax(energy) \tag{3.13}$$

$$\tilde{h}_{att} = mm(value, attention) \tag{3.14}$$

The h_{att} is of size n x d and represents the pieces of information more important of the hidden states of those children set. Therefore, as represented in the equation 3.1, the new $\tilde{h}$ will be the sum of all children with attention (Equation 3.15).

$$\tilde{h}_j = \sum_{n \epsilon C(j)} \tilde{h}_{att_n} \tag{3.15}$$

The remaining equations are the same presented for the Tree-LSTM (Equations 3.2, 3.3, 3.4, 3.5, 3.6, and 3.7), however, now all the operations are taking into account the highlighted information of each sub-tree. The higher nodes on the Tree-LSTM are inheriting the highlighted information from their children.

3.4 TRAINING AND EVALUATION METHODOLOGY

This section presents the training setup (e.g. dataset, data preprocessing, target, hyperparameter tuning, etc) and evaluation methodology.

3.4.1 DATASETS AND PRE-PROCESSING

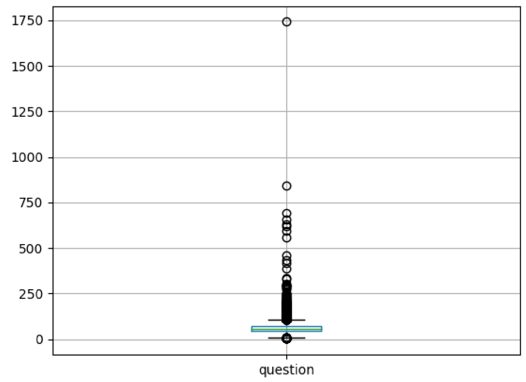
The “Large Dataset for Complex Question Answering over Wikidata and DBpedia” version 2.0 (LC-QuAD-2) was used as the main experimental dataset. The LC-QuAD-2 (DUBEY et al., 2019) is one of the most recent datasets for C-KBQA. LC-QuAD-2 included more types of complex questions, e.g., up to 6 hops questions and more constraint questions than the previous version. The dataset contains over 30,000 questions, composed of 21,258 entities and 1,310 predicates, created using Amazon Mechanical Turk[2], a crowdsourcing platform. The dataset presents the logical form of each question in SPARQL format. Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014) and DBpedia 2018 (LEHMANN et al., 2015) were used as Knowledge Base. Also, the dataset contains a paraphrase question for almost all original questions on the dataset.

Inconsistencies can be inputted on the data (e.g. text duplication, missing information, etc) when performing a crowdsourcing approach. The dataset was pre-processed to deal with those types of problems. Other works, such as Diomedi e Hogan (2021), also present problems with inconsistencies in the dataset. To standardize the dataset, first, all the instances with empty or duplicated question fields were excluded to avoid any bias in the data during the separation of training, development, and testing sets. The instances with two or more questions in the same id, e.g “What was the name of Kartikeya child? Shiva?”, were also removed as we could not ensure to which question the answer was related. In this example, the answer can be a KB entity or a boolean value. Finally, we notice that the original training and testing sets, provided by the authors of the dataset, had some questions that appear in both sets and they were removed.

[2] https://www.mturk.com/

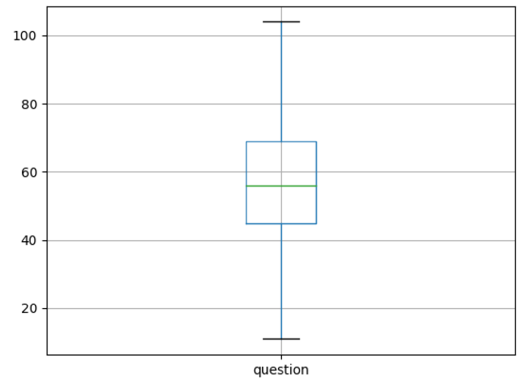
The above process was performed to create a more consistent dataset. The inconsistencies can make a QA system learn to answer questions that are not made by real users (e.g. questions so big or so small) or be overfitted on questions of the same type due to question duplication. It was verified that some instances have a size incompatible with the rest of the dataset. On the boxplot (Figure 15) it is presented the length of all questions in the dataset before the normalization. Some questions have more than 500 characters and less than 5 characters. We removed all the questions that had more them 120 characters and less than 15 characters to make the data homogeneous. Figure 16 presents the results after the outliers removal (considering the question length).

Figure 15 – Question length before sentence length normalization.



Source: Created by the author (2021)

Figure 16 – Question length after sentence length normalization.

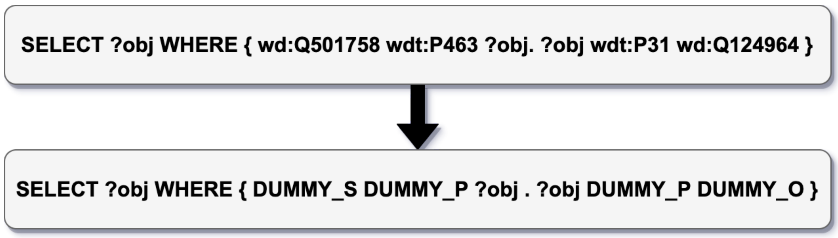


Source: Created by the author (2021)

The dataset contains a SPARQL-based logical form for each question. The original SPARQL was transformed into templates. These templates are used as answer templates that are matched in the question representation step. The subject, predicates, objects, and filters found in the original SPARQL were masked into dummy tokens (DUMMY_S, DUMMY_P, DUMMY_O, DUMMY_F). The templates are so-called Dummy SPARQL templates and in Figure 17 it is presented the creation process of these templates. After this process, 29 unique Dummy SPARQL templates were created for the Wikidata KB, and 25 unique Dummy SPARQL templates were created for the DBpedia KB.

Several Dummy SPARQL templates have a similar semantic and only minor difference between the answer templates. The difference between two Dummy SPARQL templates can be only a projection on the subject or the object, or it can be a restriction less than or greater than a value (filter), for example. We performed a semantic analysis on the dummy SPARQL templates and grouped them by proximity between the Dummy SPARQL templates. To create each SPARQL group, seven items were analyzed: the number of projections, hops, and filters and if the template contains a limit, order, count,

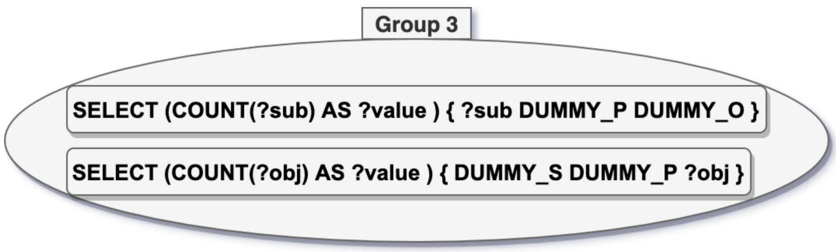
Figure 17 – Creation of Dummy SPARQL templates.



Source: Created by the author (2021)

or distinct logical operators. We order these items by the number of occurrences of each one on the dummy SPARQL templates and 12 unique dummy SPARQL groups for Wikidata and 10 unique dummy SPARQL groups for DBpedia were created. In Figure 18 it is presented the creation process of a dummy SPARQL group where the first template has a subject projection and the second has an object projection.

Figure 18 – Creation of group Dummy SPARQL templates by SPARQL aproximation.



Source: Created by the author (2021)

In the Table 12 and Table 13 it is presented the distribution of question for each Dummy SPARQL template ID and group ID for the Wikidata KB. Read GID as group ID, TID as template ID, #Q as the number of questions in the template ID, and #TQ as the number of questions in the group when reading the columns names. The Tables show that there is a high number of imbalanced data on the dataset, with different types of answer templates. The answer templates are of several types of complex questions, e.g., less/greater than or year restrictions, and multi-hops. Additional information for DBpedia 2018 can be found in the Appendix.

This cleaned version of the dataset is so-called LC-QuAD 2.1. All the preprocessing made in this paper is helpful for other C-KBQA approaches. Besides templating matching, LC-QuAD 2.1 can be used for semantic parsing, such as subgraph path detection or the creation of logical grammar rules. The LC-QuAD 2.1 is available for further research (GOMES JR. et al., 2021).

Table 12 – Questions distribution by dummy template and dummy group ids for the Wikidata KB part 1.

GID	TID	DUMMY SPARQL	#Q	#TQ
0	0.1	ASK WHERE { DUMMY_S DUMMY_P DUMMY_O }	421	1912
	0.2	ASK WHERE { DUMMY_S DUMMY_P ?obj filter(?obj = DUMMY_F) }	1027	
	0.3	ASK WHERE { DUMMY_S DUMMY_P ?obj filter(?obj > DUMMY_F) }	225	
	0.4	ASK WHERE { DUMMY_S DUMMY_P ?obj filter(?obj < DUMMY_F) }	239	
1	1.0	ASK WHERE { DUMMY_S DUMMY_P DUMMY_O . DUMMY_S DUMMY_P DUMMY_O }	363	363
2	2.1	SELECT DISTINCT ?answer WHERE { ?answer DUMMY_P DUMMY_O}	657	3580
	2.2	SELECT DISTINCT ?answer WHERE { DUMMY_S DUMMY_P ?answer}	2923	
3	3.1	SELECT (COUNT(?sub) AS ?value) { ?sub DUMMY_P DUMMY_O }	547	960
	3.2	SELECT (COUNT(?obj) AS ?value) { DUMMY_S DUMMY_P ?obj }	413	
4	4.1	SELECT ?answer WHERE { DUMMY_S DUMMY_P ?X . ?X DUMMY_P ?answer}	2911	8239
	4.2	SELECT ?answer WHERE { DUMMY_S DUMMY_P ?answer . ?answer DUMMY_P DUMMY_O}	1747	
	4.3	SELECT DISTINCT ?sbj WHERE {?sbj DUMMY_P DUMMY_O . ?sbj DUMMY_P DUMMY_O}	1694	
	4.4	SELECT DISTINCT ?obj WHERE { DUMMY_S DUMMY_P ?obj . ?obj DUMMY_P DUMMY_O}	1546	
	4.5	SELECT ?ent WHERE {?ent DUMMY_P DUMMY_O ?ent DUMMY_P ?obj}ORDER BY DESC(?obj)LIMIT DUMMY_F	341	
5	5.1	SELECT ?obj WHERE {DUMMY_S DUMMY_P ?s . ?s DUMMY_P ?obj . ?s DUMMY_P DUMMY_O}	911	2913
	5.2	SELECT ?value WHERE {DUMMY_S DUMMY_P ?s . ?s DUMMY_P DUMMY_O . ?s DUMMY_P ?value}	2002	
6	6.1	SELETC ?ent WHERE {?ent DUMMY_P DUMMY_O . ?ent DUMMY_P ?obj . ?ent DUMMY_P DUMMY_O} ORDER BY ASC(?obj)LIMIT DUMMY_F	285	569
	6.2	SELECT ?ent WHERE {?ent DUMMY_P DUMMY_O ?ent DUMMY_P ?obj ?ent DUMMY_P DUMMY_O} ORDER BY DESC(?obj)LIMIT DUMMY_F	284	

Source: Created by the author (2021)

Table 13 – Questions distribution by dummy template and dummy group ids for the Wikidata KB part 2.

GID	TID	DUMMY SPARQL	#Q	#TQ
7	7.1	SELECT ?value WHERE {DUMMY_S DUMMY_P ?s . ?s DUMMY_P ?x filter(contains(YEAR(?x)'DUMMY_F')) . ?s DUMMY_P ?value }	43	2069
	7.2	SELECT ?value WHERE {DUMMY_S DUMMY_P ?s . ?s DUMMY_P ?x filter(contains(?x,DUMMY_F)) . ?s DUMMY_P ?value}	397	
	7.3	SELECT ?value WHERE {DUMMY_S DUMMY_P ?s . ?s DUMMY_P ?x filter(contains(?x,DUMMY_F)) . ?s DUMMY_P ?value}	1419	
	7.4	SELECT ?obj WHERE {DUMMY_S DUMMY_P ?s . ?s DUMMY_P ?obj . ?s DUMMY_P ?x filter(contains(?x,'DUMMY_F'))}	210	
8	8	SELECT ?ans ?ans WHERE {DUMMY_S DUMMY_P ?ans DUMMY_S DUMMY_P ?ans}	490	490
9	9.1	SELECT DISTINCT ?sbj ?sbj_label WHERE{ ?sbj DUMMY_P DUMMY_O ?sbj DUMMY_P ?sbj_label FILTER(CONTAINS(lcase(?sbj_label), DUMMY_F)) FILTER (lang(?sbj_label) = DUMMY_F) }LIMIT DUMMY_F	802	1614
	9.2	SELECT DISTINCT ?sbj ?sbj_label WHERE{ ?sbj DUMMY_P DUMMY_O ?sbj DUMMY_P ?sbj_label FILTER(STRSTARTS(lcase(?sbj_label), DUMMY_F)) FILTER (lang(?sbj_label) = DUMMY_F) }LIMIT DUMMY_F	812	
10	10	SELECT ?value1 ?obj WHERE{ DUMMY_S DUMMY_P ?s . ?s DUMMY_P ?obj . ?s DUMMY_P ?value1 }	493	493
11	11.1	SELECT DISTINCT ?sbj ?sbj_label WHERE{ ?sbj DUMMY_P DUMMY_O ?sbj DUMMY_P DUMMY_O ?sbj DUMMY_P ?sbj_label FILTER(CONTAINS(lcase(?sbj_label), DUMMY_F)) FILTER (lang(?sbj_label) = DUMMY_F }LIMIT DUMMY_F	198	484
	11.2	SELECT DISTINCT ?sbj ?sbj_label WHERE{ ?sbj DUMMY_P DUMMY_O ?sbj DUMMY_P DUMMY_O ?sbj DUMMY_P ?sbj_label FILTER(STRSTARTS(lcase(?sbj_label), DUMMY_F)) FILTER (lang(?sbj_label) = DUMMY_F) }LIMIT DUMMY_F	286	
12	12	SELECT ?value1 ?value2 WHERE{ DUMMY_S DUMMY_P ?s . ?s DUMMY_P DUMMY_O . ?s DUMMY_P ?value1 . ?s DUMMY_P ?value2 }	448	448

Source: Created by the author (2021)

3.4.2 TRAINING AND EVALUATION SETUP

To evaluate the C-KBQA approach, the dummy templates created for Wikidata were used as the target for template matching. The POS-tagger and Semantic Dependency Tree (DT) from the Stanford CoreNLP toolkit (MANNING et al., 2014) were used to extract the semantic of a question and to create the tree structure used for the Tree-LSTM.

The inputs for the Tree-LSTM were created using the FastText pre-trained word embedding (BOJANOWSKI et al., 2017). Each word in the question was transformed into a word vector. Word embeddings reduce the computational complexity since the matrix operations through these word vectors are fast to compute. Also, the DT relations and POS-Tagger tokens were transformed into one-hot-encoding vectors. The word vector, the DT one-hot-encoding vector, and the POS-Tagger one-hot-encoding vector were concatenated to create the input vector. To decide which template a question matches, a softmax classifier is applied at the root node of the Tree-LSTM, and the cost function used is the negative log-likelihood.

The LC-QuAD 2.1 dataset contains 26,376 unique questions and 25,395 paraphrase questions. We created two training setups: one using only the original questions to train and another using paraphrase questions. The first setup was divided into 20,177 questions on the training set, 2,242 questions on the development set, and 3,957 questions on the testing set. In the second setup, the training set was composed of 45,572 questions (original questions + paraphrased questions), and the development and testing sets are the same. The paraphrase questions are used to evaluate the brittleness of the C-KBQA system when a question only with a few modifications is inserted and only added to the training set. Both setups were performed in a stratified fashion, based on the dummy templates, due to the imbalanced scenario.

Finally, a hyperparameter tuning step was performed to select the best model. Table 14 presents the values of the best model selected in the development set.

Table 14 – Model setup.

Hyperparameter	Value
Input dimensions	387
Word embedding dimensions	300
LSTM memory dimensions	150
Attention dimensions	150
Learning rate	0.01
Batch size	25
Dropout	0.2
Weight decay	1e-6
Optimizer	SGD

Source: Created by the author (2021)

3.5 RESULTS

Four works were select as baselines to compare against our C-KBQA approach. These works were select for two reasons: (i) the authors created a template matching approach/module or have it as part of a full C-KBQA system; (ii) it is possible to retrain their approach with another dataset (open source code) or the authors also evaluated the results on LC-QuAD 2.0, even without using a preprocessed version of the dataset. The baselines are listed as follow:

- (DILEEP et al., 2021): In this paper, two machine learning models and three different preprocessing techniques were used to generate results and identify the best model for template matching on Lc-QUAD-2. The authors presented that the XGBoost achieve good results in template matching.
- (ATHREYA et al., 2021): This work also uses a Tree-LSTM, but on LC-QuAD 1.0. We retrained our model using the LC-QuAD 1.0 to compare our system against theirs. LC-QuAD 1.0 (TRIVEDI et al., 2017) contains 5000 questions composed of 5042 entities and 615 predicates. The questions were created from a list of template questions that the work of Athreya et al. (2021) tries to predict what type of template a natural language question fits.
- (DIOMEDI; HOGAN, 2021): This work adopts a neural machine translation (NMT) approach to translate a natural language into a structured query language (templates). NMT is then used to create a query template with entities placeholders, similar to our dummy answer template but it was only performed for entities (subjects and objects).
- (EVSEEV; ARKHIPOV, 2020): This work performed the template matching step is using a BERT classification and Support Vector Machine to determine the SPARQL template type. The authors presented that the BERT classification achieve the best results in template matching.

It was carried out three experiments and we use HTL to refer to our Hereditary Tree-LSTM in these experiments. First, we compare HTL to Dileep et al. (2021) and Diomedi e Hogan (2021) using LC-QuAD 2.1 and 29 target classes (templates). Next, we compare HTL to Evseev e Arkhipov (2020) using LC-QuAD 2.1 and 13 grouped target classes. The classes were grouped according to protocol presented in Section 3.4 (see the Tables 12 and 13). Finally, we use LC-QuAD 1.0 to compare results HTL to Athreya et al. (2021) using the same preprocessing performed by the authors, resulting in 15 target classes. Table 15 presents the results. The experiments were evaluated using the accuracy, and macro and weighted metrics: precision, recall, and f1-score. Some metrics are not available because the original paper did not provide these values, or we could not reproduce their results.

Table 15 – Results of template matching against the baselines. Read "LC" as LC-QuAD, "Acc" as Accuracy, "MA" as Macro and "W" as Weighted.

Dataset	Approach	Acc	Precision		Recall		F1-Score	
			MA	W	MA	W	MA	W
LC 2.1	HTL	73.3	73.6	72.8	70.8	73.3	71.3	72.6
	HTL w/paraph	89.1	89.5	89.1	89.1	89.1	89.2	89.0
	Dileep et al. (2021)[a]	67.2	68.5	68.5	63.6	67.2	65.2	66.1
	Diomedi e Hogan (2021)[b]	34.3	-	-	-	-	-	-
LC 2.1 (groups)	HTL	85.5	84.9	85.5	84.8	85.5	84.8	85.5
	HTL w/paraph	92.4	91.3	92.4	92.8	92.4	92.0	92.4
	Evseev e Arkhipov (2020)[b]	90.8	-	-	-	-	-	-
LC 1.0	HTL	81.7	82.3	82.2	81.8	81.7	81.7	81.7
	Athreya et al. (2021)[b]	82.8	-	-	-	-	-	-

[a] An accuracy of 0.92 is presented on the paper. The authors used the "question_type" feature provided by the LC-QuAD 2.0 to train their approach. This feature is used to map the question type, e.g if is one intention or boolean question. HTL scored an accuracy of 0.98 with this feature. As it is not possible to reproduce this feature in a real application, we retrained their approach without this feature.

[b] Extracted from the original paper regarding the template matching/construction step.

Source: Created by the author (2021)

New approaches need to deal better with biases and do not create brittle and spurious systems. Some C-KBQA systems are brittle because they are not robust enough yet and can fail to answer a question when just a few parts of the question are a little modified, even if the main meaning is preserved (JIA; LIANG, 2017). It is necessary to handle carefully the dataset pre-processing step, separate it with balanced training classes, and also try to explore some question paraphrases to ensure that most types of questions are being explored. Our system was evaluated by adding the paraphrased questions in the training set to analyze the brittleness of the system. In Table 15, we used "HTL w/ paraph" to refer to HTL with paraphrase question.

None of the baselines used a development set during the training step and the hyperparameter tuning was performed on the testing set. Even so, the results in Table 15 show that our approach outperforms most of the baselines. HTL achieves good accuracy, precision, and recall. Regarding the results using the full template classification, HTL achieved the best result with 73.3% of accuracy and more than 70% for precision, recall, and f1-score. This shows that HTL achieves good performance even in this imbalanced scenario with multiple classes. When using the paraphrase questions, the results of HTL get even better, improving the metrics in more than 16 pp.

In Evseev e Arkhipov (2020), the authors grouped the template into smaller groups to evaluate their approach. HTL achieved 85.5% of accuracy during the group classification while the work of Evseev e Arkhipov (2020) achieved 90.8%. There are some points to observe when comparing our work with Evseev e Arkhipov (2020). Their template matching step is performed using a BERT, with is a fully connected attention mechanism. The training of these approaches can be computationally expensive and limit the amount of tuning done (LIU et al., 2019). Our approach is less expensive and can be easily specialized for other KB. Also, as the results were not publicly disclosed, we cannot assess the differences in the training/testing sets neither the performance in the imbalanced scenario (regarding the precision, recall, and f1-score metrics). Besides, when using the paraphrase questions, HTL overcomes the baseline work with 1.6 pp of accuracy.

Comparing to Athreya et al. (2021), HTL almost achieve the same results presented by the authors. The authors used one-hot encoding of characters as additional input that helps the network to achieve better results. However, this additional feature can easily overfit the training and do not generalize for characters that have not appeared before. In our HTL, the hereditary attention mechanism is used to assist the Tree-LSTM in detecting the most relevant information of a question. Only with HTL, we were capable to achieve accuracy with only 1.1 pp smaller.

As discussed in Section 3.2, the complex questions can be divide into two subgroups: multi-hop questions and constraint questions. In multi-hop questions, a C-KBQA system has to handle several subjects and predicates that can be found in the question (LI; HU; ZOU, 2020). In constraint questions, the NLQ often includes some restrictions that limit the answering options for a given question (SHIN; LEE, 2020). Those restrictions can be of several types, for example, temporal, ordinal, quantitative, and others that modify the main subjects of an NLQ and consequently change the answer. Regarding the complex question match, HTL also has a great performance. We divided the question in LC-QuAD 2.1 into different types of natural language questions to illustrate this performance. In Table 16 it is presented the results of the template matching divided into four question types: Simple question, Constraint question, Multi-hop question, and both Multi-hop and Constraint at same time. It is possible to see that HTL can achieve good result for all question types. HTL has a macro and weighted average for precision, recall, and F1-score greater than 80% for all the question types.

3.6 FINAL REMARKS

This work presented a new C-KBQA approach using a template matching approach. The C-KBQA approach combines Semantic Parsing and Neural Networks techniques to determine the answer templates that is a complex question fits. A Tree-LSTM is used to correctly analyze the structured semantic information of a question. An attention mechanism was created to assist the Tree-LSTM in selecting the most important information. In

Table 16 – Results of HTL w/paraph divided for each question type. Read TID as dummy template id (see the Tables 12 and 13 to see the SPARQL dummy template).

Question type	TID	Precision	Recall	F1	Support
Simple	0.1	91.7	95.7	93.6	69
	2.1	67.8	72.2	70.0	108
	2.2	90.7	89.8	90.2	479
	3.1	83.0	92.2	87.4	90
	3.2	83.1	79.4	87.4	90
	Macro	**83.3**	**85.9**	**84.5**	**814**
	Weighted	**86.3**	**87.3**	**86.8**	
Constraint	0.2	96.0	100.0	98.0	168
	0.3	100.0	100.0	100.0	37
	0.4	100.0	100.0	100.0	39
	Macro	**98.7**	**100.0**	**99.3**	**244**
	Weighted	**97.2**	**100.0**	**98.6**	
Multi-hop	1.0	90.9	100.0	95.2	60
	4.1	89.7	91.6	90.7	477
	4.2	93.9	86.7	90.2	286
	4.3	79.6	78.4	79.0	278
	4.4	74.9	69.3	72.0	254
	5.1	87.7	85.9	86.8	149
	5.2	93.2	95.7	94.4	328
	8.0	82.6	95.0	84.8	34
	10.0	91.4	91.4	91.4	81
	12.0	90.4	89.2	89.8	74
	Macro	**87.4**	**88.3**	**87.8**	**2067**
	Weighted	**87.4**	**86.9**	**87.1**	
Multi-hop and Constraint	4.5	75.0	85.7	80.0	56
	6.1	97.8	93.6	95.7	47
	6.2	86.1	67.4	75.6	46
	7.1	85.7	85.7	85.7	7
	7.2	95.1	89.2	92.1	65
	7.3	97.5	98.7	98.1	233
	7.4	87.5	82.4	84.8	34
	9.1	95.5	97.0	96.2	132
	9.2	95.0	99.2	97.1	133
	11.1	93.1	84.4	88.5	32
	11.2	100.0	89.4	94.4	47
	Macro	**91.7**	**88.4**	**89.8**	**832**
	Weighted	**93.9**	**93.0**	**93.3**	

Source: Created by the author (2021)

the so-called Hereditary Tree-LSTM (HTL), each neural network cell inherits the attention from another neural network cell in a bottom-up way.

We presented the inconsistent on the original dataset, e.g., question duplication, and all the steps to create the answer templates used in this work. In addition, a new cleaned version of LC-QuAD 2.0, the so-called LC-QuAD 2.1, was released. The LC-QuAD 2.1 is available for further research and was used to evaluate most of the baselines used to compare our C-KBQA approach. The results show that our HTL can overcome most of the baselines in the template matching step. Also, we presented that some C-KBQA systems have to explore some question paraphrases to ensure that most types of questions are being explored. These systems can fail to answer a question when just a few parts of the question are a little modified, even if the main meaning is preserved.

There are still some challenges and limitations. KBQA systems are KB-dependent and it is necessary to ensure that the KB structure is up-to-date with the templates used in the training. Since the evaluation dataset is created at a fixed KB version, the system learns to answer the question related to the training dataset. The answer templates were based on SPARQL 1.0, however, a new version of SPARQL 1.1 was already released. With the advances in the query language, the templates have to follow these updates and be constantly updated to answer real questions. Also, the biggest mistake of the HTL is to differentiate templates with 2 and 3 hops. These errors occur as the KB schema was not included as inputs during the Hereditary Tree-LSTM training.

The use of the HTL still needs to be further investigated and additional information may improve the accuracy of the approach. The inclusion of the KB schema information on training can create more accurate results and need to be evaluated in future work. Also, the use of Named Entity Recognition and Disambiguation and relation recognition methods to detect KB entities/relations and measure the distance between them can improve the results of our template matching. Furthermore, explore HTL for other semantic tasks such as sentiment classification and semantic relatedness may also archive good results. Finally, evaluate HTL as part of a complete C-KBQA system still have to be evaluated.

4 CONCLUSIONS

This master thesis presented a study on Complex Knowledge Base Question Answering (C-KBQA) divide into two manuscripts (Chapters 2 and 3).

In the first manuscript (Chapter 2), a systematic mapping of the literature was performed to identify how the research has been addressing this problem. A protocol was adopted for the execution of systematic mapping to reduce the bias and make the study reproducible for other researchers. We showed that C-KBQA systems try to handle two types of complex questions: Multi-hop and Constraint questions. Also, we present an overview of the process to construct C-KBQA systems and how the main approaches are performed. We notice that the papers try to use two main approaches: Semantic Parsing and Neural Networks-based and their combination (called Neural Network-based Semantic Parsing). At last, we presented that good datasets for C-KBQA are still an open challenge and that the evaluation metrics for KBQA are still mostly information retrieval metrics, like F1-score and Accuracy.

As a new solution for the C-KBQA, the second manuscript (Chapter 3) presented a new C-KBQA approach using a template matching approach. The C-KBQA approach combines Semantic Parsing and Neural Networks techniques to determine the answer templates that a complex question fits. A Tree-LSTM was used to correctly analyze the structured semantic information of a question. An attention mechanism was created to assist the Tree-LSTM in selecting the most important information. In the so-called Hereditary Tree-LSTM (HTL), each neural network cell inherits the attention from another neural network cell in a bottom-up way.

We also argue in Chapter 3 about inconsistencies on the original LC-QuAD 2.0 dataset, e.g., question duplication, and all the steps to create the answer templates used in this paper. A new cleaned version of LC-QuAD 2.0, the so-called LC-QuAD 2.1, was released. The LC-QuAD 2.1 is available for further research and was used to evaluate most of the baselines used to compare our C-KBQA approach. The results on LC-QuAD 2.1 show that our HTL can overcome most of the baselines in the template matching step and so HTL can be a good option to be incorporated into KBQA systems. HTL can help these systems to be able to answer complex questions.

After all this work, it was possible to conclude that the C-KBQA area is still rising and it is expected to see a new C-KBQA system or new modules trying to improve the current C-KBQA system over the next years. Even the use of evaluation metrics for KBQA are still using information retrieval metrics, it is expected that the authors change their evaluation approaches with the advancement of research in C-KBQA systems and Natural Answer Generation to metrics like BLEU and ROUGE. Natural Answer Generation will make the C-KBQA system not only answer a question but also generates natural answer sentences for a given question.

Furthermore, C-KBQA systems have to explore some question paraphrases to ensure that most types of questions are being explored. These small modifications can change the entire sentence structure, however, have the same answer. C-KBQA systems need to ensure that they can not fail to answer a question when just a few parts of the question are a little modified, even if the main meaning is preserved. We presented that our HTL achieved better results when incorporated paraphrased questions.

4.1 LIMITATIONS

The discussion on the limitations is be divided into two parts: (i) mapping conduction and (ii) proposed approach.

- **Mapping conduction:** The systematic literature mapping aimed to present an overview of Complex Knowledge Base Question Answering. However, there are threats to its validity and limitations, like any research method. Removing articles not written in English and those in gray literature, for example, can diminish the accuracy of the conclusions, even though the mapping covered 54 articles. Also, some exclusion criteria could be more flexible. Furthermore, we did not consider all the relevant electronic databases, e.g., ACM Digital Library. So, it is possible that relevant studies were not indexed in the selection of this mapping. However, this research relies on the representativeness of the repositories selected to answer the research questions. Besides the relevant electronic databases as SCOPUS e IEEE, we also considered Google Scholar to mitigate relevant studies that were not indexed in our selection.

- **Proposed approach:** C-KBQA systems are KB-dependent and it is necessary to ensure that the KB structure is up-to-date with the templates used in the training. Since the evaluation dataset is created at a fixed KB version, the system learns to answer the question related to the training dataset. Furthermore, KB query languages are always evolving. As the answer templates were trained based on SPARQL, it is necessary to update the training if a previous version becomes obsolete. For example, the answer templates were based on SPARQL 1.0, however, a new version of SPARQL 1.1 was already released (without major changes). With the advances in query languages, the templates have to follow these updates and be constantly updated to answer real questions. Also, the biggest mistake of our approach is to differentiate templates with 2 and 3 hops. These errors occur as the KB schema was not included as inputs for the Hereditary Tree-LSTM. However, the results and the errors are limited to the chosen evaluated bases and the number of questions in the base. Although it is a large dataset, is still very far from the number of possible questions. So, the results can vary greatly in a real-world system, as the questions in the evaluation dataset are built without grammatical errors, and that, in practice, the use of this type of system needs to deal with ambiguous questions, grammatical errors, and so on.

4.2 CONTRIBUTIONS

The main contributions of this work are:

- We presented a systematic mapping of the C-KBQA field. The study presents an overview of how C-KBQA systems usually work. The identification of the most used methods, datasets, knowledge bases, metrics, and domains in the complex question answering scenario in the literature was presented and the main gaps for C-KBQA systems were discussed. A protocol was adopted for the execution of systematic mapping to reduce the bias and make the study reproducible for other researchers, presented all the steps followed to answer the research questions. Also, we made all the data used in the systematic mapping available for use in future mappings. Thus, new researchers can continue the research based on where our mapping ended.

- A template matching system using the combination of Semantic Parsing and Neural Networks techniques to determine the answer template that a complex question maths was created. Moreover, an attention mechanism was created to assist the neural network in selecting the most important information. In the so-called Hereditary Attention, each neural network cell inherits the attention from another neural network cell, in a bottom-up way. Our attention system proved to be adequate for the template matching problem. However, Tree-LSTM has shown good results in semantic relatedness and sentiment classification of texts problems. So, the use of HTL may also be adequate for these scenarios.

- A new version of an LC-QuAD 2.0 containing answer templates called LC-QuAD 2.1 was released. This new version presents a cleaned version of the LC-QuAD 2.0, whiteout duplicated questions, malformed questions, and other problems that could be caused for the use of the crowdsourcing approach. Besides, we created and mapped dummy answer templates to all questions in the LC-QuAD 2.1. Finally, the training, development, and testing sets used in this work were released so futures works in C-KBQA can use to compare their approach with the system presented in this master thesis.

4.3 FUTURE WORK

This work opens a range of possibilities for future research. The use of the HTL still needs to be further investigated and additional information may improve the accuracy of the approach for C-KBQA. The inclusion of the KB schema information on training can improve the ability of HTL to distinguish the number of hops that a question needs and so achieve more accurate results. Also, the use of Named Entity Recognition and Disambiguation and relation recognition methods to detect KB entities/relations and measure the distance between them can improve the results of our template matching. Furthermore, exploring HTL for other semantic tasks such as sentiment classification and semantic relatedness may also archive good results.

Evaluating HTL as part of a full C-KBQA system (question parsing, question representation, and candidate ranking) still have to be analyzed. The HTL can be coupled in a C-KBQA full system where the HTL would be used in the question representation. HTL can perform the selection of the question type and the identification of the main subjects. To answer a question, the answer template selected for the HTL is filled and the candidates ranked. However, the computational cost is one of the main problems in this step as a question can generate a high level of candidates list.

REFERENCES

ABDELKAWI, A. et al. Complex query augmentation for question answering over knowledge graphs. In: SPRINGER. **OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"**. [S.l.], 2019. p. 571–587.

ABUJABAL, A. et al. Automated template generation for question answering over knowledge graphs. In: **Proceedings of the 26th international conference on world wide web**. [S.l.: s.n.], 2017. p. 1191–1200.

AGARWAL, P.; RAMANATH, M.; SHROFF, G. Retrieving relationships from a knowledge graph for question answering. In: AZZOPARDI, L. et al. (Ed.). **Advances in Information Retrieval**. Cham: Springer International Publishing, 2019. p. 35–50. ISBN 978-3-030-15712-8.

AHMED, M.; SAMEE, M. R.; MERCER, R. E. Improving tree-lstm with tree attention. In: IEEE. **2019 IEEE 13th International Conference on Semantic Computing (ICSC)**. [S.l.], 2019. p. 247–254.

ANDERSON, M.; GóMEZ-RODRíGUEZ, C. **Distilling Neural Networks for Greener and Faster Dependency Parsing**. 2020.

ATHREYA, R. G. et al. Template-based question answering using recursive neural networks. In: IEEE. **2021 IEEE 15th International Conference on Semantic Computing (ICSC)**. [S.l.], 2021. p. 195–198.

AUER, S. et al. Dbpedia: A nucleus for a web of open data. In: **The semantic web**. [S.l.]: Springer, 2007. p. 722–735.

BAKHSHI, M. et al. Data-driven construction of sparql queries by approximate question graph alignment in question answering over knowledge graphs. **Expert Systems with Applications**, Elsevier, v. 146, p. 113205, 2020.

BAO, J. et al. Constraint-based question answering with knowledge graph. In: **Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers**. Osaka, Japan: The COLING 2016 Organizing Committee, 2016. p. 2503–2514. Disponível em: <https://www.aclweb.org/anthology/C16-1236>.

BAST, H.; HAUSSMANN, E. More accurate question answering on freebase. In: **Proceedings of the 24th ACM International on Conference on Information and Knowledge Management**. [S.l.: s.n.], 2015. p. 1431–1440.

BERANT, J. et al. Semantic parsing on freebase from question-answer pairs. In: **Proceedings of the 2013 conference on empirical methods in natural language processing**. [S.l.: s.n.], 2013. p. 1533–1544.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The semantic web. **Scientific american**, JSTOR, v. 284, n. 5, p. 34–43, 2001.

BHUTANI, N.; ZHENG, X.; JAGADISH, H. V. Learning to answer complex questions over knowledge bases with query composition. In: **Proceedings of the 28th ACM International Conference on Information and Knowledge Management**. New York, NY, USA: Association for Computing Machinery, 2019. (CIKM '19), p. 739–748. ISBN 9781450369763. Disponível em: <https://doi.org/10.1145/3357384.3358033>.

BHUTANI, N. et al. Answering complex questions by combining information from curated and extracted knowledge bases. In: **Proceedings of the First Workshop on Natural Language Interfaces**. Online: Association for Computational Linguistics, 2020. p. 1–10. Disponível em: <https://www.aclweb.org/anthology/2020.nli-1.1>.

BIZER, C. et al. Linked data-the story so far. **Semantic services, interoperability and web applications: emerging concepts**, IGI Global, v. 1, n. 1, p. 205–227, 2011.

BOJANOWSKI, P. et al. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 5, p. 135–146, 2017.

BOLLACKER, K.; COOK, R.; TUFTS, P. Freebase: A shared database of structured general human knowledge. In: **AAAI**. [S.l.: s.n.], 2007. v. 7, p. 1962–1963.

BOLLACKER, K. et al. Freebase: a collaboratively created graph database for structuring human knowledge. In: **Proceedings of the 2008 ACM SIGMOD international conference on Management of data**. [S.l.: s.n.], 2008. p. 1247–1250.

BORDES, A. et al. Large-scale simple question answering with memory networks. **arXiv preprint arXiv:1506.02075**, abs/1506.02075, 2015.

BOSSELUT, A.; BRAS, R. L.; CHOI, Y. **Dynamic Neuro-Symbolic Knowledge Graph Construction for Zero-shot Commonsense Question Answering**. 2020.

CAI, Q.; YATES, A. Large-scale semantic parsing via schema matching and lexicon extension. In: **Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)**. [S.l.: s.n.], 2013. p. 423–433.

CHEN, A. et al. Evaluating question answering evaluation. In: **Proceedings of the 2nd Workshop on Machine Reading for Question Answering**. Hong Kong, China: Association for Computational Linguistics, 2019. p. 119–124. Disponível em: <https://www.aclweb.org/anthology/D19-5817>.

CHEN, X.; JIA, S.; XIANG, Y. A review: Knowledge reasoning over knowledge graph. **Expert Systems with Applications**, v. 141, p. 112948, 2020. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0957417419306669>.

COOPER, I. D. What is a "mapping study?". **Journal of the Medical Library Association: JMLA**, Medical Library Association, v. 104, n. 1, p. 76, 2016.

COSTA, C.; MURTA, L. Version control in distributed software development: A systematic mapping study. In: IEEE. **8th International Conference on Global Software Engineering (ICGSE)**. [S.l.], 2013. p. 90–99.

CROFT, W. B.; METZLER, D.; STROHMAN, T. **Search engines: Information retrieval in practice**. USA: Addison-Wesley Publishing, 2010. v. 520.

CUI, W. et al. Kbqa: Learning question answering over qa corpora and knowledge bases. **Proc. VLDB Endow.**, VLDB Endowment, v. 10, n. 5, p. 565–576, jan. 2017. ISSN 2150-8097. Disponível em: <https://doi.org/10.14778/3055540.3055549>.

CUI, W. et al. KBQA: learning question answering over QA corpora and knowledge bases. **CoRR**, abs/1903.02419, p. 565–576, 2019.

DETTMERS, T. et al. Convolutional 2d knowledge graph embeddings. In: MCILRAITH, S. A.; WEINBERGER, K. Q. (Ed.). **Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018**. New Orleans, Louisiana, USA: AAAI Press, 2018. p. 1811–1818.

DIEFENBACH, D. et al. Towards a question answering system over the semantic web. **Semantic Web**, IOS Press, p. 1–19, 2020.

DIEFENBACH, D. et al. Core techniques of question answering systems over knowledge bases: a survey. **Knowledge and Information systems**, Springer, v. 55, n. 3, p. 529–569, 2018.

DILEEP, A. K. et al. Template-based question answering analysis on the lc-quad2.0 dataset. In: **2021 IEEE 15th International Conference on Semantic Computing (ICSC)**. [S.l.: s.n.], 2021. p. 443–448.

DIMITRAKIS, E.; SGONTZOS, K.; TZITZIKAS, Y. A survey on question answering systems over linked data and documents. **Journal of Intelligent Information Systems**, Springer, v. 55, p. 1–27, 2019.

DING, J. et al. Leveraging frequent query substructures to generate formal queries for complex question answering. In: **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**. Hong Kong, China: Association for Computational Linguistics, 2019. p. 2614–2622. Disponível em: <https://www.aclweb.org/anthology/D19-1263>.

DIOMEDI, D.; HOGAN, A. Question answering over knowledge graphs with neural machine translation and entity linking. **arXiv preprint arXiv:2107.02865**, 2021.

DUBEY, M. et al. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In: SPRINGER. **International Semantic Web Conference**. [S.l.], 2019. p. 69–78.

EVSEEV, D. A.; ARKHIPOV, M. Y. Sparql query generation for complex question answering with bert and bilstm-based model. In: **Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference "Dialogue 2020"**. [S.l.: s.n.], 2020. v. 2020-June, p. 270–282.

FERNANDEZ-SOTOS, P. et al. Social cognition remediation interventions: A systematic mapping review. **PloS one**, Public Library of Science, v. 14, n. 6, 2019.

GOMES JR., J. et al. **LC-QuAD 2.1**. Zenodo, 2021. Disponível em: <https://doi.org/10.5281/zenodo.5508297>.

GREENHALGH, T.; PEACOCK, R. Effectiveness and efficiency of search methods in systematic reviews of complex evidence: audit of primary sources. **BMJ**, BMJ Publishing Group Ltd, v. 331, n. 7524, p. 1064–1065, 2005. ISSN 0959-8138.

GU, G. et al. Learning to answer complex questions with evidence graph. In: SPRINGER. **Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data**. [S.l.], 2020. p. 257–269.

GUSENBAUER, M.; HADDAWAY, N. R. Which academic search systems are suitable for systematic reviews or meta-analyses? evaluating retrieval qualities of google scholar, pubmed, and 26 other resources. **Research synthesis methods**, Wiley Online Library, v. 11, n. 2, p. 181–217, 2020.

HADDAWAY, N. R. et al. The benefits of systematic mapping to evidence-based environmental management. **Ambio**, Springer, v. 45, n. 5, p. 613–620, 2016.

HAN, W. et al. **A Survey of Unsupervised Dependency Parsing**. 2020.

HAO, Z. et al. A subgraph-representation-based method for answering complex questions over knowledge bases. **Neural Networks**, Elsevier, v. 119, p. 57–65, 2019.

HÖFFNER, K. et al. Survey on challenges of question answering in the semantic web. **Semantic Web**, IOS Press, v. 8, n. 6, p. 895–920, 2017.

HONG, M.-D. et al. Temporal ontology representation and reasoning using ordinals and sets for historical events. In: SPRINGER. **Asian Conference on Intelligent Information and Database Systems**. [S.l.], 2016. p. 75–85.

HU, S. et al. Answering natural language questions by subgraph matching over knowledge graphs. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, v. 30, n. 5, p. 824–837, 2017.

HU, S.; ZOU, L.; ZHANG, X. A state-transition framework to answer complex questions over knowledge base. In: **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing**. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 2098–2108. Disponível em: <https://www.aclweb.org/anthology/D18-1234>.

HUA, Y. et al. Few-shot complex knowledge base question answering via meta reinforcement learning. In: **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Online: Association for Computational Linguistics, 2020. p. 5827–5837. Disponível em: <https://www.aclweb.org/anthology/2020.emnlp-main.469>.

HUA, Y. et al. Retrieve, program, repeat: Complex knowledge base question answering via alternate meta-learning. In: BESSIERE, C. (Ed.). **Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20**. Virtual, Japan: International Joint Conferences on Artificial Intelligence Organization, 2020. p. 3679–3686. Main track. Disponível em: <https://doi.org/10.24963/ijcai.2020/509>.

HUA, Y. et al. Less is more: Data-efficient complex question answering over knowledge bases. **Journal of Web Semantics**, Elsevier, p. 100612, 2020.

IMAM, M. et al. **Improvements to the complex question answering models**. Tese (Doutorado) — Lethbridge, Alta.: University of Lethbridge, c2011, 2011.

JI, S. et al. A survey on knowledge graphs: Representation, acquisition and applications. **arXiv preprint arXiv:2002.00388**, abs/2002.00388, 2020. Disponível em: <https://arxiv.org/abs/2002.00388>.

JIA, R.; LIANG, P. Adversarial examples for evaluating reading comprehension systems. In: **Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing**. Vancouver, Canada: Association for Computational Linguistics, 2017. p. 2021–2031.

JIA, Z. et al. Tequila: Temporal question answering over knowledge bases. In: **Proceedings of the 27th ACM International Conference on Information and Knowledge Management**. [S.l.: s.n.], 2018. p. 1807–1810.

JIN, H. et al. Comqa: Question answering over knowledge base via semantic matching. **IEEE Access**, IEEE, v. 7, p. 75235–75246, 2019.

JOULIN, A. et al. **Fast Linear Model for Knowledge Graph Embeddings**. [S.l.]: OpenReview.net, 2017.

KARTSAKLIS, D.; PILEHVAR, M. T.; COLLIER, N. Mapping text to knowledge graph entities using multi-sense LSTMs. In: **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing**. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 1959–1970. Disponível em: <https://www.aclweb.org/anthology/D18-1221>.

KITCHENHAM, B. Procedures for performing systematic reviews. **Keele, UK, Keele University**, v. 33, n. 2004, p. 1–26, 2004.

LEHMANN, J. et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. **Semantic web**, IOS Press, v. 6, n. 2, p. 167–195, 2015.

LI, X.; HU, S.; ZOU, L. Natural answer generation via graph transformer. In: SPRINGER. **Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data**. [S.l.], 2020. p. 302–318.

LIANG, C. et al. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (ACL). **55th Annual Meeting of the Association for Computational Linguistics, ACL 2017**. [S.l.], 2017. p. 23–33.

LIN, B. Y. et al. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In: INUI, K. et al. (Ed.). **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019**. Association for Computational Linguistics, 2019. p. 2829–2839. Disponível em: <https://doi.org/10.18653/v1/D19-1282>.

LIU, Y. et al. Roberta: A robustly optimized bert pretraining approach. **arXiv preprint arXiv:1907.11692**, 2019.

LIU, Y. et al. **KG-BART: Knowledge Graph-Augmented BART for Generative Commonsense Reasoning**. 2020.

LU, X. et al. Answering complex questions by joining multi-document evidence with quasi knowledge graphs. In: **Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval**. [S.l.: s.n.], 2019. p. 105–114.

LUKOVNIKOV, D.; FISCHER, A.; LEHMANN, J. Pretrained transformers for simple question answering over knowledge graphs. In: SPRINGER. **International Semantic Web Conference**. 2019. abs/2001.11985, p. 470–486. Disponível em: <https://arxiv.org/abs/2001.11985>.

LUO, K. et al. Knowledge base question answering via encoding of complex query graphs. In: **Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing**. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 2185–2194.

MAHESHWARI, G. et al. Learning to rank query graphs for complex question answering over knowledge graphs. In: SPRINGER. **International Semantic Web Conference**. [S.l.], 2019. p. 487–504.

MANNING, C. D. et al. The stanford corenlp natural language processing toolkit. In: **Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations**. [S.l.: s.n.], 2014. p. 55–60.

MENDES, P. N. et al. Dbpedia spotlight: shedding light on the web of documents. In: **Proceedings of the 7th international conference on semantic systems**. New York, NY, USA: Association for Computing Machinery, 2011. p. 1–8.

MEYER, B. et al. Viewpoint research evaluation for computer science. **Communications of the ACM**, ACM New York, NY, USA, v. 52, n. 4, p. 31–34, 2009.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, v. 2013, 2013.

MILLER, A. et al. Key-value memory networks for directly reading documents. In: **Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing**. Austin, Texas: Association for Computational Linguistics, 2016. p. 1400–1409.

MIYAZAKI, R.; KOMACHI, M. Japanese sentiment classification using a tree-structured long short-term memory with attention. In: **Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation**. [S.l.: s.n.], 2018.

NAGARHALLI, T. P.; VAZE, V.; RANA, N. K. A review of current trends in the development of chatbot systems. In: **2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)**. Coimbatore, India: IEEE, 2020. p. 706–710.

NAMAKI, M. H. et al. Learning to speed up query planning in graph databases. **Proceedings of the International Conference on Automated Planning and Scheduling**, v. 27, n. 1, p. 9, Jun. 2017. Disponível em: <https://ojs.aaai.org/index.php/ICAPS/article/view/13849>.

NEIVA, F. W. et al. Towards pragmatic interoperability to support collaboration: A systematic review and mapping of the literature. **Information and Software Technology**, Elsevier, v. 72, p. 137–150, 2016.

NOUAR, W.; BOUFAIDA, Z. Deep semantic parsing method to capture the structure of multi-relation questions over linked data. In: **CEUR Workshop Proceedings**. [s.n.], 2018. v. 2379, p. 5–14. Disponível em: <www.scopus.com>.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: **Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)**. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1532–1543.

PERRYMAN, C. L. Mapping studies. **Journal of the Medical Library Association: JMLA**, Medical Library Association, v. 104, n. 1, p. 79, 2016.

PETERSEN, K. et al. Systematic mapping studies in software engineering. In: **12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12**. [S.l.: s.n.], 2008. p. 1–10.

PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. **Information and Software Technology**, Elsevier, v. 64, p. 1–18, 2015.

PETTICREW, M.; ROBERTS, H. Systematic reviews in the social sciences: a practical guide. 2006. **Malden USA: Blackwell Publishing CrossRef Google Scholar**, v. 6, p. 304–305, 2006.

PINXTEREN, M. M. V.; PLUYMAEKERS, M.; LEMMINK, J. G. Human-like communication in conversational agents: a literature review and research agenda. **Journal of Service Management**, Emerald Publishing Limited, ahead-of-print, p. 203–225, 2020.

POPPING, R. Knowledge graphs and network text analysis. **Social Science Information**, Sage Publications London, v. 42, n. 1, p. 91–106, 2003.

QIU, Y. et al. Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In: **Proceedings of the 13th International Conference on Web Search and Data Mining**. [S.l.: s.n.], 2020. p. 474–482.

QIU, Y. et al. Hierarchical query graph generation for complex question answering over knowledge graph. In: **Proceedings of the 29th ACM International Conference on Information & Knowledge Management**. [S.l.: s.n.], 2020. p. 1285–1294.

RADOEV, N. et al. A language adaptive method for question answering on french and english. In: SPRINGER. **Semantic Web Evaluation Challenge**. [S.l.], 2018. p. 98–113.

RAJABI, E.; SANCHEZ-ALONSO, S. Knowledge discovery using sparql property path: The case of disease data set. **Journal of Information Science**, SAGE Publications Sage UK: London, England, v. 0, p. 0165551519865495, 2019.

RAMESH, K. et al. A survey of design techniques for conversational agents. In: KAUSHIK, S. et al. (Ed.). **Information, Communication and Computing Technology**. Singapore: Springer Singapore, 2017. p. 336–350. ISBN 978-981-10-6544-6.

REDDY, A. C. O.; MADHAVI, K. Convolutional recurrent neural network with template based representation for complex question answering. **International Journal of Electrical and Computer Engineering**, IAES Institute of Advanced Engineering and Science, v. 10, n. 3, p. 2710, 2020.

RODRIGO, A.; PENAS, A. A study about the future evaluation of question-answering systems. **Knowledge-Based Systems**, Elsevier, v. 137, p. 83–93, 2017.

SAHA, A. et al. Complex program induction for querying knowledge bases in the absence of gold programs. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 7, p. 185–200, 2019.

SAHA, A. et al. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. **Proceedings of the AAAI Conference on Artificial Intelligence**, v. 32, n. 1, p. 9, Apr. 2018. Disponível em: <https://ojs.aaai.org/index.php/AAAI/article/view/11332>.

SAKOR, A. et al. Falcon 2.0: An entity and relation linking tool over wikidata. In: **Proceedings of the 29th ACM International Conference on Information & Knowledge Management**. New York, NY, USA: Association for Computing Machinery, 2020. (CIKM '20), p. 3141–3148. ISBN 9781450368599. Disponível em: <https://doi.org/10.1145/3340531.3412777>.

SAKOR, A.; SINGH, K.; VIDAL, M.-E. Falcon: An entity and relation linking framework over dbpedia. In: AACHEN: RWTH. **CEUR Workshop Proceedings 2456 (2019)**. [S.l.], 2019. v. 2456, p. 265–268.

SCHUMACHER, E.; MAYFIELD, J.; DREDZE, M. **Cross-Lingual Transfer in Zero-Shot Cross-Language Entity Linking**. 2020.

SHEN, T. et al. Effective search of logical forms for weakly supervised knowledge-based question answering. In: BESSIERE, C. (Ed.). **Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20**. International Joint Conferences on Artificial Intelligence Organization, 2020. p. 2227–2233. Main track. Disponível em: <https://doi.org/10.24963/ijcai.2020/308>.

SHI, P.; LIN, J. **Simple BERT Models for Relation Extraction and Semantic Role Labeling**. 2019.

SHIN, S.; LEE, K.-H. Processing knowledge graph-based complex questions through question decomposition and recomposition. **Information Sciences**, Elsevier, v. 523, p. 234–244, 2020.

SILVA, J. W. F. da et al. A short survey on end-to-end simple question answering systems. **Artificial Intelligence Review**, Springer, v. 53, n. 7, p. 5429–5453, 2020.

SINGH, K. et al. Why reinvent the wheel: Let's build question answering systems together. In: **Proceedings of the 2018 World Wide Web Conference**. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018. (WWW '18), p. 1247–1256. ISBN 9781450356398. Disponível em: <https://doi.org/10.1145/3178876.3186023>.

TAI, K. S.; SOCHER, R.; MANNING, C. D. Improved semantic representations from tree-structured long short-term memory networks. **arXiv preprint arXiv:1503.00075**, 2015.

TALMOR, A.; BERANT, J. The web as a knowledge-base for answering complex questions. In: **Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)**. [S.l.: s.n.], 2018. p. 641–651.

TONG, P.; ZHANG, Q.; YAO, J. Leveraging domain context for question answering over knowledge graph. **Data Science and Engineering**, Springer, v. 4, n. 4, p. 323–335, 2019.

TRAN, T.-K. et al. Fast computation of explanations for inconsistency in large-scale knowledge graphs. In: **Proceedings of The Web Conference 2020**. New York, NY, USA: Association for Computing Machinery, 2020. (WWW '20), p. 2613–2619. ISBN 9781450370233. Disponível em: <https://doi.org/10.1145/3366423.3380014>.

TRIVEDI, P. et al. Lc-quad: A corpus for complex question answering over knowledge graphs. In: SPRINGER. **International Semantic Web Conference**. Cham: Springer International Publishing, 2017. p. 210–218.

VAKULENKO, S. et al. Message passing for complex question answering over knowledge graphs. In: **Proceedings of the 28th ACM International Conference on Information and Knowledge Management**. [S.l.: s.n.], 2019. p. 1431–1440.

VASWANI, A. et al. Attention is all you need. In: **Proceedings of the 31st International Conference on Neural Information Processing Systems**. Red Hook, NY, USA: Curran Associates Inc., 2017. (NIPS'17), p. 6000–6010. ISBN 9781510860964.

VRANDEČIĆ, D.; KRÖTZSCH, M. Wikidata: a free collaborative knowledgebase. **Communications of the ACM**, ACM New York, NY, USA, v. 57, n. 10, p. 78–85, 2014.

WANG, R. et al. Structured query construction via knowledge graph embedding. **Knowledge and Information Systems**, Springer, p. 1–28, 2019.

WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: **Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering**. New York, NY, USA: Association for Computing Machinery, 2014. (EASE '14). ISBN 9781450324762.

WU, L.; WU, P.; ZHANG, X. A seq2seq-based approach to question answering over knowledge bases. In: SPRINGER. **Joint International Semantic Technology Conference**. Singapore: Springer Singapore, 2019. p. 170–181.

WU, P.; ZHANG, X.; FENG, Z. A survey of question answering over knowledge base. In: SPRINGER. **China Conference on Knowledge Graph and Semantic Computing**. Singapore: Springer Singapore, 2019. p. 86–97.

XU, K. et al. Enhancing key-value memory neural networks for knowledge based question answering. 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, 2019. p. 2937–2947.

XU, K. et al. Question answering on freebase via relation extraction and textual evidence. In: **ACL (1)**. [s.n.], 2016. Disponível em: <http://aclweb.org/anthology/P/P16/P16-1220.pdf>.

Yang, S. et al. Fast top-k search in knowledge graphs. In: **2016 IEEE 32nd International Conference on Data Engineering (ICDE)**. Los Alamitos, CA, USA: IEEE Computer Society, 2016. p. 990–1001.

YANG, S. et al. **Second-Order Unsupervised Neural Dependency Parsing**. 2020.

YANG, Y.; CHANG, M.-W. S-MART: Novel tree-based structured learning algorithms applied to tweet entity linking. In: **Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. Beijing, China: Association for Computational Linguistics, 2015. p. 504–513. Disponível em: <https://www.aclweb.org/anthology/P15-1049>.

YIH, W.-t. et al. Semantic parsing via staged query graph generation: Question answering with knowledge base. In: **Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)**. [S.l.: s.n.], 2015. p. 1321–1331.

YIH, W.-t. et al. The value of semantic parse labeling for knowledge base question answering. In: **Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)**. [S.l.: s.n.], 2016. p. 201–206.

YIN, P. et al. Answering questions with complex semantic constraints on open knowledge bases. In: **Proceedings of the 24th ACM International on Conference on Information and Knowledge Management**. [S.l.: s.n.], 2015. p. 1301–1310.

YIN, W.; GE, W.; WANG, H. Cdqa: An ontology-based question answering system for chinese delicacy. In: **2014 IEEE 3rd International Conference on Cloud Computing and Intelligence Systems**. Shenzhen, China: IEEE, 2014. p. 1–7. ISSN 2376-595X.

YU, M. et al. Improved neural relation detection for knowledge base question answering. In: **ACL (1)**. [s.n.], 2017. p. 571–581. Disponível em: <https://doi.org/10.18653/v1/P17-1053>.

ZAFAR, H.; NAPOLITANO, G.; LEHMANN, J. Formal query generation for question answering over knowledge bases. In: SPRINGER. **European Semantic Web Conference**. [S.l.], 2018. p. 714–728.

ZHANG, H. et al. Self-attention generative adversarial networks. In: PMLR. **International conference on machine learning**. [S.l.], 2019. p. 7354–7363.

ZHANG, J. et al. Semantic parsing for multiple-relation chinese question answering. In: **CEUR Workshop Proceedings**. [s.n.], 2018. v. 2242, p. 101–106. Disponível em: <www.scopus.com>.

ZHANG, L.; WINN, J.; TOMIOKA, R. Gaussian attention model and its application to knowledge base embedding and question answering. **arXiv preprint arXiv:1611.02266**, 2016.

ZHANG, Q. et al. Mutual relation detection for complex question answering over knowledge graph. In: SPRINGER. **International Conference on Database Systems for Advanced Applications**. [S.l.], 2020. p. 623–631.

ZHANG, Y. et al. Variational reasoning for question answering with knowledge graph. **Proceedings of the AAAI Conference on Artificial Intelligence**, v. 32, n. 1, p. 6069–6076, Apr. 2018. Disponível em: <https://ojs.aaai.org/index.php/AAAI/article/view/12057>.

ZHENG, C.; KORDJAMSHIDI, P. SRLGRN: Semantic role labeling graph reasoning network. In: **Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Online: Association for Computational Linguistics, 2020. p. 8881–8891.

ZHENG, W. et al. Question answering over knowledge graphs: question understanding via template decomposition. **Proceedings of the VLDB Endowment**, VLDB Endowment, v. 11, n. 11, p. 1373–1386, 2018.

ZHOU, M.; HUANG, M.; ZHU, X. An interpretable reasoning network for multi-relation question answering. In: **Proceedings of the 27th International Conference on Computational Linguistics**. [S.l.: s.n.], 2018. p. 2010–2022.

APPENDIX A – SYSTEMATIC REVIEW PAPER LIST

Table 17 – Full list of papers

ID	Citation	ID	Citation
1	(YIN; GE; WANG, 2014)	2	(REDDY; MADHAVI, 2020)
3	(YIN et al., 2015)	4	(BAST; HAUSSMANN, 2015)
5	(MAHESHWARI et al., 2019)	6	(HONG et al., 2016)
7	(BAO et al., 2016)	8	(XU et al., 2016)
9	(ABUJABAL et al., 2017)	10	(MILLER et al., 2016)
11	(SHEN et al., 2020)	12	(CUI et al., 2017)
13	(HU et al., 2017)	14	(YU et al., 2017)
15	(RADOEV et al., 2018)	16	(NOUAR; BOUFAIDA, 2018)
17	(ZHENG et al., 2018)	18	(ZHANG et al., 2018)
19	(JIA et al., 2018)	20	(TALMOR; BERANT, 2018)
21	(LUO et al., 2018)	22	(ZAFAR; NAPOLITANO; LEHMANN, 2018)
23	(ZHOU; HUANG; ZHU, 2018)	24	(ZHANG et al., 2018)
25	(HAO et al., 2019)	26	(LU et al., 2019)
27	(SAHA et al., 2019)	28	(JIN et al., 2019)
29	(XU et al., 2019)	30	(BHUTANI; ZHENG; JAGADISH, 2019)
31	(VAKULENKO et al., 2019)	32	(AGARWAL; RAMANATH; SHROFF, 2019)
33	(TONG; ZHANG; YAO, 2019)	34	(ABDELKAWI et al., 2019)
35	(HU; ZOU; ZHANG, 2018)	36	(BHUTANI et al., 2020)
37	(BAKHSHI et al., 2020)	38	(DING et al., 2019)
39	(SHIN; LEE, 2020)	40	(QIU et al., 2020a)
41	(WANG et al., 2019)	42	(WU; WU; ZHANG, 2019)
43	(DIEFENBACH et al., 2020)	44	(SAHA et al., 2018)
45	(YIH et al., 2015)	46	(LIANG et al., 2017)
47	(HUA et al., 2020c)	48	(QIU et al., 2020b)
49	(GU et al., 2020)	50	(ZHANG et al., 2020)
51	(EVSEEV; ARKHIPOV, 2020)	52	(LI; HU; ZOU, 2020)
53	(HUA et al., 2020b)	54	(HUA et al., 2020a)

Source: Created by the author (2021)

APPENDIX B – ANSWER TEMPLATES ADDITIONAL DATA

This appendix section present the preprocessing and dummy template creation results on the Lc-QuAD 2.1. The results for DBpedia 2018 were divide into 3 Tables (Tables 18, 19, and 20). Read GID as group ID, TID as template ID, #Q as the number of questions in the template ID, and #TQ as the number of questions in the group when reading the columns names.

The relation between the Wikidata and DBpedia 2018 KB was also analyzed. We mapped the co-occurrence of Dummy SPARQL templates ids across the Wikidata e DBpedia 2018 for the full dataset. Table 21 (Appendix) present the co-occurrence of the templates ids. Most of the id of the template only has one common co-occurrence in Wikidata and DBpedia. In other words, when one Dummy SPARQL template is matched in any of the KB, it is possible to create a wrapper to translate the Dummy SPARQL templates of Wikidata into a Dummy SPARQL template of DBpedia 2018.

Table 18 – Questions distribution by dummy template and dummy group ids for the DBpedia KB.

GID	TID	DUMMY SPARQL	#Q	#TQ
0	0.1	ASK { ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . }	421	1912
	0.2	ASK { ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?obj. filter(?obj < DUMMY_F) }	239	
	0.3	ASK { ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?obj. filter(?obj = DUMMY_F) }	1027	
	0.4	ASK { ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?obj. filter(?obj > DUMMY_F) }	225	
1	1	ASK { ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . }	363	363
2	2.1	SELECT (COUNT(?sub) AS ?subs) {?statement DUMMY_P ?sub .?statement DUMMY_P DUMMY_O .?statement DUMMY_P DUMMY_P .}	547	960
	2.2	SELECT (COUNT(?obj) AS ?objs) {?statement DUMMY_P DUMMY_O .?statement DUMMY_P DUMMY_O .?statement DUMMY_P ?obj .}	413	
3	3.1	SELECT distinct ?answer where { ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P ?answer. }	2923	3580
	3.2	SELECT distinct ?answer where { ?statement DUMMY_P ?answer. ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . }	657	

Source: Created by the author (2021)

Table 19 – Questions distribution by dummy template and dummy group ids for the DBpedia KB part 2.

GID	TID	DUMMY SPARQL	#Q	#TQ
4	4.1	SELECT distinct ?value where {?statement DUMMY_P DUMMY_O .?statement DUMMY_P DUMMY_O .?statement DUMMY_P DUMMY_O .?statement DUMMY_P ?value. }	2442	8563
	4.2	SELECT distinct ?obj where {?statement DUMMY_P DUMMY_O .?statement DUMMY_P DUMMY_O .?statement DUMMY_P ?obj .?statement DUMMY_P DUMMY_O }	2540	
	4.3	SELECT distinct ?obj where { ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P ?obj . ?obj DUMMY_P DUMMY_O }	1546	
	4.4	SELECT distinct ?subj where { ?statement DUMMY_P ?subj . ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . ?subj DUMMY_P DUMMY_O }	1694	
	4.5	SELECT ?ent where {?ent DUMMY_P DUMMY_O .?statement DUMMY_P ?ent .?statement DUMMY_P DUMMY_O .?statement DUMMY_P ?obj .}ORDER BY DESC(?obj)LIMIT DUMMY_F	341	
5	5.1	SELECT ?answer WHERE {?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?X . ?statement2 DUMMY_P ?X. ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P ?answer .}	2911	4658
	5.2	SELECT ?answer WHERE {?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?answer . ?statement2 DUMMY_P ?answer. ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O .}	1747	
6	6.1	SELECT ?ent where {?ent DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?ent . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?obj1 . ?statement2 DUMMY_P ?ent . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . }ORDER BY DESC(?obj1)LIMIT DUMMY_F	284	569
	6.2	SELECT ?ent where {?ent DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?ent . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?obj1 . ?statement2 DUMMY_P ?ent . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O .}ORDER BY ASC(?obj1)LIMIT DUMMY_F	285	

Source: Created by the author (2021)

Table 20 – Questions distribution by dummy template and dummy group ids for the DBpedia KB part 3.

GID	TID	DUMMY SPARQL	#Q	#TQ
7	7.1	SELECT distinct ?dataprop1 ?obj where { ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P ?obj . ?statement DUMMY_P ?dataprop1 . }	493	2107
	7.2	SELECT DISTINCT ?sbj ?sbj_label { ?statement1 DUMMY_P ?sbj . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?sbj DUMMY_S ?sbj_label . FILTER(CONTAINS(lcase(?sbj_label), DUMMY_F)). FILTER(lang(?sbj_label) = DUMMY_F)} LIMIT DUMMY_F	802	
	7.3	SELECT DISTINCT ?sbj ?sbj_label {?statement1 DUMMY_P ?sbj . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?sbj DUMMY_S ?sbj_label . FILTER(STRSTARTS(lcase(?sbj_label),DUMMY_F)). FILTER(lang(?sbj_label) = DUMMY_F)} LIMIT DUMMY_F	812	
8	8	SELECT distinct ?value1 ?value2 where {?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P DUMMY_O . ?statement DUMMY_P ?value1 . ?statement DUMMY_P ?value2 }	448	448
9	9	SELECT ?ans_1 ?ans_2 WHERE {?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P ?ans_1. ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P ?ans_2. }	490	490
10	10.1	SELECT DISTINCT ?sbj ?sbj_label {?statement1 DUMMY_P ?sbj . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement2 DUMMY_P ?sbj . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . ?sbj DUMMY_S ?sbj_label . FILTER(CONTAINS(lcase(?sbj_label),DUMMY_F)). FILTER(lang(?sbj_label)=DUMMY_F)} LIMIT DUMMY_F	198	484
	10.2	SELECT DISTINCT ?sbj ?sbj_label {?statement1 DUMMY_P ?sbj . ?statement1 DUMMY_P DUMMY_O . ?statement1 DUMMY_P DUMMY_O . ?statement2 DUMMY_P ?sbj . ?statement2 DUMMY_P DUMMY_O . ?statement2 DUMMY_P DUMMY_O . ?sbj DUMMY_S ?sbj_label . FILTER(STRSTARTS(lcase(?sbj_label), DUMMY_F)). FILTER(lang(?sbj_label) = DUMMY_F)} LIMIT DUMMY_F	286	

Source: Created by the author (2021)

Table 21 – Dummy SPARQL Template co-occurrences mapping.

Wikidata_ID	DBpedia_ID
0.1	0.1
0.2	0.3
0.3	0.4
0.4	0.2
1	1
2.1	3.2
2.2	3.1
3.1	2.1
3.2	2.2
4.1	5.1
4.2	5.2
4.3	4.4
4.4	4.3
4.5	4.5
5.1	4.2
5.2	4.1
6.1	6.2
6.2	6.1
7.1	4.1
7.2	4.1
7.3	4.2
7.4	4.2
8	9
9.1	7.2
9.2	7.3
10	7.1
11.1	10.1
11.2	10.2
12	8

Source: Created by the author (2021)