Knowledge Context for Entity and Relation Linking

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

von
Isaiah Mulang' Onando
(**Isaiah Onando Mulang'**)

aus
Siaya, Kenya

Bonn, 15.04.2021

Dieser Forschungsbericht wurde als Dissertation von der Mathematisch-Naturwissenschaftlichen Fakultät der Universität Bonn angenommen und ist auf dem Hochschulschriftenserver der ULB Bonn https://nbn-resolving.org/urn:nbn:de:hbz:5-63968 elektronisch publiziert.

1. Gutachter:	Prof. Dr. Sören Auer
2. Gutachter:	Prof. Dr. Jens Lehmann
Tag der Promotion:	01.09.2021
Erscheinungsjahr:	2021

Abstract

Knowledge graphs (KGs) are structures that provide a compendious representation of real world facts about entities and their relationships. The last decade has seen an increase in the number, size and application of knowledge graphs especially owing to the easy accessibility of the World Wide Web as a knowledge store. Adding structure to this data implies that machines can easily interpret, reason with, and infer meanings across different domains. Such rich stores of structured data have been proven to boost performances in core Natural Language Processing (NLP) tasks such as Relation Extraction, Question Answering, Dialog Systems, Web Search, etc. Furthermore, owing to these vast structured knowledge stores new research and application areas have emerged, viz. automatic KG construction, KG completion, and KG Alignment. Central to these tasks is the need to align entities and their relations in text to equivalents in referent knowledge bases. However, the difference in representation of such relations within unstructured text as compared to the formally structured knowledge bases manifest major challenges namely: lexical gap, ambiguity, complex and implicit relations, the unpredictability of natural language vs formulaic knowledge bases, and complex grammar used in text etc. Numerous research efforts have sought to provide tools and approaches for text to KG disambiguation. Notwithstanding, the aforementioned challenges still remain obstacles to overcome.

This thesis makes two considerations to address entity and relation linking. We envision tools that harness both the power of deep learning methods as well as traditional Artificial Intelligence techniques. We also view the KG as a source of information that can be anchored as features to inform machine learning models. In this view, we propose encoding this curated information for the linking models. We first devise an approach called ReMatch to perform end-to-end relation linking. ReMatch represents essential attributes of the relations in short text and models KG relations in a complementary structure to enhance the similarity scoring process. A terminology graph is used to augment these two structures with synonym relations. Next, we perform end-to-end entity linking via an attention-based encoder-decoder neural network that captures signals from a infused background KG. In this context, our approach Arjun is a first attempt to encode entity information from Wikidata KG as contextual signals in a neural network architecture. There are two neural encoders used in Arjun, where the first one recognises entity mentions. We create a local KG infused from two open domain KGs to associate entities with their aliases. The infused KG is used to power another encoder network for the disambiguation. In a subsequent implementation, We extend the Arjun idea to perform end-to-end entity linking by leveraging the power of the state-of-the-art transformers. A fine-tuned transformer model recognises entity mentions in text, but allows for a mix-and-match approach to the candidate generation step. We then utilise entity descriptions in a second transformer model for disambiguation. In another direction, we experiment with KG triples to evaluate the impact of KG context on transformer models. We desire to unearth underlying nuances in KG entities, and define appropriate representations of the same for the learning models. This work provides insightful results for the community on types, encoding and extent of KG context for NLP. Finally, we employ the novel intuition gained to enhance a model for the explanation regeneration task in elementary science QA. Our contributions target a broader research agenda by providing efficient approaches that leverage information in KGs, and to propel efforts that obtain best of both KGs and NLP.

Acknowledgements

There are numerous individuals and groups that have formed part of the journey through my P.h.D studies providing support and guidance, without which this work would not have been possible. I want to recognise the support in form of PhD research scholarship that I received from the *"Deutscher Akademischer Austauschdienst"* (DAAD) as well as the National Research Fund of Kenya (NRF-Kenya) under the DAAD-NRF program, that enabled me to stay and perform my research in Germany at the University of Bonn. Subsequently, I recognise the *Technische Informationsbibliothek* (TIB) for offering me a completion grant to finalize my PhD.

I am grateful to my supervisor Prof. Dr Sören Auer for providing me with an opportunity to do a thesis under his guidance. When I interviewed in 2015, and he decided to support my DAAD scholarship application, he conferred a belief that acted as a vote of confidence and has propelled me to not only carry my research but to also fulfil such faith. I have been very lucky to have him as my supervisor. His trust in my potentials and capabilities and his admirable leadership have always inspired me to excel in research over time. Even though he had to change employers and shift work location to the city of Hanover during the early stages of my work, I appreciate that he still maintained an unlimited interest in my work: always motivating and checking the areas in which I needed support. Above the great respect, I reserve great gratitude for Prof. Dr Sören Auer. Similarly, I am thankful to Prof. Dr Jens Lehmann for his support provided during the course of this thesis through his leadership by example approach. Being able to work at the SDA group of Uni Bonn, and interact with his level of discipline and commitment to work, was a learning experience. His valuable advices on many critical issues, especially on academic writing and research ideas helped define my research tasks.

Dr Kuldeep Singh has been more than a friend, and coauthor. To see him get through his PhD research and graduate in the rare 3 years time, then transition into a great leadership role in research and industry is something of a spectacle. I have spent most of my PhD time with him as a fellow PhD student, colleague, and as well as research mentor. I am truly grateful for his support that goes beyond academia into friendship. I am certain there is much I have gained just because I met Dr Kuldeep Singh and the collaboration we have had. I clear this stage of my profession with confidence that the foundation we have laid down for research with initiatives such as Zerotha research, will prosper into greater heights. I tip my hat to you Kuldeep for your vision and diligence, and I am happy to have interacted with your approach to research.

Special thanks go to the three postdoctoral researchers in my PhD journey: Dr Fabrizio Orlandi, Dr Simon Scerri, and Dr Jennifer D'Souza. Dr Fabrizio Orlandi and Dr Simon Scerri became part of my PhD journey way before I joined the University of Bonn. While I was writing my research proposal, they offered me tremendous support out of their busy schedules without surety on the success of their efforts. Dr Fabrizio would then be my direct supervisor in my first years as a PhD candidate, he (Dr Orlandi) forms the first research school that I attended. I salute you, Fabrizio because beyond being an academic advisor, I would share my challenges in settling in Germany and your advises became vital in this stage. Although you left for Ireland in between, the foundation of research you offered was sufficient to propel me onward. I recognise the work we have done with Dr Simon Scerri especially on industrial

projects at Fraunhofer IAIS and to be able to learn from his leadership skills. I extend my gratitude to Dr Jennifer D'Souza for her mentorship over the last phase of my work. Her push and determination to get results triggered me out of my comfort zone and allowed me not only to experiment but to learn new technologies. Your keenness on reproducing results was a new research direction that added to my view of research.

There are many influential leaders I have worked with along the way as well. Dr Johannes Hoffart has been a research mentor out of a free will, and I appreciate his guidance and professionalism that taught me how to approach research. His constant reevaluation of my work and results, and insistence on quality to target top conferences, helped me achieve great heights. Prof. Dr Andreas Both has offered me good mentorship especially concerning how I can gain a hybrid between research and industry and the process to and through the Software Campus; I view your career path as an example that I would like to emulate. I wish to thank Prof. Maria-Esther Vidal for her willingness to collaborate and offer guidance especially considering I was working on tasks not directly related to her core interests. Again Prof. Maria-Esther would be one of the research schools I learnt from, her insistence on articulating: "the what, the why, and the how" of research helped ground my lofty ideas into concrete research concepts. My direct supervisors including heads of department at EIS including Dr Christoph Lange, who first welcomed me to the department when I joined, together with Dr Steffen Lohmann. In the moments I needed support they offered unparalleled advice and help. In the same spirit, I want to thank Dr Giulio Napolitano and Dr Ioanna Lytra for their support when we worked on several projects.

I wish to recognise the role of the Software Campus during my PhD journey. I viewed this program to be in line with my professional vision, and I acknowledge that I have grown in the program into a better leader. The leadership workshops provided me with great skills and practice (like Agility, design thinking, personal awareness, team dynamics, organisation change and personnel dynamics) that are needed in both research and business projects. Working with the DATEV team Stefan Mehlen and Ahmad Jeffrey, as well as the team at EIT ICT Labs including Stefan Jazdzejewski, and Susanne Kegler made the program very smooth and interesting. In the same breadth, I recognise fellow participants and alumni of the program and my student assistants Ravikant Tyagi, and Sraya Reddy.

I further extend thanks to my friends top of which is Najmeh Nejad Mousavi. Najmeh has been close to me like a sister, giving advises especially on administrative processes. Likewise she was a constant team member in the projects I have been involved at Fraunhofer IAIS and I enjoyed working together. I am very proud of the best teams that I worked with on research projects that include: Akhilesh Vyash, Abhishek Nadgeri, Anson Bastos, Chaitali Prabhu, Manoj Probhakar, Mohamad Yaser Jaradeh and Ahmad Sakor. Thanks to all of you for the support and good time. We have complemented each other in the many tasks towards research, and that has been our strength to target very challenging problems successfully. I would also thank Dr. Saeedeh Shekarpour for being a constant co-author, always sparing time to assist my team and give valuable insights and never withholding any honest opinion that would improve my work.

In Addition, I would like to thank my family, my beloved spouse Lavender Achieng' for tolerating my PhD journey and struggles, my daughter Zuri Onando, for bringing the light moments of the journey and a motivation to press on. Seeing you grow and looking up to me gave me extra strength. I thank my mother, mama Anastasia Achieng' for her love and care through life and though she were not near me during the specific PhD period, I am certain her prayers protected and pushed me through it all. Thanks to my siblings Daniel Mulang', Martin Mulang', Catherine Mulang', and Joyce Mulang'. The journey from a small rural village in Kenya, through the education system, till Bonn was never easy, neither was it necessarily a definite personal plan. Your encouragement when the times got rough are much appreciated and duly noted.

This Ph.D. thesis is dedicated to the three ladies in my life: my mother Anastasia Achieng', my partner Lavender Achieng', and my daughter Zuri Onando. Love you all

Contents

CHAPTER 1

Introduction

Knowledge bases have been used for a long time to assist in reasoning and inference concerning decision support and to provide features for Artificial Intelligence (AI) algorithms. However, the last decade has seen two significant computing trends: i) Tremendous publishing of data on the web, and ii) the rise of powerful algorithmic techniques for data consumption. This transition into the Big Data era has given rise to better information structuring to enable machines to understand and process. As more and more data gets available in the various information sources, it has become vital for researchers and tech companies to seek a structured version of the data to get hidden insights. It is also essential that data be freely used and distributed. Further publishing of data ought to follows a set of design principles for sharing machine-readable interlinked data on the Web. Several data formats have been proposed over the years, starting from PDFs, XML, CSVs, and Linked Open Data (LOD).[1]

In the last decade, the publicly available Knowledge Graphs (KGs) evolved as one of the rich sources of structured data adhering to 5-star data principles (cf. figure 2.2) in the form of Resource Description Format (RDF) [1, 2]. KGs provide an avenue to structure knowledge in a simple relation-based construct where the focus is placed on entities and their interlinking hinged upon their relations to one another. These KGs have grown to become a mainstay of the research in various communities including Databases [3, 4], Information Retrieval [5–8], Natural Language Processing [9, 10], and the semantic web [11–13]. This is attributed to the fact that KGs are (i) schematically represented, (ii) capacious sources of facts, (iii) adequately structured (graph-based), (iv) constantly growing/updated, and (v) publicly available on the Web [11]. Public KGs such as DBpedia [1], YAGO [14, 15], Freebase [16], and Wikidata [17] have been applied to a broad range of tasks including: Question Answering [18], KG Completion [19].

At a conceptual level, systems that interact with KGs perform some form of Natural Language Understanding (NLU) that entails a prepossessing step followed by identification and disambiguation of named entities and their relationships. For instance in a Question Answering (QA) system [18], the natural language question is first transformed into formal queries (here expressed as SPARQL[2]). However, the entities and relations must be mapped to their Universal Resource Identifiers (URIs) in the KG. Thus for the question *"What is the capital of Australia?"* the SPARQL query translation maps to: `SELECT DISTINCT?uri WHERE {dbr:Australia dbo:capital?uri.}`. In this query `dbr:Australia`[3] is the linked uri for the entity *"Australia"* while `dbo:capital`[4] is the linked uri

[1] https://www.w3.org/DesignIssues/LinkedData.html
[2] https://www.w3.org/TR/rdf-sparql-query/
[3] http://dbpedia.org/resource/Australia
[4] http://dbpedia.org/ontology/capital

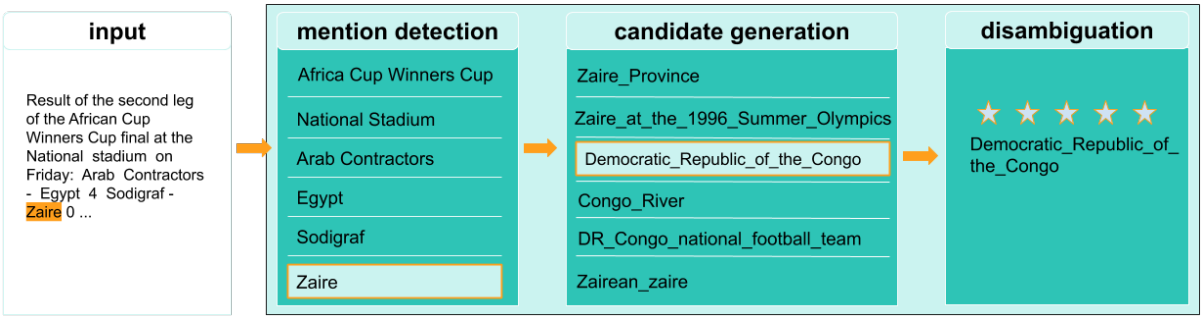


Figure 1.1: Stages in an entity linking process. The linking process can be performed in a pipeline or by using a monolithic end-to-end model.

for the relation *"capital of"* from the DBpedia KG. Figure 1.1 shows the stages involved in performing a linking task. Similar to the entities in the question, the entity *"Zaire"* is linked to the Wikidata item `wd:Q974`: *"Democratic Republic of the Congo"* [5]. With such a wide variety of applications that directly or indirectly depend on KGs, there is a need for approaches that identify mentions of entities, concepts, and relations in text and link them to the ground truth counterparts in the knowledge graphs, as seen in these examples. The two tasks that have been defined in the research community to bridge this gap are: i) Entity Linking or Named Entity Disambiguation (NED), and ii) Relation Extraction or Relation Linking [20–22]. From hereon, we will use the two forms: Entity Linking (EL) and Relation Linking (RL).

Research Objectives. Entity and relation liking has been a long-standing research domain. Several approaches have been developed for entity and relation linking ranging from rule-based systems [23–27] to approaches that rely on deep neural networks [28, 29]. In an attempt to bridge the task challenges (cf.,Figure 1.2), we observed two common trends in these systems. First, numerous approaches progressively utilise powerful algorithmic engineering to enhance performance encouraged by the work on neural networks [30–32]. It comes at a high cost of computation and the need for specialised hardware. Secondly, contextual information is often only obtained from the source sentence (local context) or discounted based on all sentences in the document (Global context) or sources such as WordNet [28, 33]. In the latter case, due to lack of rich context quality, adding more context introduces noise (i.e., too much irrelevant information), which translates to the need for more complex approaches to sift out the noise. Recent empirical evidence suggests that researchers have begun to appreciate the role that additional context can play in improving the performance of these tasks [28, 34, 35]. However, it is still not empirically studied which form of the background context positively impacts the extraction quality? For example, researchers in [28] induced entity descriptions derived from Wikipedia as additional knowledge to improve entity linking. The KGs, such as Wikidata, provide concise semantic descriptions of the entities. Is it possible that semantically detailed entity descriptions embody better signals for the underlying deep learning models to improve the entity linking? In this thesis, we hypothesise that KGs can also be used as background knowledge sources or *"knowledge context"*. This is because KGs already contain rich semantic information for entities and relations. The intuition is to leverage the power offered by these algorithms to adequately represent relevant information using knowledge context.

[5] https://www.wikidata.org/wiki/Q974

1.1 Motivation, Problem Statement, and Challenges

Entity Linking (EL) disambiguates textual mentions of entities to their corresponding entities in a reference knowledge base (e.g., Wikipedia[6]) or the knowledge graphs (e.g Wikidata[7]). On the other hand, Relation Linking identifies the knowledge graph relation between two entities in a piece of text. Figure 1.1 illustrates the steps involved in any linking task. There are two variations in the literature of approaches that perform linking. The end to end linking [36–39] undertakes all the stages where the mentions are unknown and must first be recognised before the candidate generation and ultimately followed by disambiguation. However, in disambiguation only approaches [9, 29, 34], the mentions (also referred to us surface forms) are assumed to have been identified, and the task concentrates on selecting the correct item within the KG that matches these mentions. The necessity of this research study emerged from an observation we have made on current state-of-the-art approaches, their limitations, and the open challenges in these tasks. Figure 1.2 show four examples of natural language representations of entities and relations and their corresponding representation in the two open-domain knowledge graphs (DBpedia and Wikidata). Additionally, some of the challenges in this study and the opportunities available to overcome these challenges are also indicated. In the first example, the question *"When did Guglielmo Marconi and his wife get married?"* has one entity mention *"Guglielmo Marconi"* and one relation mention *"wife / get married"*. in the DBpedia KG, this relation links to the property `dbo:spouse`. The challenges exposed in this question include the semantic gap between the representations *"married/wife"* vs. *"spouse"*. Likewise, in the KG, the other relations `dbo:marriage`,`wdt:marriage` have direct syntactic and semantic connection with the mention but are not the correct disambiguation. Researchers in different domains such as Question Answering [40, 41], and relation extraction [42] have grappled with these challenges. We identify an opportunity to leverage the term graphs that provide terms with their relationships as an avenue to solve this challenge.

The second example in figure 1.2 depicts the multi relation question: *"How many people live in the capital city of Australia?"*. The relation mention *"people live in"* links to the DBpedia relation `dbo:PopulationTotal`. However, in superficial consideration, this surface form can easily link to the class: `dbo:Population`. We observe an implicit relation brought about by the question desire. The question demands a numerical answer suggested by the words *"How many"* that implicitly translates the relation from `dbo:Population` to `dbo:PopulationTotal`. The other challenge here is the semantic gap challenge that we have already discussed since *"people live in"*, and *"Population Total"* have no direct semantic link. The opportunity we identify in research for bridging this relation is the existence of structural semantics entailed in DBpedia e.g. the connection of *"Australia"* to *"Canberra"* through the `dbo:capital` relation and the connection of the literal number *"381488"* as the population of *"Canberra"* via the `PopulationTotal`. A number of these structural semantics can assist the linking process, e.g. the *"instance of"* and the *"same as"* relations [24]. Our next example *"Soccer: Late Goals Give Japan win Over Syria."* depicts a statement about the *"Japan National Football Team"* `wd:Q170566` however the entity mention *"Japan"* has a direct link to the entity *"Japan"* - `wd:Q7` refering to the country. In this example, the correct entity subsumes the wrong entity. It is challenging to link such complex entities since the simple match exists on a legitimate entity. This particular example needs more than just the knowledge graph context, since we can obtain an equal match for the two entities from the KG. Indeed the shorter entity scores higher because it has both syntactic and semantic match to the mention. However, suppose we obtain an appropriate algorithm to match the overall sentence context that contains words like *"Soccer, Goals"* to specific portions of the KG context, e.g.

[6] https://www.wikipedia.org/
[7] https://www.wikidata.org/

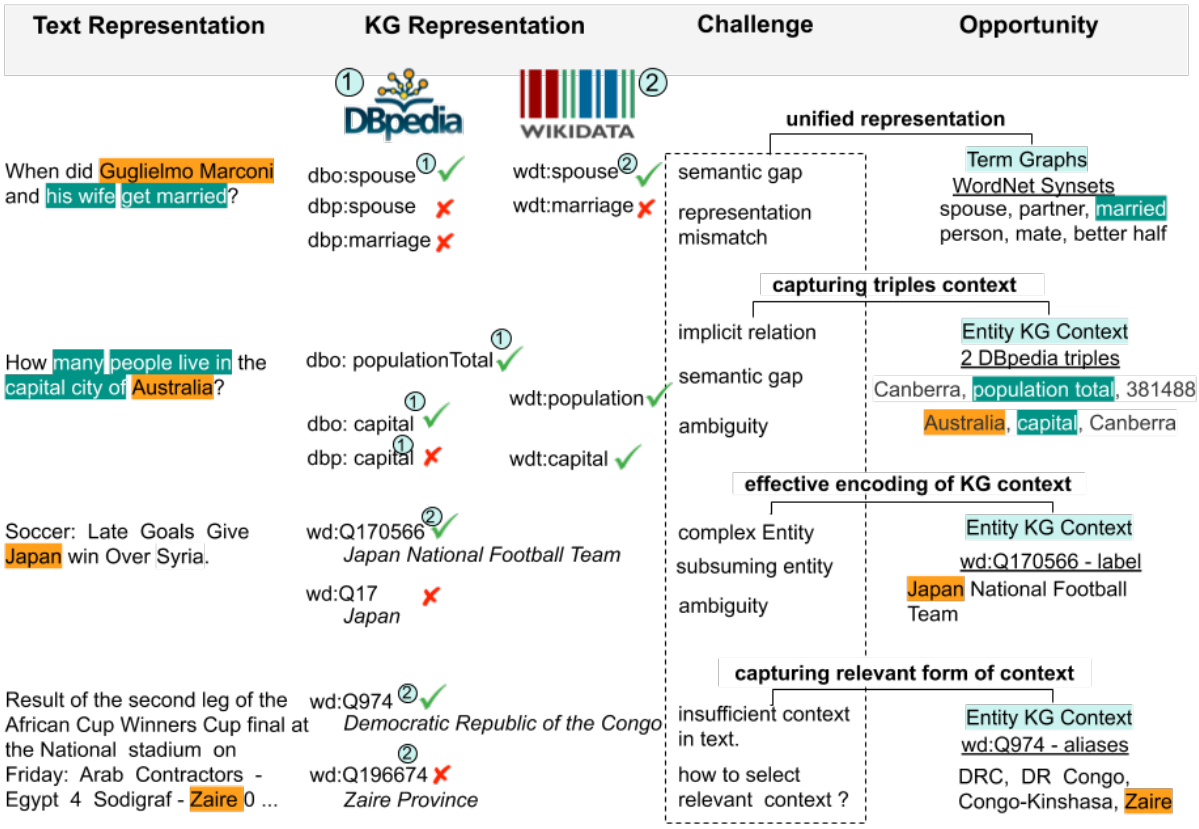


Figure 1.2: Four natural language sentences with different relations and entities that exhibit different challenges and the opportunities present to assist in bridging these challenges through context from KGs

"Football Team". In that case, we can separate the two entities. Finally, we take a look at the statement: *"Result of the second leg of the African Cup Winners Cup final at the National Stadium on Friday: Arab Contractors - Egypt 4 Sodigraf Zaire0, halftime 2:0 Scorers: Aly Ashour 7', 56'(penalty), Mohamed Ouda 24' 73'.Contractors won 4-0 on aggregate"* with the entity mention *"Zaire"*. This entity should link to `wd:Q974` - *"Democratic Republic of the Congo"*. We observe that the sentence does not contain any information to assist in disambiguating this entity. However, the KG contains other attributes that can be employed to assist the process. The challenge is how to perform filtering of the information to achieve correct matching.

Research Problem Definition

How can knowledge context be leveraged to improve performance of entity and relation linking?

1.1.1 Challenges for Linking Entities and Relations to Knowledge Bases

There are four distinguishable research challenges that we tackle in this thesis. With evidence from our motivating example in the previous section, we deduce the following challenges. Each challenge correspond to a sub research question except for challenge 3 and challenge 4 which are closely related and are observed as a single research question in this thesis.

Challenge 1: Dissimilar Representation Between KG and Natural Language Relations

Due to richness and evolutionary behaviour of natural languages, relations expressed in text exhibit characteristics that are not directly compatible with knowledge graph relations which are more structured and precise. Natural language relations emanate from open and infinitely growing vocabulary. Hence unpredictable sentential structured symbols are expressed sequentially and possess elaborate grammatical structure causing a vocabulary mismatch problem [43, 44]. On the other hand, the knowledge graphs are structured using standardised formal representation languages such as semantic web tools (RDF[8], OWL[9] etc). Due to specificity, items in the KG are expressed using a very limited vocabulary. However, the graph structure in the knowledge graph offers expressivity in the form of reasoning and inference. From this difference in representation, three challenges arise. i) **Semantic gap**: Take for example the relation `wd:p26` in Wikidata which is same as the DBpedia relation `dbo:spouse`. In the KG, this relation has the natural language label *"spouse"*, but in text, it is commonly represented by several forms: *"wedded, married to, marry, wife, husband, spouse e.t.c."* as shown in the first example of figure 1.2. This leads to the well-known challenge of "semantic gap"; in which a single meaning is represented in different forms. ii) **Implicit relations**: sometimes there may exist no direct mention of the relation within text (Commonly referred to as implicit relations [45]). E.g. in the statement *"Bocelli also took part in the Christmas in Washington special on Dec 12, in the presence of president Barack Obama and the first lady"*, the mentions *"Barack Obama"* and *"first lady"* refer to two entities in real world i.e. `wd:Q76` and `wd:Q13133` respectively within the Wikidata KG. There is an implicit relation between these two entities in the text because there is no mention. iii) **Ambiguity**: where the word or phrase representing the entity or relation has more than one possible interpretations. Ambiguity is also brought about when two items in the KG refer to the same mention in text. For example, `dbo:spouse`[10], and `dbp:spouse`[11] are two relations in different named graphs of DBpedia but semantically have the same meaning. Likewise, the word *"Apple"* in a sentence could refer to several items including the technology company *"Apple Inc."* `wd:Q312`[12], Apple the fruit `wd:Q89`[13] or the UK international record label; *"imprint of Apple Corps Ltd."* `wd:Q213710`[14], the *"1990 album by Mother Love Bone"* `wd:Q1754545`[15].

Opportunities: We observe that to overcome these challenges. There is a need to provide semantic augmentation. The term graphs such as Wordnet [46], ConceptNet [47], BabelNet [48], and linguistic thesaurus provide word relationships that can help bridge semantic gap and vocabulary mismatch. For instance, the question answering system AskNow [40] tries to overcome these challenges using the PATTY [49] relation patterns knowledge base. There is therefore, an opportunity to bridge these

[8] https://www.w3.org/RDF/
[9] https://www.w3.org/OWL/
[10] Linked to the DBpedia ontology named graph: http://dbpedia.org/ontology/spouse
[11] Linked to the DBpedia properties named graph: http://dbpedia.org/property/spouse
[12] https://www.wikidata.org/wiki/Q312
[13] https://www.wikidata.org/wiki/Q89
[14] https://www.wikidata.org/wiki/Q213710
[15] https://www.wikidata.org/wiki/Q1754545

challenges by employing Term graphs such as Wordnet [46]. Beaumont et al. [41] utilised this avenue and gave proof of the possibility of leveraging term graphs as an augmentation to bridge the representation gap.

Challenge 2: Difficulty of Encoding Knowledge Context for Deep Learning Models

For a long time, the need to encode inputs for machine learning has been major subject of research. Through these efforts numerical data are easily induced into statistical models. However, textual (and categorical) data is much more complicated to capture. Recently, researchers defined the word embedding [50, 51] techniques for obtaining vector representation of words and opened a plethora of avenues to represent textual data. This family of new approaches include the attention mechanism and transformer based language modelling [32, 52, 53]. These deep learning for NLP techniques offer the ability to learn distribution representation of words from text. Therefore, they have seen implausible applicability in the last half-decade. In end-to-end entity linking, the models have mostly been used as black boxes. In such cases, it becomes difficult to influence the behaviour of a model using external signals. This thesis seeks to define approaches to encode contextual information from knowledge graphs for use with neural networks. To achieve this goal, we need to leverage deep learning techniques for encoding context, while breaking the black box view of models to allow extra signals in the learning process. Consider the monolithic end-to-end entity linking approaches [37, 54]. The approach takes as input a piece of text and performs both the detection and disambiguation of named entities, in a combined training model. The challenge is then how to capture external knowledge context to boost the performance.

Opportunities: To tackle this challenge, we consider the existence of encoder/decoder neural network architectures and the ability to retrieve and verbalise KG information. The first task is to employ semantic web technologies for querying data from the KGs and organise this into a sequential input format. Since the attentive neural networks models [55] are designed to filter out relevant information from long input sequences, we see an opportunity to obtain an appropriate representation of the input context.

Challenge 3: Inadequate Contextual Information in Text for Disambiguation.

Any linking task heavily relies on sufficient information concerning the entity or relation to be processed. At the core of this process is the similarity scoring function between the textual features and the KG entity features. traditional linking approaches depend on the sentence to provide such features. However, in some instances, the textual context does not contain relevant information to assist in the linking process. Take for example the sentence: *"Result of the second leg of the African Cup Winners Cup final at the National Stadium on Friday: Arab Contractors - Egypt 4 Sodigraf Zaire 0, halftime 2:0 Scorers: Aly Ashour 7', 56'(penalty), Mohamed Ouda 24' 73'.Contractors won 4-0 on aggregate"* where the mention *"Zaire"* links to the Wikidata entity: `wd:Q974` with the title *"Democratic Republic of the Congo"*. The sentence contains very little that can semantically relate the mention to the entity. To tackle this problem, several researchers have sought to consider other sentences occurring in the same document [28, 56, 57] to obtain better contextual representation. Nonetheless, it is not always true that these other sentences contain more useful information. Moreover, for the true cases, it introduces extra irrelevant information that needs to be filtered out.

Opportunities: Considering the running example, the referent entity `wd:Q974` has several attributes that relate to the local (sentence) context, most striking of which is the aliases that contain the form: *"Zaire"*. With the aliases, we have a direct semantic relationship hence enabling the disambiguation process. We observe that such readily available information concerning entities and relations in the KG offer a tremendous opportunity to solve this challenge.

Challenge 4: Selecting the Relevant Knowledge Context for Disambiguation

In challenge 2, we see the gap in how to encode KG context for neural networks, while in challenge 3, we see the need for augmenting insufficient context with KG context. However, the structure of KG and the amount of information available about entities and relations in the KG is a significant factor when we need to obtain quality information for linking. Researchers have already attempted to incorporate KG context to improve entity and relation linking. For instance RESIDE [35] uses KG descriptions for relation linking, and employ a graph convolution network for correctly classifying the relation.

1.1.2 Overall Thesis Approach

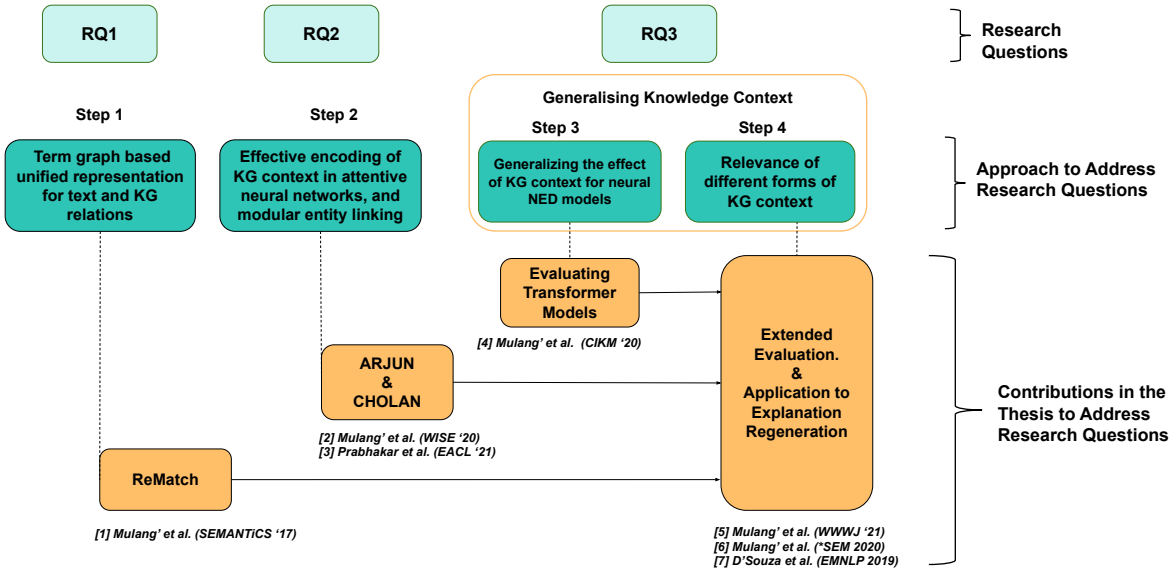


Figure 1.3: Approach for addressing the main research problem comprises four steps. Each Step addresses individual challenges of the overall approach, and is supported by research publications.

To answer the research questions, we define a multi-stage approach that addresses each of the four identified challenges as depicted in Figure 1.3. In the first step, we attempt end-to-end relation linking carried out under the short text scenario. Existing relation extraction approaches do not apply to the situation with unknown entities. In short text scenarios such as Question Answering, where the named entities are unknown, the task becomes extra challenging. The second challenge in this approach is the scarcity of training data. To overcome this, we attempt a heuristic approach that aims to coalesce the representation of text and KG to allow a similarity matching function. In this step, we attempt to alleviate the challenges faced in this task over the last decade and broaden the solution reach by targeting end-to-end RL. The solution we provide is the first of its kind, and attempts end-to-end relation linking and tackles *Challenge 2*.

The second step emerges from an observation we made in the first step, that entity linking is an integral aspect of relation linking. As such, we design two approaches in the second step to tackle the entity linking problem. However, we follow a novel path that seeks to integrate KG context by perturbing the linking process to allow the intermediate addition of contextual information about entities. Secondly, we leverage the power of neural networks to obtain appropriate representations (encoding) of the KG context for our models. This encoding of context and the methodology to incorporate the same into

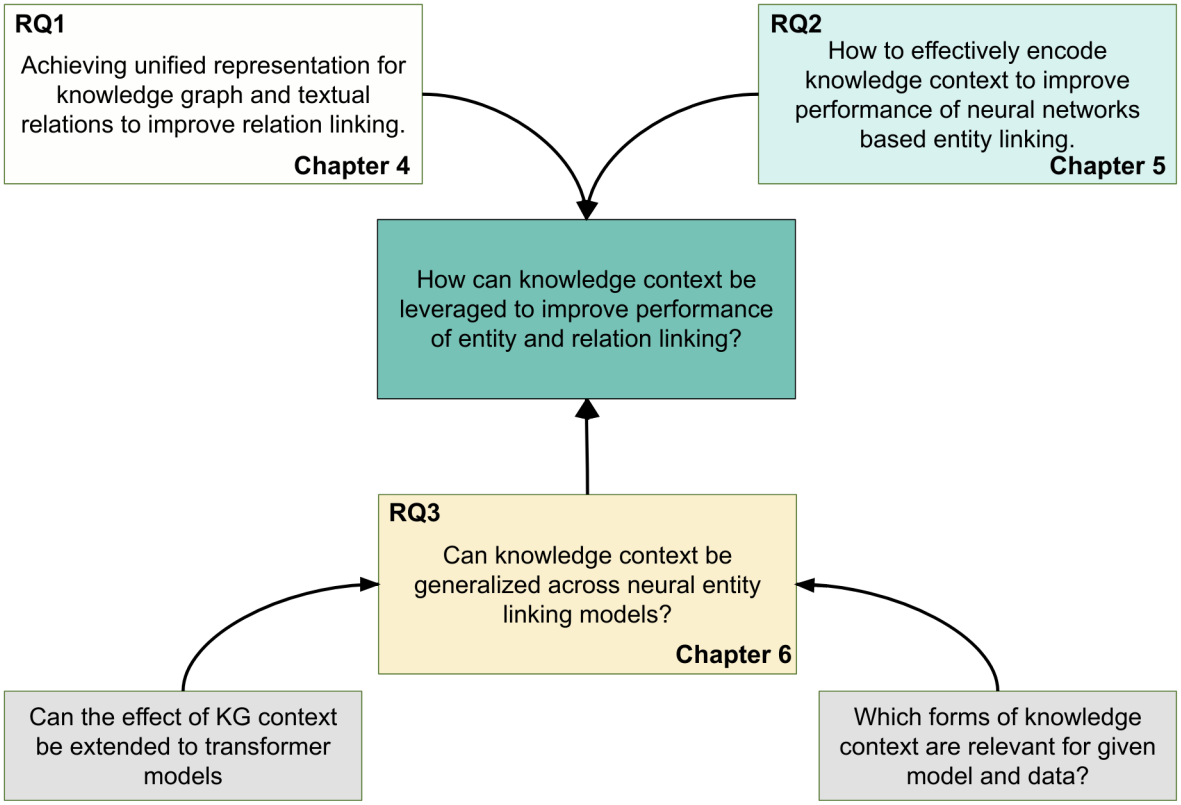


Figure 1.4: Three sub research questions contribute to the overall research objective of the thesis

neural network models constitutes the second stage (and *Challenge 2*) of the proposed approach. In the third step, we present a model agnostic view of KG context. In the evaluation, we incorporate extensive KG context into several state-of-the-art models to understanding their performance behaviour with and without such contextual information. This evidence-based exposition on the generalisability of KG context addresses *Challenge 3*. To address *Challenge 4*, we tackle the gaps left by research challenges *Challenge 2* and *Challenge 3* that is specifically brought about by the structure and volume of KG context. Since there is important inference obtainable from the ontological and triple layout of an entity in the KG besides the attribute information stored about an entity.

1.2 Research Questions

Based on the revealed challenges, we devise the following research questions to be addressed in the thesis. Each challenge is mapped to one sub-research question and collectively contributes towards the overall research question as illustrated in figure 1.4.

Research Question 1 (RQ1)

How can we achieve a unified representation of both knowledge graph and textual relations to enhance similarity matching?

Although work on relation linking has been done since the 1990s, initial approaches focused on binary relations. Later on, as datasets emerged for the task, and the surge in deep learning, several methods have been proposed. However the datasets on this tasks namely: New York Times dataset (NYT) annotated on Freebase KG [58, 59], TACRED [60], and recently the Sorokin dataset [61] aligned with Wikidata KG, assume that the entities are already annotated and linked, hence only the relation is to be disambiguated. This assumption is not always the case in real-world applications, such as Question Answering or dialogue scenarios. Therefore, it leaves a gap to find approaches that can perform end to end relation linking without knowledge of entities. When attempting a Zero-Shot relation linking with Question Answering datasets like QALD [62–64], LC-QUAD [65], and Simple Questions [66], a compulsive challenge arises concerning lack of labelled data for supervised machine learning. Under these two challenges, we examine how to bridge this gap by attaining a unified representation of textual relation and the KG relations to enhance the linking process. Thus forms our research first research question (RQ1)

Research Question 2 (RQ2)

How can KG context be effectively encoded in neural network architectures to improve Entity Disambiguation?

Current entity and relation linking approaches employ powerful deep learning techniques such as the attention mechanism of Neural Networks [28, 29, 37, 61], Transformers-based Language Models [32, 34, 52, 56], and Graph Neural Networks [22, 67]. These models are engineered to capture maximum insight from the local and global context in text. For this reason, a number of these models utilise a predefined set of candidates that allow for a monolithic end to end approach where the model acts as a black-box that accepts inputs and provides output. In a black-box approach, it becomes challenging to influence model performance using external signals. Moreover, the knowledge and structure of the KG do not readily lend themselves as direct input to these models since all such models demand a numerical representation in the form of vector space representation. Therefore, it is paramount that we define approaches that capture KG context into vector representation for use in state-of-the-art neural models. There is numerous research effort in knowledge graph completion [19, 68] that attempt to represent entities as vector representation about their position in the KG. These are, however, merely early efforts and have not matured enough to obtain commendable performance. Such KG embedding algorithms also provide a more global representation of entities and relations that may not be specific to pick the most vital aspects. Therefore, the challenge is how to obtain a fitting representation of KG context that captures as much relevant information from the KG as needed for a specific task.

Research Question 3 (RQ3)

Can the effect of knowledge context be generalised for neural entity linking models?

The recent rise of pre-trained transformers based language models (LMs) [32, 52, 53] can be seen as a huge breakthrough in NLP research. Pre-trained transformers possess two strong capabilities namely: i) Ability to capture distributed semantic representations over large amounts of text, and ii)the ease of transfer learning through fine-tuning [32]. Consequently, they have enabled many applications, including machine reading comprehension [69], and Question Answering [32, 52]. These models are intricate and trained on large datasets such as the whole Wikipedia and news articles, and hence expected to be highly contextualised. We set out to evaluate the counter-question: whether the deep contextual signals in these models are sufficient to assist in task-specific scenarios or whether extra knowledge context may improve performance. We explore this research question in two directions as indicated in figure 1.4.

- In the first phase of evaluation, we also juxtapose the performance of less powerful models such as the LSTM under KG context to obtain a robust and expansive view of the effect of this contextual information. To fairly perform this evaluation, we rely on open domain Entity disambiguation datasets owing to their proximity to the original data used for training. To further elaborate the generalisability of context enabled models, we choose datasets aligned to two different knowledge bases. The first concerns the general text-based knowledge base Wikipedia with datasets like CoNLL-Wikipedaia, AQUAINT [70, 71], MSNBC, Wikipedia datset, and ACE-2004). The second group of datasets aligns with the more structured knowledge graph Wikidata and includes the T-REX dataset [72], and the Wikidata-Disamb [73]. This research question aims to establish the impact of knowledge context under different datasets and models.

- In the next phase, we ask the question: are different forms of knowledge context relevant for neural entity linking models? This sub research question is directly related to Challenge 4. We set out to identify the relevance of different forms of KG context in given scenarios. The KG has several aspects that can be leveraged to provide concrete signals for linking of items. Entities possess attributes such as labels, aliases, descriptions, and instance of. These attributes are fostered by the ontological structure from the KG schema; additionally, the entity to entity triple relations constitutes a relatively large volume of information. especially for more common entities such as the entity `wd:Q76` *"Barack Obama"* with 1202 unique 1-hop relation (out-degree). we hypothesise that only specific forms of this KG contextual information are relevant for disambiguation on different tasks and datasets.

1.3 Thesis Overview

To present a high-level but descriptive overview of the achieved results during the conducted research, this section will highlight the main contributions of the thesis. We provide references to scientific articles covering these contributions published throughout the whole term.

1.3.1 Contributions

Contributions for RQ1

Leveraging term graph to achieve a unified representation of text and KG relations for relation linking.

Term graphs (e.g. ConceptNet [47], Babelnet [48], Wordnet [46] etc) are special types of graph networks that represent the relationships between terms in languages as per the well defined language vocabularies. Such relations include synonyms, hypernyms, hyponyms, and meronyms etc. such relations can assist by enriching data to assist in reasoning or provide features for training AI algorithms. To address the first research question, we present the **ReMatch** approach that incorporates the term graph *"WordNet"*[16] to augment relation words in a questions. In a similar construct, we enrich the candidate KG relations with elements from the term graphs and structural information from the KG. This similarity in representation achieved is then used in the next step to aid in heuristic similarly matching. Our overall plan in this solution is to target short text communication, such as questions, tweets, and brief conversational texts. To evaluate our work, we operate within the question answering task where the main concern is to translate natural language patterns into formal queries to be used in retrieving the answers from KG. This unique challenge in this research area is to identify which property within a Knowledge Graph matches the predicate found in a Natural Language (NL) relation. However, formal query generation approaches attempt to resolve this problem by first retrieving the named entity from the KG together with a list of its predicates. The next step filters out one from all the entity's predicates. **ReMatch** endeavours to directly link the natural language (NL) predicates to KG properties in *(zero-shot setting)* for use in QA pipelines. In our contribution, we provide a systematic approach and implement a tool that can directly be employed to solve the relation linking task in different pipelines. We model KB relations with their underlying parts of speech and *dependency parsing* characteristics before adding the previously discussed Wordnet augmentation. From a question, we model a similar representation of query relations. Ultimately, we define distance measurements between the query relation and the properties representations from the KG. This approach is a first of it's kind to attempt zero-shot relation linking in the challenging scenarios with short text such as a question.

Contributions for RQ2

Effective encoding of KG context for attentive neural network

In the entity linking task, the underlying KG is generally utilised as the source of target entities. However, these KGs often contain other relevant information, such as aliases of entities (e.g., *"President Obama"* and *"Barack Hussein Obama"* are aliases for the entity *wd:Q76 "Barack Obama"* that has the KG description: *"44th president of the United States"*). Historically, EL models tend to ignore such readily available entity attributes. In our first contribution- **Arjun** towards addressing this research question, **(RQ 2)**, we examine the role of knowledge graph context on an attentive neural network approach for entity linking. The overall research plan is to utilise such information to capture nuances that are otherwise not available in text. To evaluate our work, we tackle a relatively unexplored KG:

[16] https://wordnet.princeton.edu/

(the *"Wikidata"* KG). Wikidata is a collaborative knowledge graph that excessively relies on the crowd to author the content. Given that the crowd does not adhere to standard protocols for assigning entity titles, the KG is populated by nonstandard, noisy, and long titles. This gives rise to challenges that impact the precision and recall of several Entity Linking (EL) approaches long, implicit, and nonstandard entity representations. **Arjun** contributes by exploiting the sufficient context from a KG as a source of background knowledge, which is then fed into the neural network. This approach demonstrates merit to address challenges associated with entity titles (multi-word, long, implicit, case-sensitive).

Our second contribution in this direction is an improved modular pipeline to experiment with the recent bidirectional transformer models, which targets end-to-end entity linking (EL) over knowledge bases. The architectural formulation here is geared towards breaking the tasks of EL to allow a flexible candidate generation. Models can then include more contextual knowledge during the interchange between mention detection and disambiguation stages. Therefore, the pipeline leverages two transformer-based models [32] integrated sequentially to accomplish the EL task. The first transformer model recognises the entity mentions (surface forms) in the given text. A second transformer model is employed to classify the target entity among a predefined candidates list for each mention. The latter transformer is enhanced by an enriched context captured from the sentence (i.e. local context), and *entity description* retrieved from Wikipedia. We conduct our empirical study on two well-known knowledge bases (i.e., Wikidata and Wikipedia). The empirical results suggest that we outperform state-of-the-art approaches on standard datasets such as CoNLL-AIDA, MSNBC, AQUAINT, ACE2004, and T-REx.

Contributions for RQ3

Generalising the effect of KG context for Named Entity Disambiguation models

After our work in the RQ2, we determined that KG context provides quality signals for deep learning models. Likewise, we determined that the encoding ability of the encoder-decoder architecture of neural networks provides a powerful avenue for representing the knowledge context. To address the third research question, we take two closelz related paths as described below:

Evaluate the generalisability of context on different models:. Pretrained Transformer models [31, 32, 52, 53] have emerged as state-of-the-art approaches that learn contextual information from text to improve the performance of several NLP tasks. These models are trained with huge volumes of data from the internet and news articles over long periods using highly specialised hardware. The community unanimously agrees that the representations expressed in pre-trained transformers are essential in several NLP tasks. Notwithstanding, we identified the need to empirically study the behaviour of these models in comparison to other simpler models under KG context. In this work, we postulate that context derived from KGs provide valuable features to inform pre-trained transformer models and improve performance for the named entity disambiguation (NED) task. We further seek to standardise our approach to the more general knowledge base of Wikipedia. We evaluate the impact of KG context on state-of-the-art NED model for the Wikipedia knowledge base.

To analyse the relevance of different forms of KG information and Applications: The Named Entity Disambiguation (NED) task assigns referent entities from a knowledge base to entity mentions appearing in text. The recent proliferation of Knowledge Graphs (KGs) as special forms of structured knowledge has offered a new opportunity for research and applications to access better targeted and quality information. Research has shown that KGs can be used as rich sources of contextual features to improve the performance of several downstream natural language processing (NLP) tasks. For the

NED task, several forms of entity-specific information from KGs such as entity type, entity labels, description and aliases, and triples have been utilised to improve system performances. However, existing approaches have thus far arbitrarily chosen the kind of this contextual data, without empirical attestation as to which information suits which type of scenarios. Therefore, in this work, we investigate the role of KG context on Named Entity Disambiguation (NED) models. Notably, we experiment with different forms of information from the KG to evaluate the behaviour of two state-of-the-art NED models when supplemented with such extra context. Our empirical evaluation using Wikidata KG indicates that different forms of entity-specific KG-context have varying influence on a model depending on the nature of underlying data.

In a last step presented in chapter 7, we desire to apply our findings on a different application domain. The multi-hop inference for explanation regeneration is an emerging research task in NLP. Our application dataset is obtained from the Textgraph-13 shared [74] workshop task at EMNLP 2019. The general target is to mimic students' reasoning as they select one answer from among several choices. To achieve this, there is a need to utilise commonsense knowledge bases and linguistic reasoning capabilities. We evaluate a Support Vector Machine (SVM), powered by extensive human-engineered features. We seek to collect relevant contextual information from online term graph ConceptNet [47, 75] and triple KB [76]. Moreover, we attempt a different approach that employs powerful transformer models [32]. We fine-tune this model by adding extra contextual features relating to the importance of words in the sentence. This novel feature termed as the Focus words [77, 78], reinforces the relevance of specific terms.

1.3.2 Publications

The following list of publications contributes a scientific basis of this thesis and acts as a reference point for numerous figures, tables and ideas presented in the later chapters. Please note that the co-authors in the papers are either Professors, post-docs, or masters students. For the papers co-authored with other PhD students, individual contribution is clearly mentioned. Therefore, parts of the contributions of this dissertation which are mentioned below, were achieved as the result of effective teamwork. The author (Isaiah Mulang' Onando) will use the "we" pronoun throughout this dissertation. Still, all of the contributions and materials presented in this work originated from the work of the author solely by himself.

- *Journal Papers (peer reviewed)*

  1. **I.O. Mulang'**, K. Singh, A. Nadgeri, S. Shekarpour, J. Hoffart, S. Auer. *Its just the silly context! Analyzing the Role of Wikidata context on Entity Disambiguation Models* (Under Review - World Wide Web Journal)

  2. J. D'Souza, **I.O. Mulang'**, S. Auer. *Ranking Facts for Explaining Answers to Elementary Science Questions*. (Under Review - Journal of Natural Language Engineering)

- *Conference Papers (peer reviewed)*

  3. **I.O. Mulang'**, K Singh, F Orlandi. *Matching Natural Language Relations to Knowledge Graph Properties for Question Answering.* In Proceedings of the Semantics, ACM, 2017.

  4. **I.O. Mulang'**, K. Singh, A. Vyas, S. Shekarpour, M.E. Vidal, Jens Lehmann, Sören Auer *Encoding Knowledge Graph Entity Aliases in Attentive Neural Network for Wikidata Entity Linking*. In proceedings of the Web Information Systems Engineering – WISE 2020. WISE 2020. Lecture Notes in Computer Science, vol 12342. Springer.

5. M. Prabhakar, K. Singh, **I.O. Mulang'**, S. Shekarpour, J. Hoffart, J. Lehmann. *CHOLAN: A Modular Approach for Neural Entity linking over Wikidata and Wikipedia*. In the Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics.
6. **I.O. Mulang'**, K. Singh, C. Prabhu, A. Nadgeri, J. Hoffart, J. Lehmann. *Evaluating the Impact of Knowledge Graph Context on Entity Disambiguation Models* In CIKM'20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020, pages 2157–2160, 2020. ACM.
7. **I.O. Mulang'**, J. D'Souza, S. Auer. *Fine-tuning BERT with Focus Words for Explanation Regeneration*. In Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics. StarSem, 2020.

- *Workshop Articles (peer reviewed)*

8. Jennifer D'Souza, **Isaiah Onando Mulang', Sören Auer**.*Team SVMrank: Leveraging Feature-rich Support Vector Machines for Ranking Explanations to Elementary Science Questions. TextGraph workshop, EMNLP 2019*

- *Miscellaneous Papers (peer reviewed)*

  Following publications originated during and are related to this thesis but are not part of the thesis itself.

9. K Singh, **I.O Mulang'**, Jaradeh, A Sakor, I Lytra, ME Vidal, C Lange, S Auer. *Capturing Knowledge in Semantically-typed Relational Patterns to Enhance Relation Linking.* In Proceedings of the Knowledge Capture Conference (K-Cap), 2017, ACM;
10. A. Sakor, **I.O. Mulang'**, K. Singh, S. Shekarpour, M.E. Vidal, J. Lehmann, S. Auer. *Old is Gold: Linguistic Driven Approach for Entity and Relation Linking of Short Text* - NAACL, 2019.

The full list of publications completed during the PhD term is available in Appendix A.

1.4 Thesis Structure

The thesis consists of night chapters structured according to specific thematic partitioning. In chapter 1 we introduce the thesis by discussing the motivation for the conducted study, the main research problem, research questions, and the scientific contributions that address research questions, together with a list of published scientific papers that formally report the contributions in this thesis. Chapter 2 presents underlying concepts and foundational background in the fields of Knowledge Graphs, Machine Learning, for a comprehensive summary of the research problem. An outline of state-of-the-art efforts in the entity and relation linking is detailed in Chapter 3. We describe approaches, tools, and early attempts at leveraging Knowledge Graphs as signals for statistical machine learning to provide detailed insights into the limitation and gaps we identified in this thesis. In Chapter 4, we lay out our first approach to relation linking. Deviating from the conventional Relation Extraction task that assumes entities have been identified and disambiguated, we approach relation linking under a Question Answering environment where Questions are analysed and answered from scratch. This first attempt to end-to-end relation linking employs a Term graph to semantically augment the identified mentions and the KG relations. This results in a unified representation that enhances a similarity matching for linking the relations. From

work in Chapter 4 we determine that it is important first to achieve proper named entity disambiguation to assist the relation linking process. Therefore in Chapter 5 we describe Two approaches to encode KG context in neural networks for end-to-end entity linking. The Arjun approach first builds a local infused KG that indexes entities with their aliases from the Wikidata KG. We then utilise a neural encoder-decoder architecture to encode this information into vectors for disambiguation. Subsequently, our extended architecture [79] employs a modular approach that allows to mix and match the candidate generation stage (see figure 1.1) and employs a transformer-based architecture for both the recognition and disambiguation modules.

These two approaches discussed in chapter 5 provide evidence that knowledge context improves the performance of entity linking, especially when encoded for deep learning models. However, we employed only a minimalistic portion of this information, hence to generalise the effect of KG context on these models, we seek to evaluate the performance of SOTA models when powered by KG context. Chapter 6 first describes our evaluation findings (in section 6.1), in which we define an input representation of KG context for Transformer architectures and report performance again models that have not been induced with such context. To further this research agenda, we extend our evaluation to understand the behaviour of our models under different forms of context and report our findings in section 6.2 of chapter 6. Chapter 7 describes an application of context enabled models to a specific problem domain. The explanation regeneration task is a relatively new problem in NLP and had a whole task proposed at the EMNLP 2019 Textgraph-13 workshop. We use the dataset released in this workshop and derive contextual information from term graphs to provide signals to machine learning models. Finally, Chapter 8 concludes the thesis with directions of future work. We revisit the research questions and answer them based on the results and findings described in the contribution chapters.

CHAPTER 2

Background

To address the problem of entity and relation linking by leveraging knowledge context from knowledge graphs as defined in chapter 1, a comprehensive approach is needed that draws insights from different viewpoints and perspectives. This chapter describes the underlying principles and concepts that act as the foundations for addressing the challenges. Figure 1.4 depicts the main constituents of the defined research problem. Section 2.1 describes the concepts related to Knowledge Graphs, which is the cornerstone of our work to provide knowledge context. In this section, we first describe the structuring of a KG using the Description Framework (RDF), followed by a formal description of Knowledge Graph and context as used in this thesis. This is relevant for all our research question (RQ1,RQ2, and RQ34) as the solutions rely on context and it's representation. Machine Learning approaches described in section 2.2 are utilised both as models to implement our approaches in RQ2 and RQ3 whereas the specific encoders in section 2.2.2 are foundational ideas for the models we employ to represent KG context in RQ2 and RQ3. Finally section 2.3 provides a summary.

2.1 Knowledge Graphs

Knowledge Graphs have been developed from several concepts in computer science and mathematics. This section discusses the founding principles that acted as stimulus to the emergence of knowledge graphs. The Semantic Web (section 2.1.1) encouraged the growth of KGs by availing a simple way to structure data in graph-like structures in which a single factual information is represented in simple forms called triples. Over a decade of research on the Semantic Web has availed several tools for design and processing information, which has facilitated research on design and use of KGs. Likewise, the mathematical principles of a graph and the operation a graph extends are preserved in a KG. as such, we briefly discuss graph principles in section 2.1.3.

2.1.1 Semantic Web and RDF

To address the problem of entity and relation linking by leveraging knowledge context from knowledge graphs. The worldwide web (WWW) has grown to be a repository of several types and forms of information including documents, images, videos and various files identified through unique Uniform Resource Locators (URIs)[1]. To access these items on the web, the Hypertext Transfer Protocol (HTTP)[2] [80] has become the standard protocol enabling communication and interchange. Although the web

[1] https://www.w3.org/Addressing/URL/url-spec.html
[2] https://www.w3.org/Protocols/

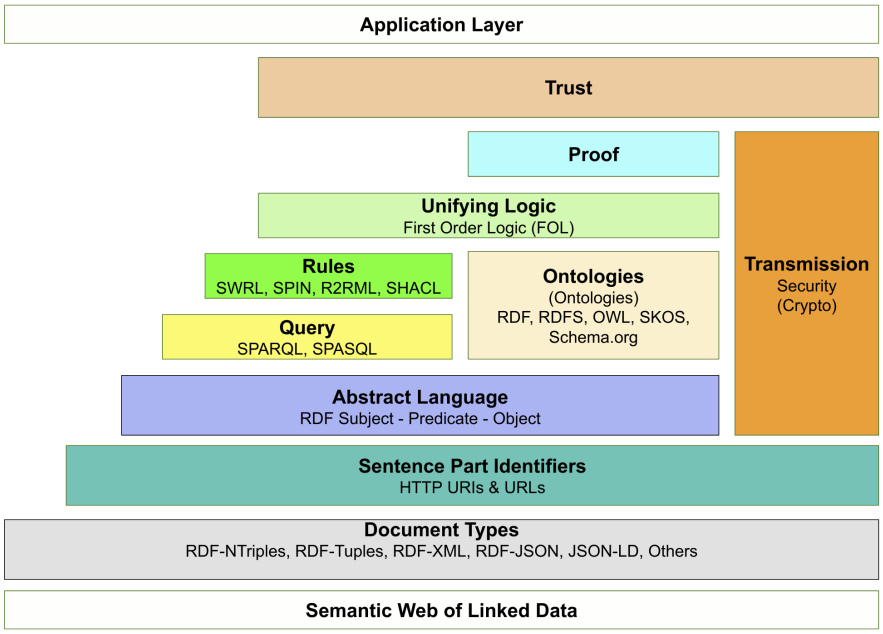


Figure 2.1: The Semantic Web Stack (Layer Cake): Depicts the major aspects of the semantic web.

was largely successful in enabling communication, transfer and storage of files, the world wide web fosters machine-to-human and human-to-machine interaction. Therefore, the vision to enable machines to comprehend semantic documents and data was proposed by Tim Berners-Lee [81]. The inspiration is to enable machine-to-machine communication, where machines can understand the context of data and software agents can automatically process information. The semantic web is an extension of the existing web in which data is made more structured and accessible by adding meaning to the information. The Semantic Web envisions three major attributes, namely: i) Build upon existing web with content existing in native WWW formats ii) Expressive semantic description that allows the ability for reasoning and proof. iii) Security and trust. Figure 2.1 shows the semantic web layers.

The document layer, there are several languages on the web that have evolved over the years. The Extensible Markup Language (XML) [82] is one of the significant breakthrough technologies for semantic description of data and objects on the web. XML focuses on simplicity, generality, and usability of textual data and represent information in a hierarchical format. Incrementally, several technologies have been built that anchor upon the XML including i) Extensible Style Language (XSL) which describes how the XML document should be displayed and includes a description of a transformation language (XSLT), ii) XLink: A language that allows elements to be inserted into XML documents so as to create and describe links between resources, and iii) XQuery: facilitates the data extraction from XML documents. In the quest to achieve more expressivity in data representation, other notable data formatting technologies have emerged.

The Comma Separated Values (CSV) stores textual content in a flat-file with fields separated using commas (or tabs for TSV). JavaScript Object Notation (JSON): is an open standard file format, and data interchange format, that employs human-readable text for storage and transmission of data objects in the form of attribute-value pairs and array data types (or serialisable value). The five-star deployment scheme illustrated in figure 2.2 was introduced by Tim Berner-Lee in 2010 [83] indicates the levels of data deployment scheme in which Linked Open Data (LOD) occupies the highest order. each level

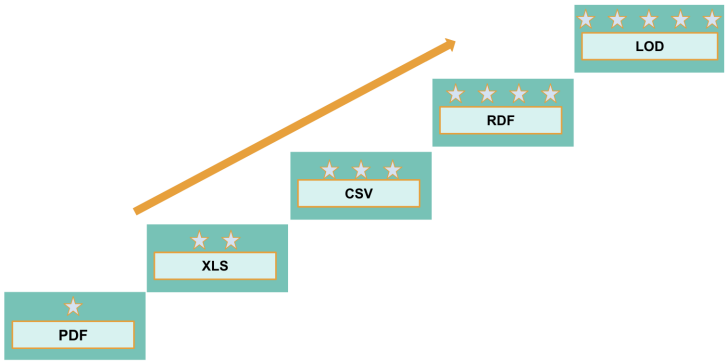


Figure 2.2: 5-Star Deployment Scheme. PDF, XSL, CSV, RDF, and Linked Open Data (LOD) represent five levels of data deployment schemes in increasing order of openness as proposed by Tim Berners Lee [83].

incrementally adds a machine readability feature to data e.g. the first level simply requires data to be published on the web with an open licence. The second level requires the first level but the data must be availed as machine-readable structured data (e.g. excel). At the third level, the first two levels are required but the data is published using non-proprietary format like the CSV. Level four follows requirements of the first 3 levels but requires use of open standards from W3C (RDF and SPARQL) to identify things. The last level (level 5) demands an extra interlinking of data to other data repositories. Furthermore, there are four linked data principles proposed [83] to promote reusability and add semantics:

- To use URIs as names of the things (i.e. resources);
- To use HTTP URIs for dereferencing such that user can look up for these names easily;
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL);
- Interlink URIs so that people can discover more things.

These principles promote openness and interlinking of information. Adopting these principles has led numerous data providers to easily publish information on the Linked Open Data (LOD) Cloud [84] that interconnects a large and constantly growing 5-star datasets. Following the same steps, more structured datasets that concentrate on entity and relationship representation have emerged. Examples include DBpedia [1, 85] that creates a structured form of Wikipedia from Wikipedia info-boxes; Wikidata [17, 86] authored by the crowd and Wikipedia [3]; and LinkedGeoData [87] geospatial data obtained from OpenStreetMap [4].

Resource Description Framework - RDF

At the core of the data representation layer in the semantic web stack of figure 2.1 is the *Resource Description Framework (RDF)*. RDF is the de facto data model for data interchange on the semantic web and is a W3C recommendation since 1998 [88]. In RDF specification, triples (*"<subject, predicate, object>"* or *"<entity, attribute, value>"*) are used to define the meaning of data and provide a formal resource description. These triples are referred to as *RDF Statements*: i.e. the statement made by a token of an RDF triple. The subject and object of an RDF statement are instances of `rdfs:Resource` while

[3] https://www.wikipedia.org/
[4] https://www.openstreetmap.org/

the predicate of RDF statement is an instance of `rdf:Property`. The RDF subjects and objects define the resources or the assets and can be either a URI or a blank node (a blank node is used to represent an individual with certain properties without a name). RDF object can also be literal data values. The predicate or property denotes the relationship between the subject and the object and is always a URI. A specific interesting type RDF statements is when RDF documents represent a resource (e.g. *"Uhuru Kenyatta - dbo:Uhuru_Kenyatta*[5]*"*) has properties (e.g. *"president of"*,*"nationality"*) with certain values (another resource *"dbr:Kenya*[6]*"*). RDF was initially used to describe metadata for web resources but has since been generalised to encode structured information.

RDF is limited in expressivity because it only possesses assertional knowledge. Consequently, it is possible to define irrational triples when using RDF, a drawback that makes RDF unsuitable for modelling schemas. RDF Schema (RDFS) is an extension of RDF with a special vocabulary for terminological knowledge (more expressive than assertional knowledge used with RDF). Therefore RDFS is used to define RDF vocabularies and is recommended by W3C in RDF family. It exerts constraints on the use of RDF by providing the power to define classes and properties [88]. Since RDFS is an extension of RDF, every RDFS graph is an RDF graph. RDF things or entities are instances of the *rdfs:Class*. Inside the schema, a property is defined with the classes whose instances can form the subject or object referred to as the *domain* and *range* respectively. A property's domain is defined using *rdfs:domain* while the range is defined using *rdfs:range* (c.f. the next section 2.1.3). RDF can be defined using several syntactical formulations referred to as *RDF serialisation* as shown in the second layer (syntax layer) in the semantic web layer cake (figure 2.1). Below is a list of the common RDF serialisation formats:

- `Turtle`[7]: a text format known for its human readability;
- `N-Triples`[8]: a text format focusing on simple parsing;
- `N-Quads`[9]: which is a superset of N-Triples for serialising multiple RDF graphs;
- `Notation 3`[10]: or N3 is a text format with advanced features beyond RDF;
- `RDF/XML`[11]: the official XML [**XML**] serialization of RDF;
- `JSON-LD`[12]: the official JSON [**JSON**] serialization of RDF;
- `RDFa`: a mechanism for embedding RDF in HTML.

2.1.2 SPARQL and Querying

To access information from semantic web-based datastores, the SPARQL [89] (recursively defined as: SPARQL Protocol and RDF Query Language)) is a query language designed specifically for use to retrieve data on the semantic web. The W3C released SPARQL as a recommendation in 2008 as a language able to queries unknown elements of a relationship from RDF data through *graph patterns*. It employs *Basic Graph Patterns (BGP)* to describe complex operation such as conjunctions and disjunctions. The

[5] http://dbpedia.org/page/Uhuru_Kenyatta
[6] http://dbpedia.org/page/Kenya
[7] https://www.w3.org/TR/turtle/
[8] https://www.w3.org/TR/n-triples/
[9] 5https://www.w3.org/TR/n-quads/
[10] 4http://www.w3.org/DesignIssues/Notation3.html
[11] https://www.w3.org/TR/rdf-syntax-grammar/
[12] https://www.w3.org/TR/json-ld/

results of a SPARQL query can either be result sets or RDF graphs.

Basic Graph Pattern [90]

Definition 2.1.1 (Basic Graph Pattern) *If $\mathcal{U}, \mathcal{B}, \mathcal{L}$ are infinite disjoint sets of URIs, blank nodes, and literals, respectively, and V is a set of variables such that $V \cap (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L}) = \emptyset$; A triple pattern tp is a member of the set $(\mathcal{U} \cup \mathcal{V}) \times (\mathcal{U} \cup \mathcal{V}) \times (\mathcal{U} \cup \mathcal{L} \cup \mathcal{V})$. Given the triple patterns $tp_1, tp_2, \ldots, tp_n$ be triple patterns, a Basic Graph Pattern BGP is given by is the conjunction of triple patterns, i.e., $BGP = tp_1 \wedge tp_2, \ldots, \wedge tp_n$.*

Special functions defined in SPARQL include: `FILTER`, `OPTIONAL`, and the logical operators (`UNION` and `AND`), aggregate functions in SPARQL. To retrieve information, SPARQL defines the `SELECT` query that returns all or a subset of variables bound in the query. Like wise the `ASK` query checks for a match of the query pattern with the given BGP and returns a truth value `TRUE` or `FALSE`.

2.1.3 Defining Knowledge Graph Context

Knowledge graphs are based on the mathematical field of graph theory. A knowledge graph is therefore a special type of a graph constructed to represent world facts. To define a KG, we begin by looking at the theoretical definition of a graph. For semantic consistency in this work, we represent the vertices and edges of a graph using the notation $\mathcal{E}$ and $\mathcal{R}$ to denote entities and relations, respectively. Therefore, a graph is defined as follows:

Graph - Graph Theory

Definition 2.1.2 (Graph) *A graph is an ordered pair $G = (\mathcal{E}, \mathcal{R})$, where $\mathcal{E}$ is a finite set called the set of vertices of G, and $\mathcal{R} \subseteq \binom{\mathcal{E}}{2}$ is a set of pairs of elements in $\mathcal{E}$ called the set of edges of G.*

- *The edge $r = \{v, e\} \in \mathcal{R}$ is also denoted by $r = ve$.*
- *If $r = ve \in \mathcal{R}$ is an edge of G, then v is adjacent to e and v is incident to r.*

Building on this definition, a knowledge graph (KG) is a special graph that incorporates unique characteristics brought about by the type of information represented. The need to follow a given data format but allow for expressivity and reasoning makes KGs very interesting structures. Whereas a simple graph has single-valued vertices and simple labelled or unlabelled edges, a KG's vertices are themselves sub-graphs that describe attributes of entities. Depending on the KG design, the edges may also be simple labelled edges or hyper-relational (where an edge has several attributes). Generally, a Knowledge graph is defined as follows:

Knowledge Graph

Definition 2.1.3 (Knowledge Graph) *Generally: A knowledge graph KG is a labeled directed multi-graph expressed as a quadruple* $KG = ((\mathcal{E}, \mathcal{R}), L, F, \mathcal{T})$ *such that:*

1. $\mathcal{E}$ *is the set of all the entities in KG that represent the vertices, referenced with unique identifiers:* $e \in \mathcal{E}$.
2. $\mathcal{R}$ *is the set of all relations in the KG that represent the edges between vertices, referenced by unique identifiers:* $r \in \mathcal{R}$.
3. L *represents the natural language labels of vertices and edges in the KG*
4. F *is a function* $F : (E \cup R) \rightarrow L$ *that returns the label of any item in the KG such that given* $u \in (\mathcal{E} \cup \mathcal{R})$; $l_u \in L = F(u)$.
5. $\mathcal{T}$ *is the ordered set of all triples in the KG represented as* $\mathcal{T} \subseteq (\mathcal{E} \times \mathcal{R} \times \mathcal{E})$. *Where a triple* $(h, r, t) \in \mathcal{T}$ *implies* $h, t \in \mathcal{E} \ \wedge \ r \in \mathcal{R} = ht$ *where h is the head entity and t is the tail entity.*

The open-domain knowledge graphs published on the web, such as DBpedia, Wikidata, YAGO, and Freebase, are designed using semantic web concepts and principles. These group of KGs is referred to as the RDF Knowledge Graphs. RDF KGs differ from other types of KG (e.g., triple stores or graph databases) because they utilise RDF as the core formatting language. Consequently, these KGs exhibit specific characteristics that are relevant for use in different NLP and inference tasks. The models in this thesis are evaluated using datasets build for these knowledge graphs. Hence the knowledge context deployed in these models is also obtained from the open-domain KGs. We define as an RDF KG as follows.

RDF Knowledge Graph - Holistic View

Definition 2.1.4 (RDF Knowledge Graph) *A RDF Knowledge Graph is a Knowledge Graph with additional set of vertices called concepts (classes) and special non-entity value vertices called literals. Formally: A RDF Knowledge Graph* $KG^{RDF} = ((\mathcal{C}, \mathcal{P}, \mathcal{E}, \mathcal{R}, \mathcal{L}), L, F, \mathcal{T}^{+})$ *Where:*

1. $\mathcal{C}$ *is the set of all concepts in the RDF schema.*
2. $\mathcal{P}$ *is the set of all properties between classes in the RDF schema.*
3. $\mathcal{L}$ *is the set of all literals in the KG*
4. $\mathcal{T}^{+}$ *is a set of all triples: subject,predicate,object* (s, p, o) *denoted as:* $(s, p, o) \in (\mathcal{T} \cup \mathcal{S} \cup \mathcal{I} \cup L)$
   - $\mathcal{S} \subseteq (\mathcal{C} \times \mathcal{P} \times \mathcal{C})$ *is the set of RDF Schema triples defining the concepts and their relationships.*
   - $\mathcal{I} \subseteq \bigcup_{e \in \mathcal{E}, c \in \mathcal{C}} (e, isa, c \mid e \prec c) \bigcup \bigcup_{r \in \mathcal{R}, p \in \mathcal{P})} (r, isa, p \mid r \prec p)$ *is the set of instance of (is-a relations between entities and relations to Concepts and properties in the schema).*

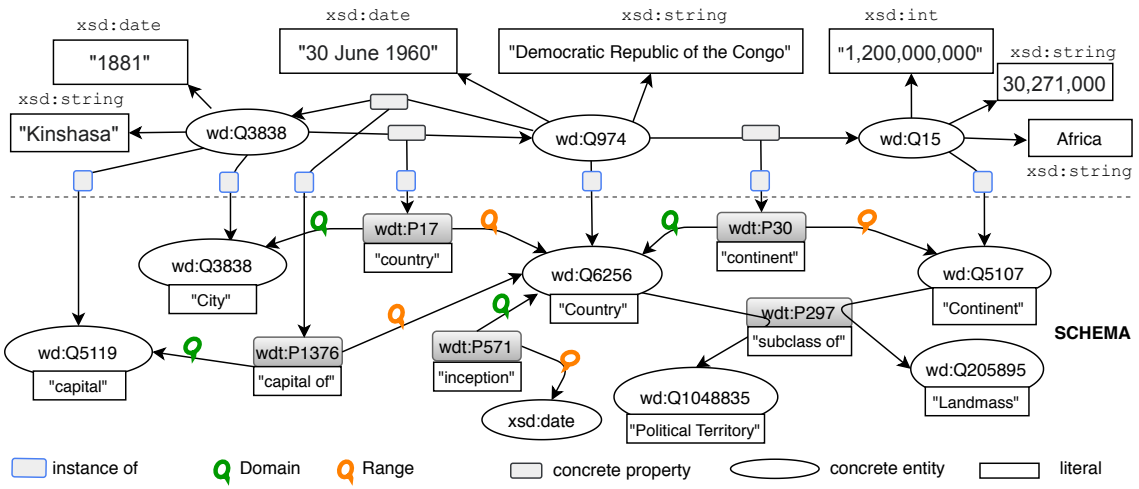


Figure 2.3: Illustration of a Subgraph of Wikidata Depicting the schema modelling and the instance entities of an RDF knowledge graph. The types of relations depend on which part of the KG the relation is defined (The Terminologies Box T-Box or the Assertions Box)

- $L \subset (\mathcal{E} \times \mathcal{P} \times \mathcal{L}) \cup (\mathcal{R} \times \mathcal{P} \times \mathcal{L})$ *is the set of all triples where the object is a literal. This set defines, in part, the attributes of an entity.*

After formally introducing the knowledge graph, we will spend the next part of this section to illustrate the richness of information contained in a KG. using the running example from figure 1.2: *"Result of the second leg of the African Cup Winners Cup final at the National Stadium on Friday: Arab Contractors - Egypt 4 Sodigraf Zaire 0, halftime 2:0 Scorers: Aly Ashour 7', 56'(penalty), Mohamed Ouda 24' 73'.Contractors won 4-0 on aggregate"*, where the surface form *"Zaire"* disambiguates to Wikidata entity `Q974`. Figure 2.3 illustrates how the entity `Q974` *"Democratic Republic of the Congo"* is represented in the KG. The two major design partitions for RDF KGs are: i) The Schema, which defines the ontological terms (also referred to as Terminologies Bos / T-Box) consists of the concepts and their relationships that model the real-world objects. ii) The second partition is the Assertions Box. In this partition, the instance of the T-Box concepts are stored as entities, and the instances of the relations are stored as predicates to form triples (subject, predicate, object - s,p,o). With the open-world assumption of KG design, these concepts can be extended, and a limitless number of triples can be added to the A-Box.

Rep.	**Type**	**Subj.** s	**Pred.** p	**Obj.** o	**Synopsis** (s,p,o)
$\mathcal{S}$	Schema	$c \in \mathcal{C}$	$p \in \mathcal{P}$	$c \in \mathcal{C}$	c, p, c
$\mathcal{I}$	Instance	$e \in \mathcal{E}$	isa	$\$ \in \mathcal{C} \cup \mathcal{P}$	$e, isa, \$$
L	Literal	$v \in \mathcal{C} \cup \mathcal{E} \cup \mathcal{R}$	$p \in \mathcal{P}$	$literal$	$v, p, literal$
$\mathcal{T}$	E2E	$h \in \mathcal{E}$	$r \in \mathcal{R}$	$t \in \mathcal{E}$	h, r, t

Table 2.1: *KG* Triple Classification

Table 2.1 shows a generic classification of triples in a KG, in which all the triples fall into one

of the four partitions: Schema, Instance, Literal Triples, and entity to entity (E2E) triples. Strictly modeled, the concepts in a RDF-Schema are represented using the triple format: (c, `rdf:type`, `rdfs:Class`), such that all $c \in C$ are instances of `rdfs:Class`, while the properties are represented in the triple format: (p, `rdf:type`, `rdfs:Property`) indicating that all $p \in \mathcal{P}$ are instances of `rdf:Property`. instances of a schema concept $c \in C$ become a subject of a given property $p \in \mathcal{P}$, if the special triple: (p, `rdfs:domain` , c) exists in $\mathcal{S}$. On the other hand a property in the Schema can either take instances of a concept or literal values. Instances of a Schema concept $c \in C$ become objects of a property $p \in \mathcal{P}$, if the special triple: (p, `rdfs:range` , c) exists in $\mathcal{S}$. properties which take instances of concepts are referred to as object properties, where as those that take literals are datatype properties. For all $(s, p, o) \in \mathcal{T}^+$, we therefore have the following tautologies in a RDF Graph:

1. $(s,$ `rdf:type` $, c_1) \in \mathcal{I} \implies (p,$ `rdfs:domain`$, c_1) \in \mathcal{S}$
2. $(o,$ `rdf:type` $, c_2) \in \mathcal{I} \implies (p,$ `rdfs:range` $, c_2) \in \mathcal{S}$
3. if 1 & 2 then $(c_1, p, c_2) \in \mathcal{S}$

Item	**synopsis** $(s \mid o)$	**Example** (s, p, o)	**Verbalised**
Domain	$p \in \mathcal{P} \mid c \in C$	wdt:P30, rdfs:Domain, wd:Q6256	continent domain Country
Range	$p \in \mathcal{P} \mid c \in C$	wdt:P30, rdfs:Range, wd:Q5107	continent range Continent
SubClass	$c \in C \mid c \in C$	wd:Q6256, wdt:P297, wd:Q1048835	Country subclass of Political Territory
SubProperty	$p \in \mathcal{P} \mid p \in \mathcal{P}$	wd:P276, wd:P1647 , wd:P361	location sub property of part of
inverse	$p \in \mathcal{P} \mid p \in \mathcal{P}$	wd:P36, wdt:P1696, wd:P1376	capital inverse property of capital of
Instance	$c \in C \mid c \in C$	wd:Q974, wdt:P31, wd:Q6256	Democratic Republic of the Congo instance of Country

Table 2.2: Example *KG* representation of properties and relations. Where: $p \in \mathcal{P}$ denotes properties, and $c \in C$ denotes Classes/Concepts in the schema

Knowledge Graph Entity Context

Definition 2.1.5 (Knowledge Graph Entity Context) *A knowledge graph context (KG-Context) is a single fact from the KG expressed as a triple s, p, o - subject, predicate, object according to the definition of the set $\mathcal{T}^+$.*

Entities in a KG naturally exhibit these triples. Given the classification of KG triples in table 2.1 only the triples in the KG schema (i.e. set $\mathcal{S}$) do not apply to any specific entity. In this work we view an entity as a collection of two sets of contextual information namely: i) knowledge that refers to the attributes of the entity which we denote as $\mathcal{A}^e$, and ii) information that refers to the relationship of an entity to other entities, obtained by following the outgoing edges of the entity, which we denote as $\mathcal{T}^e$.

1. $\mathcal{A}^e = (s, p, o) \in \mathcal{I} \cup \mathcal{L} \mid s = e$ *all instance of and literal triples where the head entity is e.*
2. $$\mathcal{T}^e = \begin{cases} hop = 1 & \bigcup_{i=1}^{outDegree(h)} (F(h), F(r), F(t) \\ else & (h_{hop}, r, t) \in \mathcal{T} \mid h_{hop} \nrightarrow h_{hop-1} \end{cases} \quad (2.1.1)$$

Inducing Semantic Knowledge in Deep Learning Models In this section, we have defined *"Knowledge Context"* especially in the purview of a knowledge graph (KG). We take a systematic look at the underlying technologies that enabled research into KG that culminates in Semantic Web technologies such as RDF for knowledge representation. We saw the need to describe the KG from the theoretical definition of a Graph to the structure and information represented in a KG. Having this in-depth view of how a KG is structured and the depth of knowledge represented is vital in addressing the challenges we define in chapter 1. For example challenges 2, 3, and 4 in section 1.1 involve accessing attributes of an entity (challenge 2), or triples (challenge 3) or evaluating the relevance of different forms of information from the KG (challenge 4). This thesis will combine semantically rich knowledge from several knowledge sources with the power exposed in machine learning concepts to achieve targeted research objectives. In the next section (section 2.2) we introduce concepts in Machine Learning that will be vital in our contribution chapters (cf. chapters 5, 6, and 7) in next section.

2.2 Machine Learning

Machine Learning (ML) is a fundamental part of Artificial Intelligence, with several applications implementing ML algorithms to observe patterns from data. The recent growth in hardware and computing power of machines, coupled with the increased availability of data published on the web, has enabled huge growth in research and implementation of ML algorithms. Deep Learning is a branch of ML that has recently received major attention due to ability to capture deep latent semantic features in data. This behaviour allows for learning of representations that can be reused to perform several tasks in NLP. In this section, we provide an overview of the ML techniques that have been used in the contributing chapters.

2.2.1 Support Vector Machines (SVM)

The Support Vector Machine (SVM) is an optimal margin algorithm that finds a hyperplane (an optimal separation line) to accurately separate two or more different classes in a classification problem. An adequate classes separation is realised by obtaining a hyperplane with the largest distance to the nearest training data points. The data points that are the most difficult to classify, lie closest to the hyperplane between the two or more classes. These points are referred to as *"Support Vectors"*. Generally, the larger the margin or distance between the support vectors, the easier it is for the algorithm to classify accurately. Hence, once the hyperplane is optimised, it is assumed as the optimal separation or the maximum margin classifier. Several fields have applied the SVM to solve various well-known real-world problems including: image classification [91], text characterisation [92], biomedicine [93], and time series prediction [94, 95] to mention but a few, which justifies the popularity of SVM.

Linear vs Non-Linear Classification

Linearly separable data is simple to classify and consists of classes that can be separated using a single straight line passing between two groups. A linear SVM is used in this classification category and essentially uses a one dimensional plane for classification. However, most data in real applications are non-linear by nature. To tackle this challenge, the SVM maps the original feature space to some higher-dimensional space where the training set is separable. This transformation must be an affine transformation to preserve relevant dimensions of relatedness between data points. This allows the resultant classifier to still generalise well. The SVM algorithm employs a *"Kernel"* to be able to perform such transformation. SVM classifiers are solved by computing the convex Lagrange dual of the max-

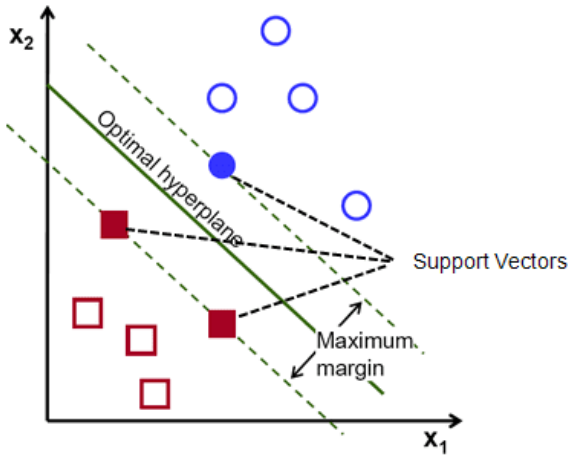


Figure 2.4: SVM: Illustration of the Hyperplane and the Support Vectors(source: [96])

margin SVM formulation as follows (Lagrange multipliers are determined by solving the problem using Quadratic Programming) [97]:

$$f(x) = \sum_{i=1}^{N} \alpha_i \cdot y_i \cdot K(x, x_i) + b \tag{2.2.1}$$

where b and α_i are learned parameters, N is the number of support vectors, i is a support vector instance, t is the vector of training instances, y_i is the class value of a particular training instance of vector t, and a(i) is the vector of support vectors.

The Kernel Function

For SVM, we apply the Kernel function K according to the choice from the following different forms:

- **Linear Kernel**

$$K(x, y) = x^T y \tag{2.2.2}$$

- **Polynomial**

$$K(x, y) = (x^T y + 1)^d \tag{2.2.3}$$

- **Sigmoid**

$$K(x, y) = tanh(ax^T y + b) \tag{2.2.4}$$

- **Radial Basis Function (RBF) kernel**

$$K(x, y) = exp\left(-\gamma \|x - y\|^2\right) \tag{2.2.5}$$

- **Gaussian Radial Basis Function (RBF) kernel**

$$K(x, y) = exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \tag{2.2.6}$$

Support Vector Regression (SVR)

Regression aims at finding a straight-line function $f(x) = Wx + b$ subject to the condition that the obtained $f(x)$ value is within a certain accuracy (ξ) of all data points. SVR gives us the flexibility to define the amount of acceptable error in our model and will find an appropriate line (or hyperplane in higher dimensions) to fit the data. The objective function of SVR is to minimise the regularisation coefficients, especially the l2-norm of the coefficient vector (fundamentally differing from the Ordinary Least Squares (OLS) problem that minimises the squared error). The error term is instead handled in the constraints, where we set the absolute error less than or equal to a specified margin, called the maximum error, ϵ (epsilon). Varying the value of epsilon to finetunes the model to attain the desired accuracy.

Our covering of Support Vector Machines as background information is vital to our contributions presented in this thesis in chapter 7

2.2.2 Neural Networks

Artificial Neural Networks (ANN) commonly called Neural Networks (NN) are statistical learning algorithms that learn linear or non-linear functions from given data. The simplest neural network is the *Perceptron*: a combining function that accepts several real numbered inputs and provides a single output after passing the combined value through an activation function. For example, given a set of values (x_1, x_2, x_3), each of these values is multiplied by some learnable weight. Thus the set of weights is given by: (w_1, w_2, w_3). These weights are the parameters in the form of real numbers that describe the underlying mathematical function which maps the given inputs to the desired output value. Often, the function obtains the value through a weighted sum of inputs. The resultant value is passed through an activation or transfer function. The simplest Perceptron employs the threshold activation (also known as the step function) where the neuron's output is set to 1.0 if the sum ($\sum w_i x_i$) is greater than a real number threshold value or zero otherwise. For a perceptron that is employed as a supervised binary classifier, the activation function is relevant to ensuring the output is mapped between required values (0,1) or (-1,1). Noteworthy is that the learned weight of input points to the strength or its contribution to the overall output value. Similarly, an input's bias value gives the ability to shift the activation function curve up or down.

$$y = W^T x + b \tag{2.2.7}$$

Where $x = (x_1, x_2, \ldots, x_n)$ is the input, W is the weight matrix and b is the bias. The Perceptron is essentially a Single-Layer Neural Network that consists of four parts: input values, weights and bias, net sum, and the activation function whose output is linear by nature. In a *Multi-Layer Perceptron (MLP)*, several of these constructs (also referred to as neurons) are stacked together such that the output of each layer feeds into the next subsequent layer until the final output is obtained. Given that this output always remain linear no matter the number of layers in the network, there is need to apply nonlinearity to the final layer's output (the expected output determines the type of non-linear function to be used). This transformation from linear to non-linear is performed via different (non-linear) activation function, as shown below:

$$y = \sigma(W^T x + b) \tag{2.2.8}$$

Where σ stands for the applied activation function. There are a number of possible activation functions that have been introduced over the years including: *sigmoid*, *the Hyperbolic Tangent Function (Tanh)*,

Rectified Linear Unit (ReLU) Function [98], *Gaussian Error Linear Units (GELU* [99], *Leaky ReLU*, *Swish*, and Maxout activation [100]. The type of activation function used differentiates a MLP from an *Artificial Neural Network (ANN)*.

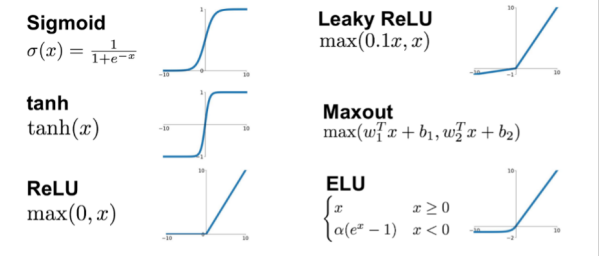


Figure 2.5: Various activation functions (source: [101])

Neural networks employ the Back-propagation algorithm [102] for optimisation. On the high level, the back-propagation uses the gradient optimisation approach (gradient descent) [103] to minimise a cost function of the network.

Deep Neural Network (DNN)

The recent increase in computational power and the emergence of specialised hardware for numerical calculation stability has allowed for extended design of the neural networks. A deep neural network (DNN) (a deep feed-forward network) consists of several intermediate *hidden layers* of neurons between the first (input) layer and the last (output) layer, where the forward connection allows for information flow. Figure 2.6, illustrates a simple comparison between a deep vs a simple neural network. The deep neural networks have more learnable parameters that allow the ability to solve complex mathematical functions [104]. For example, simply two hidden layers having quadratic size can sort N N-bit numbers [104] by grasping the more complex mathematical function. DNN's layers use different activation functions (few examples are shown in figure 2.5, over the different layer level outputs while propagating inputs to the final output values. The high achievements of the deep neural network in the machine learning community prompt us to apply its some of the functionality in our proposed approach. DNNs have played a significant role for solving the tasks of different fields including medical operation [105], image recognition [106], speech recognition [107], natural language processing (NLP) [108, 109], and recommendations systems [110]. With the growth of computing resources, such DNNs have found its direct usage in many real-world applications.

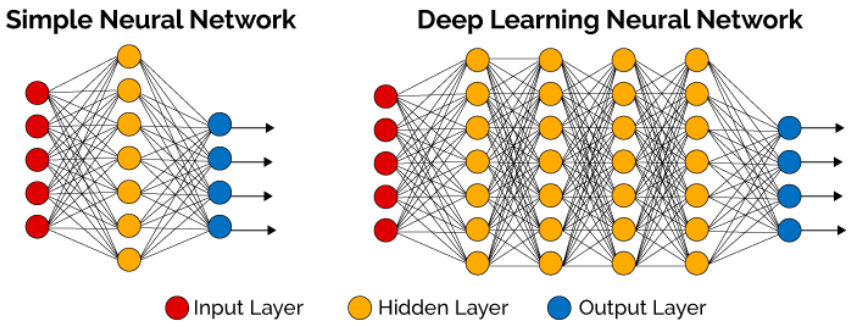


Figure 2.6: Deep Neural Network vs Simple Neural Network where DNN includes more than one hidden layer (source:[111]).

Recurrent Neural Network (RNN)

[112], temporal data like time series [113], text[114], speech[115] etc. The RNN adopts its internal state as a memory cell to capture information from input data at time intervals such that the final output depends on all previous state (memory cells). Recently, a type of RNNs is used in encoder and decoder part of machine translation system NLP task [116, 117]. Whereas, the feed-forward network takes input as a vector and propagates the layer-level outputs through all hidden layers to yield the final output, the flow of information is unidirectional (forward). Intermediate outputs can be viewed as *network output states*. However, in sequential input data, the parts of a sequence are related to each other, and decisions can not be made only based on the current output state. For instance, the question *"How many people live in the capital city of Australia?"* each word has a semantic relation to both the previous and next words. Words are fed as input vectors into the network. However, for a simple RNN, there are no backward connections between the output layer and rest of the layers hence a basic RNN may not capture the information of all previous words [102, 118].

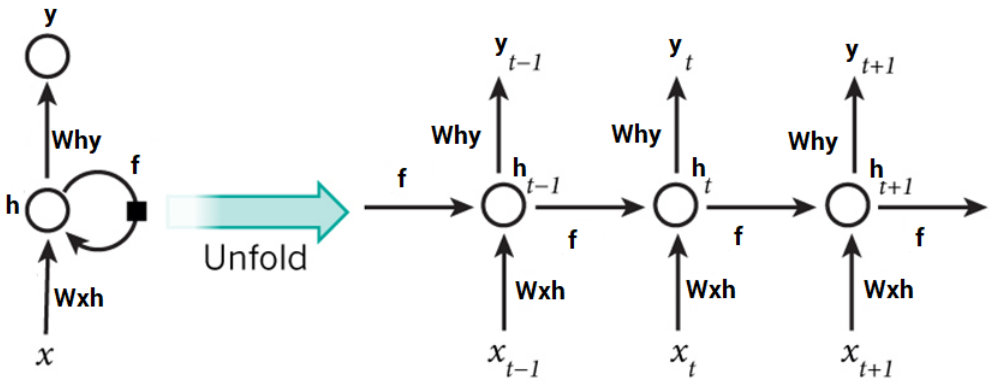


Figure 2.7: **Recurrent Neural Network** where x_t: input vector, h_t: hidden state vector and y_t:output vector at time-step t and loop feed information from previous step $t-1$ to next step t. W_x, W_y and W_h are weight matrices between separate nodes connections [119]

.

The figure 2.7 shows that the current output state of the network not only depends on the current input state but rather, also on the previous output states. Contrary to the DNNs, the RNN shares learnable parameters across all time steps; hence the number of parameters in RNNs are significantly reduced. RNNs employ separate weight matrices (**W**) for both forward (current input to current output) and backward (previous output to current output) connections (equation 2.2.9).

$$h_t = f(W_x x_t + W_h h_{t-1}) \tag{2.2.9}$$
$$y_t = g(W_y h_t)$$

Where f, g are the different activation functions. Hidden states h_t of the network are memory cell vectors that encapsulate the information from all previous states and transfer it to subsequent states. For natural language processing related tasks, this information is regarded as contextual knowledge. RNN relies on the gradient-based *Backpropagation Through Time (BPTT)*[118] algorithm to perform loss minimisation and optimise or update the learnable parameters (BPTT operates to reduce the margin between predicted and target outputs. Research has applied RNNs with good success in fields of text

classification, speech recognition, and machine translation [117, 120], natural language processing (NLP) [116]. The challenge for RNNs comes into play in tasks with long input sequences. Due to *exploding gradient* and *vanishing gradient* problem [121, 122], RNNs suffer from memory loss while performing the propagation of information between the hidden layers. This challenge causes no convergence during training; the model does not get to the optimal minimal value. To solve this problem, researches introduced the *Long Short Term Memory Networks (LSTM units)* [121, 122]. 2.2.2 briefly elaborates how LSTM units are used in a recurrent neural network to address this problem.

Long Short Term Memory Network (LSTM-N)

When the final output is derived from several previous states in the sequence, RNNs incur the exploding and vanishing gradient problem. RNNs perform effortlessly in short texts. E.g. to predict the last word *"Kenya"* in the sentence *"Raila Odinga was the second prime minister of Kenya"*, an RNN-based language model is sufficient. Because the gap between the desired (last) words and the rest of the words is relatively small, allowing relevant contextual knowledge to be gathered. On the contrary, the sentence *"In March 2018, President Uhuru Kenyatta accepted the now-famous Handshake after he held a long meeting with the Opposition leader Raila Odinga to help cool the temperatures in his government in Kenya"*. The last portion *"his government in Kenya"* has a contextual connection to knowledge from the earlier parts *"president Uhuru Kenyatta"*. Long Short-Term Memory Network (LSTM-N) models are proposed to counter the long-range dependency of sequential data in RNNs [30, 123, 124]. LSTM is a RNN with its own LSTM cell depicted in figure 2.8.

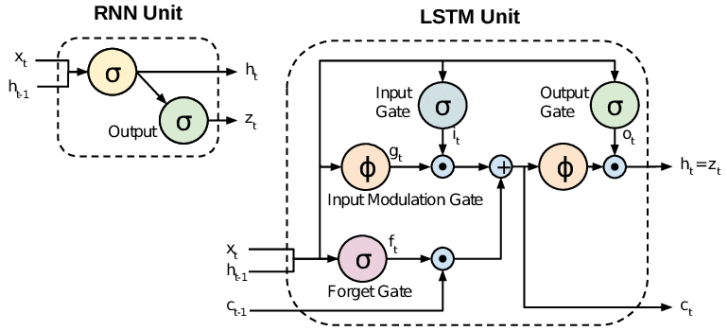


Figure 2.8: Architecture of LSTM Cell (source:([125]))

The LSTM cell [30, 126] is constituted of two state variables i.e. hidden state (h) and cell state (c). This assists the model to store and reference the network state over time intervals during the processing of the sequence. At every time step, the two cell parameters get updated. Equation 2.2.10 elaborates the underlying mathematical computations in a LSTM cell. The hidden state (h_t) and cell state (C_t) vectors for each time step t are computed from the activation vectors (input (i_t), forget (f_t), output (o_t)) and candidate vector ($\tilde{C}_t$) of LSTM unit. The following set of equations show the update steps for the state variables of an LSTM unit [30, 126].

$$
\begin{aligned}
i_t &= \sigma(W_x^{(i)} * x_t + W_h^{(i)} * h_{t-1} + b_i) && = \sigma(W_i \cdot [x_t, h_{t-1}] + b_i) \\
f_t &= \sigma(W_x^{(f)} * x_t + W_h^{(f)} * h_{t-1} + b_f) && = \sigma(W_f \cdot [x_t, h_{t-1}] + b_f) \\
o_t &= \sigma(W_x^{(o)} * x_t + W_h^{(o)} * h_{t-1} + b_o) && = \sigma(W_o \cdot [x_t, h_{t-1}] + b_o) \\
\tilde{C}_t &= \tanh(W_x^{(C)} * x_t + W_h^{(C)} * h_{t-1}) + b_C) && = \tanh(W_C \cdot [x_t, h_{t-1}] + b_C) \\
C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\
h_t &= o_t * \tanh C_t
\end{aligned}
\tag{2.2.10}
$$

where: σ and tanh represent Sigmoid function and Hyperbolic tangent function respectively. W_m is a weight matrix and b_m is a bias of m gate ($m \in \{i, o, f\}$).

The simple LSTM cell shown in In figure 2.8 is also known as a memory cell. The input gate(i) performs read operations, while the output gate(o) perform write, and the forget gate(f) performs erase / delete operation[127]. These gates are used to control information flow corresponding to the amount of information to read, write and erase from the memory cell at different time steps. Similar to RNNs, LSTM-Networks parameters are also trained using the BPTT algorithm [118].

Sequence To Sequence Models

A sequence to sequence network [117, 128] uses the recurrent neural networks in its architecture. It also referred to as Encoder-Decoder network. Each part of the network (encoder and decoder) uses a different recurrent neural network. The encoder takes an input sequence or source sequence and transforms it into a context. The context vector, also known as thought vector expresses the whole meaning of the sequence. The decoder interprets the context vector to generate a target sequence over different time steps. Our basic architecture of the training model is inspired by the papers "Sequence to Sequence Learning with Neural Networks" [117] and "Effective Approaches to Attention-based Neural Machine Translation" [120]. Both deep neural networks and vanilla recurrent neural networks cannot translate a source sequence to a target sequence consisting of variable lengths. For instance, in machine translation, one language texts (source sequences) are translated into another language texts (target sequences), but both sequences have the variable lengths in different sentences.

A vanilla DNN can be used when the input and output dimension is fixed. A vanilla RNN can take the variable lengths sequences as inputs to encode them; however, without second RNN, we can not decode the context vector to generate the target sequence. Therefore as suggested above, to deal with variable length (or variable time dimension) of input and output sequence problem, we use an Encoder and a Decoder in the network. The encoder takes a sequence of fixed-size vectors (word vectors in our case), and its final state returns an encoder-state (context vector or thought vector) as shown in figure 2.9. The decoder takes this vector as initial input and generates a sequence of output vectors.

In our approach, we use Long Short-Term Memory (LSTM) cell instead of Recurrent Neural Network (RNN) cell in encoder and decoder parts of the model. We previously concluded that vanilla RNN suffers from gradient exploding or gradient vanishing problem and the network is not able to collect enough contextual information of a long sequence for further processing[30, 123, 124]. As discussed in 2.2.2, LSTM cells are better to hold contextual knowledge or high temporal dependencies in a long sequence. A Long Short-Term Memory network layer (Encoder) can encode the semantic information of the entire input sequence to produce another sequence with another LSTM layer (Decoder).

Basic Sequence To Sequence Model:

A basic sequence to sequence model architecture [117, 128] is illustrated in figure 2.9. Model network maps a sequence of variable length (inputs:$(x_1, x_2, x_3, ..., x_N)$) to the another sequence of variable length (outputs: $(y_1, y_2, y_3, ..., y_M)$). This flow of information is presented in the figure 2.9. Encoder and decoder of the model employ equations 2.2.10 for the training. The model network uses separate LSTM layers for encoder and decoder.

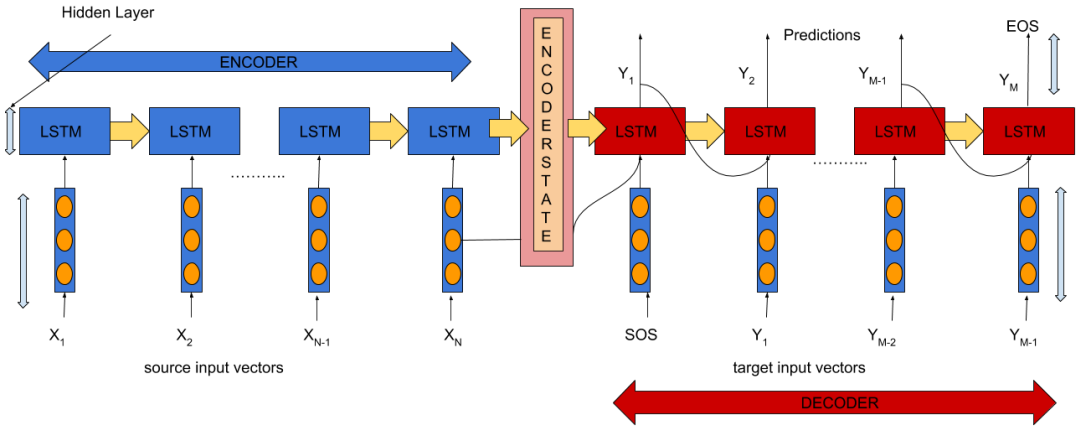


Figure 2.9: **Encoder-Decoder architecture of the Model using LSTM unit**— Encoder interprets input sequence and generates encoder state. The Decoder reads this state to predict outputs until EOS (End of Sequence). SOS represents Start of Sequence). Each y_t is obtained using softmax [129] over the vocabulary of the target sequence.

Model behaviour can be represented by conditional probability [117]:

$$p(y_1, y_2, y_3, ..., y_M | x_1, x_2, x_3, ..., x_N)$$

where N and M are the lengths (time-dimension) of the input sequence $(x_1, x_2, x_3, ..., x_N)$ and output sequence $((y_1, y_2, y_3, ..., y_M))$ where we want to maximize the probability p for the given input sequence. To compute this conditional probability p we first draw the fixed size vector encoder state v from the given input sequence $(x_1, x_2..., x_N)$. It is achieved by the last hidden state of the LSTM layer of the encoder, and then computed the probability of $y_1, y_2..., y_M$ with a standard LSTM-LM formulation [114] whose initial hidden state (encoder state v) extracts the essence of input sequence $(x_1, ..., x_N)$ [117].

$$p(y_1, y_2..., y_M | x_1, x_2, x_3, ..., x_N) = \prod_{t=1}^{M} p(y_t | v, y_1, y_2..., y_{t-1}) \tag{2.2.11}$$

In machine translation task (eg. translating English language to German language) inputs and outputs are word-embeddings (word-vectors) of vocabulary size (K). In equation 2.2.12, it is illustrated that decoder LSTM layer outputs (h_m) are feed into a softmax [129] layer for probability distribution to predict target word(j) with the highest probability.

$$P(y = j | h_m) = \frac{\exp(h_m^T w_j)}{\sum_{k=1}^{K} \exp({h_m}^T w_k)} \tag{2.2.12}$$

Similar to RNN, we use BPTT [118] optimisation algorithm to train network model's learn-able

parameters. Training loss of the network is calculated using negative log-likelihood loss function 2.2.14 [130](described in section 2.2.3).

2.2.3 Neural Network training

In training a neural network to learn patterns in data, several concepts are used to describe aspects of the data, the models and the training process.

Weight Initialisation

Weights (w) refer to the parameters of the neural network that are needed to be learnt [131]. Initial values of weights influence the learning process of the model. Therefore, appropriate initialisation of weights can help faster and efficiently learn during model training. There are different methods to initialise weights, as mentioned below.

1. ***Zero Initialisation:*** initialises the weights of a neural network to zero value. The training model learns nothing with zero Initialisation of weights as it cannot retain its enough complexity for learning and acts as a linear model.

2. ***Random Initialisation:*** Generally, weights are initialised with a standard normal distribution. If the weights are initialised with very high value or low value, deep neural network suffers from exploding, and vanishing gradient problem [121, 122, 131, 132]. Below is a brief explanation:
   - **Exploding Gradients:** High values of weights create a problem of exploding gradient. This is due to the substantial, large error gradient of a model during weights multiplication over the layers during the optimisation process. Network computations regarding weight update provide numerical instability, and it leads to overflow (NAN) values. Therefore, the training model loses its stability and learns nothing.
   - **Vanishing Gradients:** Low values of weights create a vanishing gradient problem as they are multiplied over the layers during optimisation, which can go as low as zero value weights. A low value of error gradients has no impact on the weight update. Under such small weight changes, the model training assumes convergence before obtaining the actual minima.

Learning Rate

A learning rate is an important hyper-parameter while training a neural network model [129]. It plays a significant role in the model for converging to local minima. It is also termed as a step size for weight update towards the optimisation direction (update or move weight in the negative direction of the gradient to reach local or global minima) to minimise loss of the network. If the model learning rate is too high, then loss never converges to local minima. Moreover, if the learning rate is too slow, then the learning model takes many iterations for weights updates before converging to minima. In equation 2.2.13 we explain weight θ_1 update using a gradient of loss function w.r.t weight θ_1.

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta) \tag{2.2.13}$$

where: θ_1Weight, α = Learning rate
J Loss function

Gradient Clipping

Clipping the gradient norms of the model's parameters to a maximum norm during training is an important technique to prevent gradient exploding. It stops the model's gradient parameters from increasing exponentially in recurrent neural networks for the long sequences. It assists the learning model to converge smoothly. Without gradient clipping, a model is prone to miss the local minima hence not converge [129].

Dropout

Dropout is a widely used regularisation technique to prevent neural network model from overfitting [133]. An overfitted model performs better on training data. However, it fails to produce the same performance on unseen test data. The overfitted model becomes so complicated to learn a problem solution effectively on the given training data. It loses the regularity, which decreases its performance on unseen test data. We balance the complexity of the model by dropping some random units from the neural network.

Loss

The loss function of the neural network is used to measure the performance of the training model. The supervised learning model algorithm strives to match the predicted values with the target value for the given training data in training. An enormous loss function value signifies that the model's predicted values are too far from the actual values (target values). We optimise our model's parameters by reducing the loss function value.

In our proposed approach, we consider negative log-likelihood (softmax cross entropy) loss function 2.2.14 as a loss function [129]. Our model delivers a prediction of a target sequence (in words) using source sequence (in words) therefore in equation 2.2.14 k and j represent "target word" and "target data vocabulary size" respectively and f is a vector.

$$J_k = -\log(\frac{e^{f_k}}{\sum_j e^{f_j}}) \tag{2.2.14}$$

Gradient Computation and Optimisation

Deep neural networks are trained using Backpropagation algorithm ("*backward propagation of error or loss*") [102]. Backpropagation algorithm uses gradient descent as an optimisation technique to minimise the cost. A cost function measures the deviation between the target and predicted values by a neural network model. The algorithm calculates gradients of the cost function with respect to model's learn-able parameters (eg. weight and biases) using chain-rule [$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$] and delta-rule. Model's parameters are updated iteratively using the error gradient calculated as described in equation 2.2.13. Usually, the network's training stops when the cost function is minimised, which means network converge to local or

global minima. We illustrate the Backpropagation algorithm in 1.

Algorithmus 1 : Back Propagation algorithm

Input Data: $(X, Y)_{n=1}^{n=N}$ training examples and learning_rate;
initialise Weights of the network randomly;
while *Iterate until the average loss is decreasing* **do**
    Forward pass to compute the output of the network;
    Calculate loss or error at the output layer;
    Backward pass to calculate gradients using the chain and delta rules;
    update all weights;

optimisation Algorithms

1. ***Stochastic Gradient Descent (SGD)*** It is the most widely used optimisation algorithm for the training of neural network models. SGD is defined for a single example (k) and a batch of size n from training data in equation 2.2.15 [134].

$$w := w - \alpha \nabla J_i(w) \tag{2.2.15}$$
$$w := w - \alpha \sum_{n=1}^{n} \nabla J_i(w)/n$$

2. ***Adaptive Moment Estimation (ADAM)*** Stochastic Gradient Descent applies a single learning rate (α) for each parameter whereas ADAM [135] updates the learning rate of each network learning parameter with the help of the sparse gradient. Adam algorithm computes a running average of the first and second moment of the gradient for adaptive learning rate for weights. It has achieved a good result in high dimensional parameters spaces of NLP and computer vision problems. In our training model, we use ADAM optimiser for better and fast convergence.

2.2.4 Learning Distributional Representation

A word vector (also referred to as word-embedding) represents a word in the form of a one-dimensional vector of the real numbers [50, 136–138]. Word embeddings have been adopted widely in the machine learning models for natural language processing (NLP) tasks [139, 140]. Many models are available to represent words into vectors. For instance, Word2Vec [141], Glove [51], and FastText[13] are popular models for learning word-embeddings. The word-embedding consolidates contextual knowledge of a word using different dimensions. Machine learning model (neural networks) accepts word embeddings as inputs for training. PCA and t-SNE are machine learning algorithms to transform a high dimensional word vector into a low dimensional word vector. It is illustrated in figure 2.10.

To elaborate more about this concept, consider an example of a book. First, we scale the categories of books into the real values from -1.0 (false) to 1.0 (true). A multi-dimensional space is defined for the categories. Each dimension represents a different category. We then interpret the dimensions of the categories in the order as follows (Mystery-Thriller, Literature-Fiction, Children-Book, Teens, Science-Fiction, Cooking-Food-Wine, Romance). For instance, a book has a word vector (0.2, 1.0, 1.0, −1.0, 0.0, −1.0, 0.0). From the given book vector, we can conclude that the book mainly belongs to the categories Literature-Fiction and Children-Book and very less with other categories. Similarly, other books can be vectorized.

[13] https://en.wikipedia.org/wiki/FastText

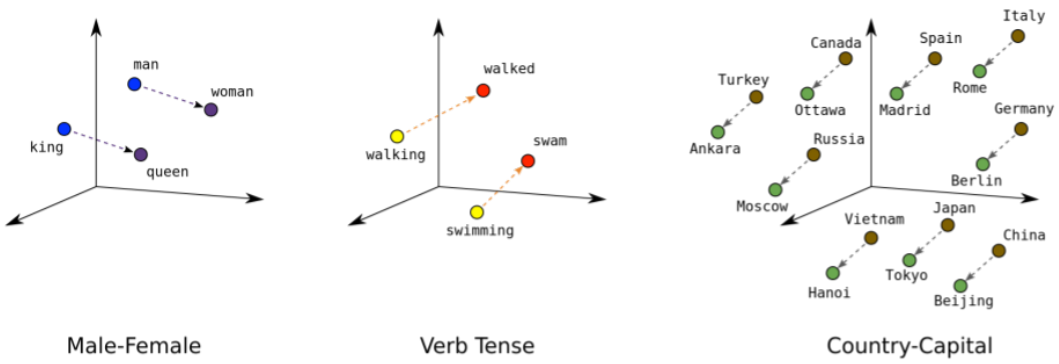


Figure 2.10: Word Embedding Examples, It represents how contextually or semantically words are close to each other in the vector space. [142]

If two books are similar in categories, then their cosine similarity is 1.0. Hence, we can conclude these books have very close semantic relation.

A domain of data on which word embeddings are learned is an influential deciding factor for the training of the sequence to sequence model NLP tasks.

Glove Word Embedding

We use glove[14] pre-trained word embeddings of 300 dimension for our training model. Generally, word embedding is trained using supervised and unsupervised machine learning model or algorithms. The glove (Global Vectors for Word Representation) which is learned through an unsupervised learning model, trains pre-defined word vocabulary to get word vectors [51]. This model includes the benefits of local context window and global matrix factorization models. Glove pre-trained word vectors or embeddings are trained on Wikipedia-2014 and Gigaword-5 corpus having 6 billion tokens and 400K vocabulary. This word embedding is available in different dimension 50, 100 and 300 [15].

Embedding Layer and Embedding Matrix

Embedding layer is the first layer of a sequence to sequence neural network (text sequence) to train and to provide computational efficiency to word embedding matrix. Embedding matrix is defined as $W_e \in R^{KXD}$ where K is the size vocabulary and D is the dimension of word embedding. Each word of the vocabulary has a unique index value. For embedding lookup, word index of the word is used to access its vector in the embedding matrix.

There are various methods to create embedding matrix, and all are listed below:

1. We can initialise a random vector for each word of the embedding matrix and then learn it while training of the neural model. It requires a large corpus for better representation of word vectors.

2. Embedding matrix initialised by using available pre-trained word embedding from Word2Vec[16],

[14] https://nlp.stanford.edu/projects/glove/
[15] http://nlp.stanford.edu/data/glove.6B.zip
[16] https://code.google.com/archive/p/word2vec/

Glove and FastText embedding models. In our case, the dataset is not so large to train word-embedding from scratch. Therefore we employ pre-trained word embedding of Glove[17].

Transformers and Language Modelling

The concepts of word Embedding, previously discussed have been a great breakthrough in Natural Language (NL) understanding because researchers and developers can now attain vectors or numerical representation of NL tokens. However the earlier models such Word2Vec [50], and Glove [51] provided static word representations which contain contextual embedding but could not be adapted to new domains and changes in context. The publishing of the transformer-based neural networks architecture [31] provided an avenue to capture specific relevant context over long text ranges. Inspired by this, Several researchers began investigating research directions to provide more dynamic contextual embeddings, while at the same time be able to transfer model parameters into new tasks (*Transfer Learning*). This has given rise to an array of Language models including: ELMO [143], The Generative Pretraining (GPT and GPT-2) [144, 145], Deep Bidirectional Transformers (BERT) [32], Attentive LM beyond fixed-length vectors [146], Generalised Autoregressive Pretraining for Language Understanding (XLNet) [52] to mention but a few. Numerous other models have been released in this line, and the research community envisions that this general research direction will continue to be explored in the foreseeable future.

2.3 Summary

In this chapter, we took a look into the background concepts that are relevant to our work. In section 2.1 we presented approaches and data formats used to represent information on the semantic web. We determine that these techniques have become de facto standards for structuring data on the web into construct referred to as Knowledge Graphs (KGs). Such KG exhibit rich structure and open wealth of information that continuously grows. For our work, we are keenly interested in capturing this information and any extra inferential signals that can be obtained from online knowledge bases, especially the KGs as defined in section 2.1.3. However, so often, we need to input such contextual information into machine learning models in numerical values or vectors. As such we proffer in section 2.2 a set of machine learning concepts to introduce both the models we shall use for our work and encode the knowledge context. This combination of the power presented in the semantic web with machine learning is the advantage we attain in our work. We defined deep learning concepts in section 2.2.2 since these have certainly become indispensable in modern-day Artificial Intelligence (AI) research. Although powerful and able to capture in-depth features from data, these models miss the rich semantic information specific to the tasks such as question answering, entity linking, relation linking etc. Hence, we propose to explore the value of semantically rich information available in public KGs (explored in section 2.1). Further, we consider how such information can assist these deep learning models (section 2.2) in solving specific challenges of entity and relation linking effectively.

[17] https://nlp.stanford.edu/projects/glove/

CHAPTER 3

Related Work

This chapter presents a targeted summary of the state-of-the-art approaches associated with the main research problem and research questions articulated in Chapter 1. We aim to give a brief overview of the solutions available in literature while sowing the relative use of knowledge context to solve the challenges. Although the tasks are mature in the NLP community with authority datasets that have been studied for nearly a decade (e.g. for entity linking: the CONLL-AIDA dataset [147], MSNBC [25], ACE2004 [148], AQUAINT [148] and the QALD dataset series [64], and for relation extraction), we determine that they are far from solved. To begin with, a summary of state-of-the-art Relation Linking approaches is given in section 3.1. We identified that most linking systems concentrate on engineering complex algorithms to capture the local and global context of entities and relations. In addition, a great number of these approaches assume that entities are already recognised and rely on predefined candidate lists. As we shall see, this is not always applicable, especially in short text communications such as Question Answering. Section 3.2 describes some of the existing approaches for Entity Linking (Both end to end EL and disambiguation only approaches). This is refined to gradually show how knowledge context is beginning to gain traction. In section 3.3, we describe the open-source knowledge graphs and term networks relevant for use as knowledge context in our work. Section 3.4 discusses the existing efforts in literature that have already incorporated knowledge context for the task of disambiguation. Finally, we provide a summary in section 3.5.

3.1 Relation Linking and Short Text

Relation extraction (RE) is a well-known task in natural language processing (NLP). This task was first formulated as part of the Message Understanding Conference (MUC) in 1998 [149]. In the field of NLP and machine learning, researchers have addressed this problem using different approaches. The work in [149, 150] introduces a kernel-based machine learning methods on parse trees that can be learned through Support Vector Machines or Voted Perceptrons for relation extraction in given natural language text. Probabilistic models have also been applied for open RE. Initially, researchers such as TEXT RUNNER [151] use a probabilistic model in the form of Naive Bayes model on language textual features. While Markov Logic Networks [152] and Conditional Random Fields [153] have been identified as possible avenues to improve the extraction accuracy. RelEx [42] uses dependency parse trees and applies a few simple rules to these trees to extract relation from free text. The recent success in RE can be attributed to the availability of vast training data curated using distant supervision [154]. Methods for distant supervision assume that if two entities have a relationship in a KG, then all sentences containing those entities express the same relation, this may sometimes lead to noise in the data. To overcome the

challenges, researchers in [59] initiated the multi-instance learning followed by [155] which extracted relation from a bag of sentences. For detailed survey on multi-instance RE, please refer to [156].

Knowledge Context: There have already been some attempts to use aspects of information from knowledge graphs and other knowledge bases to supplement learned features from text. For instance, Entity descriptions from Freebase [58] and Wikipedia [157] pages task were used in [158] to supplement background knowledge and attained performance improvements. The RESIDE approach [35] utilises entity type together with the aliases for both the relation and entities in the model. However, RESIDE does not incorporate the entity descriptions that were applied in [158]. RELE [159] jointly learns embeddings of structural information from KGs and textual data from entity descriptions to improve relation extraction.

Relation Extraction for Question Answering: We review the efforts in the community that have carried out relation linking in the challenging short text scenario. To achieve this, we look at the question-answering task and the research efforts related to relation linking for QA. The relation patterns KB: PATTY [49] is a popular work which is used in many question-answering systems for linking relations to the underlying knowledge base properties. PATTY mines semantically-typed relational patterns from large corpora. However, it can not be used directly as a component in a QA system but needs to be modified based on individual developer requirements. For example, AskNow QA system [40] has a dedicated component for the relation extraction task that uses PATTY as a large underlying corpus to find semantic relational patterns. TBSL [160] and LODQA [161] implement a two step process to directly translate a natural language question into a SPARQL query. During this translation process, TBSL uses BOA pattern [162] identifiers to detect properties (i.e. relations) that connect the entities present in the input question. Moreover, additional work such as [163] presents a question answering approach using Freebase that implements a neural network-based relation extractor to retrieve answers from Freebase. Although these QA approaches implement relation extraction and linking tasks, it is not trivial to reuse this specific module in other QA approaches due to the monolithic implementation of their QA pipeline. For example, they are reusing it in frameworks such as OKBQA[1], QANARY/ Frankenstein [2, 164–167], and openQA [168] that allow QA developers to build QA systems or pipelines adopting many existing Question Answering components. These frameworks provide an infrastructure allowing developers to implement QA tasks as individual modules. The description in this section forms a foundation for the problem and solution described in our contributions in chapter 4.

3.2 Entity Linking

In the previous section, we provide an overview of Relation Linking (RL) systems over structured data. In addition to Relation Linking, we take a look at the progress made in the Entity Linking task and the progress that has been made in respect to including KG context to assist the task. This section concentrates on the approaches for entity linking, laying foundation for the contributions in chapters 5 and 6.

End-to-end Entity Linking

Research has traditionally treated Named Entity Recognition and Disambiguation (NERD) (also referred to as Entity Linking (EL) or Named Entity Resolution) as a three-step process involving Entity spotting, candidate selection, and disambiguation [169]. A wide range of tools and research work exists in the area of NER and NED which can mainly be attributed to the fact that NER/NED (jointly as NERD)

[1] http://www.okbqa.org/

task is closely similar for free text as well as question answering as restricted domains albeit with a few differences. The tool TagMe [170] is one of the popular works in this area that links elements in short texts to their corresponding Wikipedia pages. It uses a dictionary of entity surface forms extracted from Wikipedia to detect entity mentions in the parsed input text. These mentions are then passed through a voting scheme that computes the score for each mention-entity pair as the sum of votes given by candidate entities of all other mentions in the text [170]. Finally, a pruning step filters out less relevant annotations. Researchers in [171] used the term Wikification to refer to this process, the difference being that Wikification is not only restricted to named entities but rather any keyword terms in the text are identified and linked to a Wikipedia article. The process of extracting these keywords is similar to that of entities in that it involves a controlled vocabulary that acts as a look-up for n-grams which ultimately form candidate entities when matched to the vocabulary. Candidate entities are then ranked via various algorithms and passed through a special Word Sense Disambiguation (WSD) algorithm to resolve them into Wikipages [171]. We can observe the constant need to have background knowledge assisted models wherein these two early works; researchers sought to represent knowledge context through look-up tables and dictionaries.

More research that performs NED on Wikipedia includes the approach proposed in [172] for collective NED through local hill-climbing, rounding integer linear programs, and pre-clustering entities. In contrast, [25] maps the surface forms of entities together with contextual information then employs vector space models to perform the disambiguation. Likewise, researchers in [173] try to disambiguate emergent entities. Many of the approaches rely on already existing NER tools such as Stanford NER[2],CoreNLP[3],GATE[4] or OpenNLP[5] among others, for the spotting stage. Notwithstanding, there is existing work J-NERD [174] which attempts to jointly perform both NER and NED (NER/D) using probabilistic graphical models. In these probabilistic models, they treat both tasks as a sequence labelling task through linear-chain CRF and dependency parse tree based tree factor graphs.

Recently, following the popularity of knowledge graphs (KGs), scholars have shifted focus to use KGs such as DBpedia [1], Freebase [16] and Wikidata [17] for the NED task. DBpedia Spotlight [36] is one such tool that performs NED on DBpedia. After an initial step of entity spotting, DBpedia Spotlight uses contextual information to resolve an entity's surface forms to corresponding DBpedia resources. The tool Babelify [175] on the other hand, employs an underlying graph structure of a lexicalised semantic network to perform word sense disambiguation of words in a given text. The next step is to generate a graph that is finally used for linking through graph walk algorithms. Another graph-based system AGDISTIS [176], relies on the number of hops between entities for disambiguation. These tools carry out NED as a two-step process utilising the Stanford NER [6] for the initial step. These tools offer interfaces for use by other applications in the form of APIs; however, NERD [177], proposes a framework for unifying results of these tools for easy usage and combination via an ontology for alignment. It is important to note that most of these approaches use state of the art machine learning techniques and require a large amount of training data. However, when these tools are applied to short text in a new domain such as question answering, the performance is limited [45]. This gives rise to the question concerning how extra knowledge context can be represented to enhance performance, especially in scenarios where the local context is not sufficient.

In general, relation and entity linking for short text remain open research areas for the community, and it would be expected that there shall be more tools to solve the RE/RL and NER/D tasks. These tools

[2]https://nlp.stanford.edu/ner/
[3]https://stanfordnlp.github.io/CoreNLP/
[4]https://gate.ac.uk/
[5]https://opennlp.apache.org/
[6]https://nlp.stanford.edu/ner/

find direct applicability areas of biomedical information extraction, or in collaborative component-based QA frameworks such as OKBQA [178], QANARY [179], and Frankenstein [18] where input is relatively very short.

Components for Named Entity Recognition and Disambiguation The Named Entity Recognition (NER) recognises the subsequence of text that refers to an entity while the entity disambiguation (NED) links these entities to their mentions in a referent knowledge base (e.g., for DBpedia [1]). For instance, in the example *"Soccer: Late goal gives Japan win over Syria"*, NER component ideally recognises `Japan` as an entity while a tool for NED task link it to its Wikidata mention `wd:Q170566`[7][8]. Below is a list of some of the NER and NED components.

1. **Entity Classifier** uses rule base grammar to extract entities in a text [180]. Its REST endpoint is available for wider use for NER task.
2. **Stanford NLP Tool**: Stanford named entity recogniser is an open-source tool that uses Gibbs sampling for information extraction to spot entities in a text [181].
3. **Babelfy** [175] Attempts multilingual EL through a graph-based approach that uses random walks and subgraph algorithm to identify and disambiguate entities present from text [175].
4. **AGDISTIS** [176] is a graph-based disambiguation tool that combines a novel HITS algorithm with label expansion strategies. Further, string similarity measures are used to disambiguate entities in a given text [176].
5. **DBpedia Spotlight** is a web service[9] that uses vector-space representation of entities and using the cosine similarity, recognise and disambiguate the entities [36].
6. **Tag Me** matches terms in a given text with Wikipedia, i.e., links text to recognise named entities. Furthermore, it uses the in-link graph and the page dataset to disambiguate recognised entities to their Wikipedia URls [182]. Tag Me is open source, and its REST API endpoint[10] is available for further (re-)use.
7. **Other APIs**: Besides the available open-source components, there are many commercial APIs that also provide open access for the research community. Aylien API[11] is one of such APIs that use natural language processing and machine learning for text analysis. Its text analysis module also consists of spotting and disambiguation entities. TextRazor[12], Dandelion[13], Ontotext[14] [5], Ambiverse[15], and MeaningCloud[16] are other APIs that have been providing open access to researchers for their reuse.

Several comprehensive surveys exist that detail the techniques employed in entity linking (EL) research; see, for example, [169]. An elaborate discussion on NER has been provided by Yadav Bethard [183].

[7] `wd` corresponds to `https://www.wikidata.org/wiki/`
[8] `Q170566` is the Wikidata ID (Q-valu) for the entity *"Japan National Football Team"*
[9] `https://github.com/dbpedia-spotlight/dbpedia-spotlight`
[10] `https://services.d4science.org/web/TagMe/documentation`
[11] `http://docs.aylien.com/docs/introduction`
[12] `https://www.textrazor.com/docs/rest`
[13] `https://dandelion.eu/docs/api/datatxt/nex/getting-started/`
[14] `http://docs.s4.ontotext.com/display/S4docs/REST+APIs`
[15] `https://developer.ambiverse.com/`
[16] `https://www.meaningcloud.com/developer`

However, the use of knowledge graphs as background knowledge for EL task is a relatively recent approach. Here, a knowledge graph is not only used for the reference entities but also offers additional signals to enrich both the recognition and the disambiguation processes. For entity linking, FALCON [24] introduces the concept of using knowledge graph context for improving entity linking performance over DBpedia. Falcon creates a local KG fusing information from DBpedia and Wikidata to support entity and predicate linking of questions. We reused the Falcon Background knowledge base and then expanded it with all the entities present in the Wikidata (primarily non-standard entities). Another work in the similar direction is by Seyler et al. [184]. Authors utilise an extensive set of features as background knowledge to train a linear chain CRF classifier for the NER task.

The developments in deep learning have introduced a range of models that carry out both NER and NED as a single end to end step using various neural network-based models [37]. Kolitsas et al. [37] enforces during testing that gold entity is present in the potential list of candidates, however, Arjun doesn't have such assumption and generates entity candidates on the fly. This is one reason Arjun is not compared with Kolitsas's work in the evaluation section. Please note, irrespective of the model opted for entity linking, the existing EL approaches and their implementations are commonly evaluated over standard datasets (e.g. CoNLL (YAGO) [147]). These datasets contain standard formats of the entities commonly derive from Wikipedia URI label. Recently, researchers have explicitly targeted EL over Wikidata by proposing new neural network-based approach [73]. Contrary to our work, authors assume entities are recognised (i.e. step 1 of Arjun is already done). Inputs to this model is a "sentence, one wrong Wikidata Qid, one correct Qid"; using an attention-based model predicts correct Qid in the sentence- more of a classification problem. Hence, 91.6 F-score in Cetoli et al.'s work [73] is for linking correct QID to Wikidata, given the particular inputs. Their model is not adaptable for an end to end EL due to input restriction. OpenTapioca [185]) is an end-to-end EL approach on Wikidata that relies on topic similarities and local entity context. Howbeit, it ignores the Wikidata specific challenges (chapter 5). Works in [10, 24] are other attempts for Wikidata entity linking.

Mention Detection (MD): The first attempt to organise a named entity recognition (NER) task traced back to 1996 [186]. Since then, numerous attempts have been made, ranging from conditional random fields (CRFs) with features constructed from dictionaries [187] or feature-inferring neural networks [188]. Recently, contextual embedding based models achieve state of the art for NER/MD task [32, 189]. We point to the survey by [183] for details about NER. Few early EL models have performed MD task independently such as [190, 191].

Candidate Generation (CG): There are four prominent approaches for candidate generation. First is a direct matching of entity mentions with a pre-computed candidate set [192]. The second approach is the dictionary lookup, where a dictionary of the associated aliases of entity mentions is compiled from several knowledge base sources (e.g. Wikipedia, Wordnet) [193–195]. The third approach is to generate entity candidates using empirical probabilistic entity-map $p(e|m)$. The $p(e|m)$ is a pre-calculated prior probability of correspondence between positive mentions and entities. A widely used entity map was built by [196] from Wikipedia hyperlinks, Crosswikis [197] and YAGO [147] dictionaries. End-to-end EL approaches such as [37, 198] relies on the entity map built by Ganea and Hofmann. The next approach for generating the candidates is proposed by [24]. Authors build a local KG by expanding entity mentions using Wikidata and DBpedia entity labels and associated aliases. The local KG can be queried using BM25 ranking algorithm [199]. The modular architecture of Arjun 5 gives us the flexibility to experiment with several ways of generating entity candidates. Hence, we reused candidate list proposed by [196] and built a new CG approach based on [24] for the second implementation in section 5.3.

End to End EL: Few EL approaches accomplish MD and ED tasks jointly. [174] propose joint recognition and disambiguation of named-entity mentions using a graphical model and show that it improves EL. [198] combine local and global features using knowledge about the neighbouring mentions and

respective entities to solve EL task. Work in [37] also proposes a joint model for MD and ED. Authors use bi-LSTM based model for mention detection and compute the similarity between the entity mention embedding and set of predefined entity candidates. This approach first selects a set of entities with a high local score and computes the similarity between the in-process entity embedding and an average of the selected entity embeddings. The work in [39] employs BERT to model three subtasks of the EL jointly. The authors use an entity vocabulary of 700K top most frequent entities to train the model. Work in [200] uses a Transformer architecture with large scale pre-training from Wikipedia links for EL. For CG, authors train the model to predict BIO-tagged mention boundaries to disambiguate among all entities. For Wikidata KG, Opentapioca is an entity linking approach which relies on a heuristic-based model for disambiguation of the mentions in a text to the Wikidata entities [185]. Our Arjun [201] approach discussed in chapter 5 offers flexibility in the candidate generation process by leveraging a modular system to EL.

3.3 Knowledge Graphs for Contextual Representations

Several knowledge bases available online have been built using semantic web technologies. These methods that allow design and structuring of vocabularies and ontologies play an important role in allowing access to information on the web. Such knowledge repositories can be used as knowledge sources (e.g. for question answering) or as sources of features to inform machine learning models. **DBpedia** [1, 85] is a conspicuous examples made up of cross-domain dataset of structured data extracted from Wikipedia articles (infoboxes, categories, etc.). The DBpedia Ontology is "a shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia".[17]. Users can configure annotations to their specific needs using the DBpedia Ontology.

The YAGO knowledge graph [202] combines semantic knowledge fetched from the Wikipedia knowledge base (e.g., categories, redirects, infoboxes) with the taxonomy of WordNet (e.g., synsets, hyponymy) and GeoNames. As of 2019, YAGO3 [15] has knowledge of more than 10 million entities and contains more than 120 million facts about these entities. YAGO links temporal and spatial dimensions to many of its facts and entities.

Wikidata [17] is a large, curated open domain KG, that depends on the crowd to provide content while also incorporating information from Wikipedia. The main characteristics of Wikidata include the ability for the community to directly and openly edit the data. Wikidata also allows the incorporation of facts based on different sources. Hence, there can be a plurality of facts, and information such as time-sensitive data or data that varies over a period of time (e.g. the president of the United States) can also be included easily by qualifiers and ranks. This is possible because Wikidata envisions hyper-relational statements [203] representation that employs the qualifiers to represent more detailed information.

Freebase: [58, 204] was a large knowledge base designed as an open, community-curated repository of knowledge based on the semantic web. The designers of this KG [58] had the intention to support highly diverse and heterogeneous data simultaneously while maintaining high scalability. As of 2014, Freebase constituted more than 40 million topics consisting of a little over 2 billion facts, making it the most comprehensive publicly available source of general-knowledge facts until its termination in 2016. Other Knowledge graphs such as the Wikidata and DBpedia previously discussed have continued to be updated to date and merged, in part, some of the facts contained in the deprecated Freebase KG. The complete Freebase data is available as data dumps[18] for free use, sharing, and adaption under a creative

[17] http://wiki.dbpedia.org/services-resources/ontology

[18] The University of Freiburg, for example, have an online share: https://freebase-easy.cs.uni-freiburg.de/dump/

commons license. The format used to represent Freebase data is the N-Triples RDF and can be loaded into any state-of-the-art triple store for querying via standard semantic query languages such as SPARQL. Freebase also provided an own API that required a special query language: Metaweb Query Language (MQL).

Term Graphs: are special types of graph networks that model relationships between terms and phrases of consistent meaning within a language. **Wordnet** [46, 205] was proposed to provide an effective combination of traditional lexicographic information and modern high-speed computation. A striking difference between WordNet and standard dictionaries is the fact that WordNet divides the lexicon into five categories: nouns, verbs, adjectives, adverbs, and function words. While dictionaries organise lexical in terms of word forms, Wordnet employs word meanings to structure its content. As such, the lexical relations of Synonymy, Antonymy, Hyponymy, and Meronymy are unique word relations that make wordNet key for use in language understanding. Over the years, WordNet has evolved to incorporate the names and location of entities. This makes WordNet more than just a term graph but rather a hybrid ontology and KG. Other Term graphs include **BabelNet** [48] that aims at providing Multilingual interfacing of terms and entities from Wikipedia and WordNet. It is an extensive, wide-coverage multilingual semantic network. BabelNet contains more than 3 million concepts and covers 52 noun senses in WordNet and is originally made up of lemmas in 6 languages and is continuously growing. **ConceptNet:** [47, 75] is a special-purpose knowledge graph that connects terms (words and phrases) in NL via assertions (labelled, weighted edges). Originally, ConceptNet [47] aimed at providing a curated representation of "Open Mind Common Sense", and inherently crowd-sourced knowledge. ConceptNet 5.5 [75] extends the original ideas of ConceptNet to incorporate lexical and world knowledge from different sources and multilingual aspects. With the growth of the semantic web, ConceptNet included links to other knowledge graphs such as WordNet, Wiktionary, OpenCyc, and DBPedia.

Although these well structured and often highly curated knowledge graphs exist and continue to grow, much research effort has often relied on textual descriptions of entities and relations. **Wikipedia** is the largest open-domain knowledge repository that offers textual descriptions of resources indexed by their URLs. Each page on Wikipedia represents a unique entity created or edited by the community. Due to its volume and growth, many linking datasets have employed the Wikipedia KB as the referent KB, naturally leading to it's use to provide knowledge context. E.g. the approach presented in [28] employs Wikipedia abstracts as entity descriptions to enhance an attentive NN.

3.4 Knowledge Context Enabled Models

Earlier in this chapter, we looked at the approaches for relation linking. A comprehensive survey of entity linking (EL) techniques is provided by [193]. In this subsection, we focus on the related work closely associated with our contribution, in chapter 5 and chapter 6 namely: background knowledge to support EL. Work in [219] proposed joint modelling of entity linking and coreference resolution to reduce entity linking errors. The use of entity descriptions as additional information has been widely used in the research community. Work in [28, 220] use first paragraph of Wikipedia entity descriptions as additional context. Entity types and aliases were other contexts that have been exploited in the literature for the background knowledge [220, 221]. However, these contexts were derived mostly from Wikipedia. Mulang' et.al. [10] argued that Wikipedia entity descriptions introduce other unnecessary information, and the concise human-curated Wikidata entity descriptions offer further improvements to the overall performance. Authors first used two transformer models (XLNet [52], and RoBERTa [53]) and fed Wikidata triples as context. The results showed that adding Wikidata triples as a different entity context has improved the transformer's performance. In the next experiments, the authors use an attention-based

Approach	Labels	Aliases	Description	Instance	KG Structure	Types
End-to-End Models						
DBpedia spotlight [36]	✓					
OpenTapioca [185]	✓	✓			✓	✓
Falcon [24]	✓	✓				
Falcon 2.0 [206]	✓	✓				
Kolitsas et.al. [37]	✓					
NED-Graphs [73]	✓	✓	✓	✓	✓	
Hedwig [207]	✓	✓	✓	✓	✓	
VCG [208]	✓	✓			✓	
KBPearl [209]	✓	✓			✓	
PNEL [210]	✓	✓			✓	
Huang et. al. [211]	✓	✓			✓	
Boros et. al. [212]						✓
Provatorov et. al. [213]	✓	✓				
Labusch [214]						
Botha et al. [215]						
Tweeki [216]	✓	✓	✓			✓
Disambiguation Only						
Yamada et.al. [29]	✓		✓		✓	
Ganea&Hofmann [196]						
Yang et al. [217]			✓			✓
Le&Titov [218]			✓		✓	
DeepType [34]						✓
Fang et al. [194]			✓			
Le& Titov [9]						✓
DCA [28]	✓		✓			
Chen et al. [57]						✓

Table 3.1: Analysing the use of Knowledge Context Entity Linking Models: Forms of knowledge context information used by different EL Models.

deep neural network to feed Wikidata descriptions and positively impact overall performance. Table 3.1 shows the extent to which knowledge context has been employed in EL modles.

Falcon [24] built a background knowledge graph by aligning DBpedia and Wikidata entity labels. This background KG is used as a source of entity candidates in the EL process. Falcon 2.0 [206] uses a similar background KG to support EL over Wikidata. In our contribution: Arjun [201] described in chapter 5 a pipeline of two neural networks induced with the KG context derived from Wikidata entity aliases is proposed. The first attentive neural network (ANN) identifies the entities' boundaries in a given text. The second step expands each entity mention with a set of candidates derived from the Falcon background KG. The last step uses a second ANN fed with entity mention and associated candidates to predict the target entity. Arjun's empirical studies established the role of KG context in deep neural networks for EL task aiming for Wikidata. Cetoli et al. [73] and work in [10] use Wikidata triples in a neural model for NED task over Wikidata. Another implementation of Arjun (section 5.3) seeks to build upon the finding of Arjun [201], and we analyse the impact of Wikidata context on Wikipedia based entity disambiguation

model. Hence, contributions extended in this thesis targeting this direction are as follows: 1) analysing the impact of various forms of KG context (entity alias, label, descriptions, triples) on two different models: DCA [28] and XLNet [52]. 2) Provide an understanding of the impact of KG context for semi-structured knowledge base (i.e., Wikipedia) based named entity disambiguation models. Furthermore, entity context has also been used in various other domains besides EL task. For instance, [222] utilise entity context to support link prediction. Wang et al. [223] introduced KGAT, a knowledge-aware recommendation framework representing an initial attempt to exploit structural knowledge for effective recommendations. RESIDE [35], and RECON [22] induces KG context to support relation extraction. For KG completion, work in [224] proposes the notion of schema-correctness in KG embeddings.

3.5 Summary

In this chapter, we have taken a survey of the relation and entity linking landscape. In section 3.1 we discussed the state-of-the-art in Relation Linking and the challenges involved and realised the gap in the literature. First, we observed that most Relation linking tools and the task involves a scenario in which entities are already linked. This extends a disadvantage, especially in situations where the Relation linking must be performed in an end to end fashion as experienced in Question Answering and short texts. Secondly, Relation Extraction has existed in the community for over a decade; however, end-to-end RE (RL) has not been attempted until our work described in chapter 4. This could explain the fact that the community has not attained the necessity for incorporating extra knowledge context. With this gap, we proposed our approach that inherently relies on knowledge context to perform end-to-end RL in short text environments presented in chapter 4. In section 3.2 we then described the Entity Linking (EL) literature and state of the art in this task. As opposed to RL, EL has been more extensively explored with researchers already beginning to appreciate the value of knowledge context. Table 3.1 gives a summary of entity linking approaches and how they incorporate contextual information. Albeit, the level of use of knowledge context and approaches for encoding the same is still rudimentary. Hence there remains a gap for improving the performance of models in this task. In our approaches presented in chapters 5 and 6 we discuss incremental avenues for encoding and selecting context for entity linking. Lastly, we observed that application of knowledge context is only at a preliminary stage with researchers only recently beginning to introduce aspects of KG context into models.

CHAPTER 4

Unifying Knowledge Graph and Text Representations for Relation Linking

The previous chapters elaborated on the research problems, challenges, and state-of-the-art approaches to relation and entity linking. We also demonstrated in section 3.1 that relation linking (RL) tools and approaches majorly perform linking based on pre-disambiguated entities. This means that they do no lend easily in different scenarios, such as short text communication. Although the community has studied relation linking for several years, we observe that it is still a very open research problem. Therefore, we enlist the following observations:

- Several approaches for relation linking assume that entities are already linked, concentrating on just identifying the underlying relation. When the task exposes pure text without recognised entities, it becomes a challenge for most relation lining approaches.
- Relation Linking is a significant aspect of several NLP tasks, such as Question Answering (QA). In the QA scenario, the short text extends several challenges such as those discussed in Challenge 1 of section 1.1. Moreover, the QA context demands a zero-shot relation linking approach where both relations and entities are not recognised.
- Due to communication brevity shot text often lacks enough semantic context, e.g. the relation is implicit rather than explicitly mentioned. In some cases, there are little other aspects of the texts that support the relationship to be linked, e.g., no mentioned entity. Finally, several examples exist in which several relations occur in the same question. All the relations must be linked to obtain the answer to the question. For example, the question *"How many people live in the capital city of Australia?"* from figure 1.2 has two relation: *"capital city of"* and *"live in"*. The second relation has a reliance on the implicit numerical desire of the question.
- Since the community datasets rely on relations with already identified entities; there is a lack of datasets for supervising machine learning algorithms. This results from the fact that Question Answering datasets are not annotated with relations.

  Hence, considering the challenges of Relation Linking, there is a need for an intuitive approach that relies on insights and features from data to match the natural language and knowledge graph relations.

In addition to these challenges, in this chapter, we address an extra relevant problem. Question answering systems implement QA tasks either by dedicating individual components in their architecture

to each task or by combing a few tasks together in their implementation. In component-based QA systems and frameworks like OKBQA[1], QANARY [166], QALL-ME[225], openQA [168], researchers have implemented individual components dedicated to particular tasks. Stanford NER [2], NERD [3], Alchemy API, FOX[4], AGDISTIS[5] are some of the most popular dedicated tools/components for specific tasks like named entity recognition, named entity disambiguation in QA systems. This contribution was the first step towards an independent web service/tool/component that performs relation extraction for natural language questions over KGs. We identify this as a major research gap in collaborative question answering system development. Creating a standalone and reusable component for relation extraction and linking in this context would facilitate the reuse of the component in different QA systems and create a benchmark for the research community for future comparison and evaluation. We provide a novel approach, and an implementation, that addresses some of the challenges above via the following features:

- It is capable of dealing with large KGs such as DBpedia;
- It addresses the lexical gap problem through the combination of different similarity measures;
- It is designed as an independent component that can be easily reused in different QA systems.

Approaches for relation extraction over KG attempt to first retrieve from the KG the named entities identified in a question, together with a list of their KG predicates, then selecting one from all the entity's predicates. In our approach, we match natural language relations (or predicates) extracted from the questions directly with KG properties that can be employed within QA pipelines. *First*, we model KB properties with their underlying parts of speech. These are then enhanced with extra attributes obtained from taxonomies like Wordnet[6] and dependency parsing characteristics. *Second*, from a question, we model query relations using a similar representation. *Third*, we define similarity measures between the question query relations and the KG properties representations to identify which property is referred to by each relation within the question. We exclude PATTY [226], a large corpus of relational patterns and associated DBpedia predicates due to its noisy behaviour. For example, in an input question *Who is the wife of Donald Trump?*, natural language pattern *wife of* which is appearing in the question is associated with DBpedia relations like `dbo:parent`, `dbo:predecessor`, `dbo:successor`, `dbo:child`, `associatedMusicArtist` and many other in PATTY corpus. Hence, direct usage of PATTY knowledge base will cause more noise in retrieved relations for an input question than improve overall performance. For our work, we performed evaluation using the QALD-5 dataset [227], which consists of over 400 questions together with the corresponding formal queries (SPARQL) to be applied against DBpedia. Positive results have been shown with this evaluation in terms of accuracy (reaching almost 48% precision with questions containing one relation) but especially in terms of recall values (75% recall with questions having one relation). In this way, we will establish, for the first time, a conceptual view of the existing QA systems. Therefore, the following research question is addressed in this chapter:

[1] http://www.okbqa.org/
[2] https://nlp.stanford.edu/software/CRF-NER.shtml
[3] http://nerd.eurecom.fr/
[4] http://aksw.org/Projects/FOX.html
[5] http://aksw.org/Projects/AGDISTIS.html
[6] "About WordNet". WordNet, Princeton University. 2010. http://wordnet.princeton.edu

Research Question 1 (RQ1)

How can we achieve a unified representation of both knowledge graph and textual relations to enhance similarity matching?

Contributing publication: Mulang' et.al. [11]
All experiments for this publication were carried out by the PhD candidate, who also handled the writing of the paper.

The contributions in this chapter address the first research question **RQ1** as stated above. This chapter is derived from the publication ReMatch [11]: We approach matching NL relation to KB properties by processing the two complementary sides of the problem, namely the natural language (query side) and the knowledge graph side. The aim is to provide an equal representation for both sides that would lead quickly to a comparison. We then employ a set of syntactic and semantic similarities measures to select which property matches each relation in the question best. Figure 4.1 depicts the overall structure of the system.

The rest of the chapter is structured as follows. The introductory portion of section 4.1 motivates the problem with a real-world example. The section 4.1.1 describes the representation for the relations in a question while section 4.1.2 illustrates the representation of a KG relation. The target is to have these two representations exhibit similar comparable composition. In section 4.2, we describe the similarity measures chosen and how they have been employed to score the relationship between text relation and candidate relations from the KG. Section 4.3 presents our results for this approach. We also look at the impact in the overall research community this work has had since the contribution was published. We summarise in section 4.4.

4.1 Unifying Representation of NL and KG Relations

We approach the problem of matching NL relation to KB properties by processing the two complementary sides of the problem, namely the natural language (query side) and the knowledge graph side. The aim is to provide a similar representation for both sides that would lead easily to a comparison. We then employ a set of syntactic and semantic similarities measures to select which property matches each relation in the question best. We motivate our work by considering a natural language question such as *"What is the capital of Australia?"* to be asked in a QA system as shown in figure 1.2. For this question, *"capital of"* is the natural language (NL) relation. In QA domain, a relation extraction process goes a step further compared to a typical relation extraction task in NLP. It links the identified relation in an input question to its mentions in a KB (e.g. DBpedia, Freebase etc.) available on the Web. In our example, the entity *"Australia"* has its DBpedia property `dbo:capital` which needs to be mapped to the relation *"capital of"* by a relation mapping tool/component of any question answering system. Hence, the input for a relation mapping tool is an NL question, and the output is the RDF property in a knowledge graph of the associated named entity. As such, for the exemplary question *"What is the capital of Australia?"*, the expected output from a relation linking/extraction tool is the property *"http://dbpedia.org/ontology/capital"* (when using DBpedia as KB). Figure 4.1[7] depicts the

[7]Numbers 4.1 to 4.3 in figure 3 indicate the respective section in the paper where each component is described

overall structure of the system.

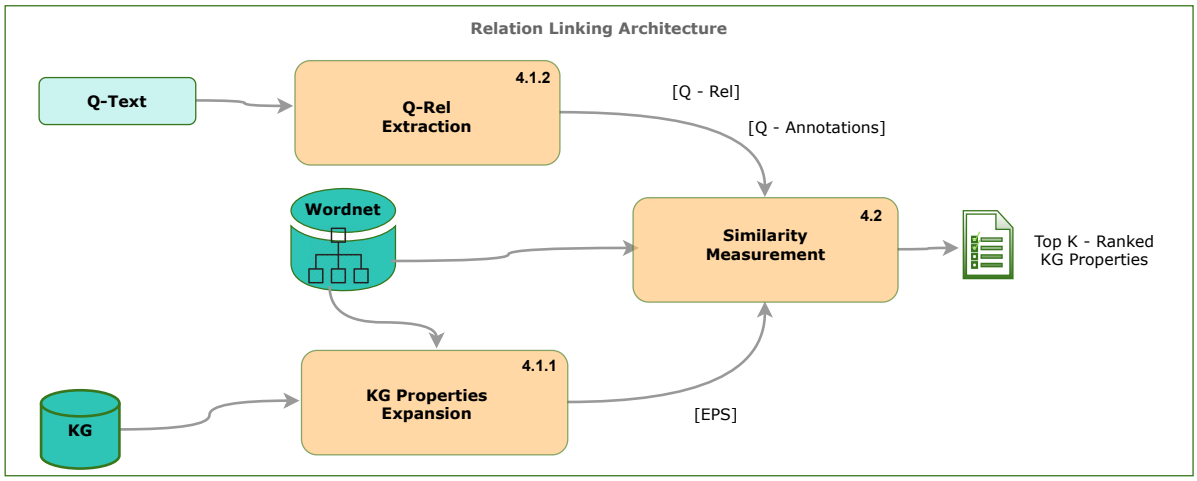


Figure 4.1: Overall relation matching system architecture: from a question (Q-Text) as input to a ranked list of top K properties in the KG matching the relations in the input question

4.1.1 KG Properties Expansion

A KG property is defined by a directed labelled edge between two nodes of the graph that is identified via a unique URI. Properties can be visualised in two levels within a KG, on one level they can be conceptual as found within the structural definition of the KG. In this case, they connect two concepts referred to as the range and the domain of the property. The domain and range of a property are conceptual representations of real-world entities. The second view of a property is as a predicate within a factual instance in the KG. in which the property URI is a link between two entity objects which are themselves instances of the domain and range. Since the target of our work is to produce a tool that can be used within QA pipelines, we adopt the first view in this work. The second view demands first to disambiguate the named entities before matching the properties.

We develop a data structure which we refer to as the `Expanded Properties Set (EPS)` that contains a URI for each property within the KG (in our experiment, DBpedia properties), augmented with characteristics present within the KG and annotations obtained from the syntactic analysis. At this stage, we only consider extracting synonyms and hyponyms from a taxonomy like Wordnet and ignore elements related to the derivational forms. Thus we retain the structure of the EPS and reduce the memory load time. We observe here that the hypernyms are not required on the properties side of the relation matching process owing to the design characteristics of a KG which entails a taxonomical relationship in which properties are defined as classes within a hierarchy. For example, the property `dbo:child` is a more general concept and would match its hyponyms *"son"* and *"daughter"*. In case the question requires a hypernym of this relation (e.g. `dbo:relative`) then the design structure already captures this hierarchy. A similar approach was employed by Beaumont et al. [41] in which they enhance property labels obtained from the KG with variations from Wordnet. This is necessary since the relation in natural text often does not map directly to the label of the desired property (i.e. lexical gap). For example, the property *"spouse"* does not match its natural language forms *"wife of / husband of"* or *"married to"*. Considering two related concepts, we can enhance the matching of the relation to the property in the KG with a set of natural language patterns that are commonly used to refer to that property [68]. The label attribute of the property provides a natural language mention of the property, commonly one to three

words. In this work, we also consider the comment attribute related to each property in the KG. The comment attribute of an element provides additional textual information about the given property.

In DBpedia there are two sets of properties which can be found either in the DBpedia ontology (*dbo*[8]) namespace or the DBpedia properties one (*dbp*[9]). Out of a possible total of 63,764 items classified as properties in the DBpedia ontology, only about 3,500 have instances within the KG. We identify 2,795 properties[10] defined within *dbo* as key properties for our experiments and fetch the instantiated properties from *dbp*, leading to a total of 4,748 properties represented in the EPS. We consider these properties sufficient to answer questions on DBpedia KG since questions would demand properties that have participated in at least one factual instance within the KG.

Expanded Property Set (EPS)

Definition 4.1.1 (EPS) *Formally, a property $p \in P$, where P is defined in a graph $G = \{S \times P \times O\}$ as the set of all properties in G, is expanded into a septuple $(\varrho, \beta, \lambda, \omega, c, \mu, A)$ such that:*
$\varrho \leftarrow$ *The uri of the property in the KG*
$\beta \leftarrow$ *The text label referring to the domain of the property*
$\lambda \leftarrow$ *The text label of the property*
$\omega \leftarrow$ *The label referring to the range of the property*
$c \leftarrow$ *The count of instances in the KG containing the property*
$\mu \leftarrow$ *A ratio associating unique subjects and unique objects instantiated by the property*
$A \leftarrow$ *Annotations derived from syntactic analysis of the constructed sentence from the other attributes.*

All the elements of a property are obtained directly from the KG except the annotations A. To produce A, we attempt a derived Sentence by concatenating a section of the tuple. In this form, β acts as the subject,λ the relation, and ω the object with the comment appended as a descriptive text of the relation separated by a comma. For example for the property with λ as *"capital"*, $\beta \leftarrow$ *"PopulatedPlace"* and $\omega \leftarrow$ *"city""" '* we constructs the text: *Populated place capital city*. For this relation, there is no comment represented in the KG. To elaborate the role of comments lets consider the property `dbo:spouse` which has both the β and λ elements of value *"Person"* from the class `dbo:Person`. The derived sentence: *Person spouse Person, the person they are married to.* contains a comment that complements the basic triple elements. The sentence is not grammatically complete but rather have a form that can suggest the syntactic structures.

[8] *dbo* stands for: http://dbpedia.org/ontology/
[9] *dbp* stands for: http://dbpedia.org/property/
[10] This figure can be obtained from: http://wiki.dbpedia.org/services-resources/ontology

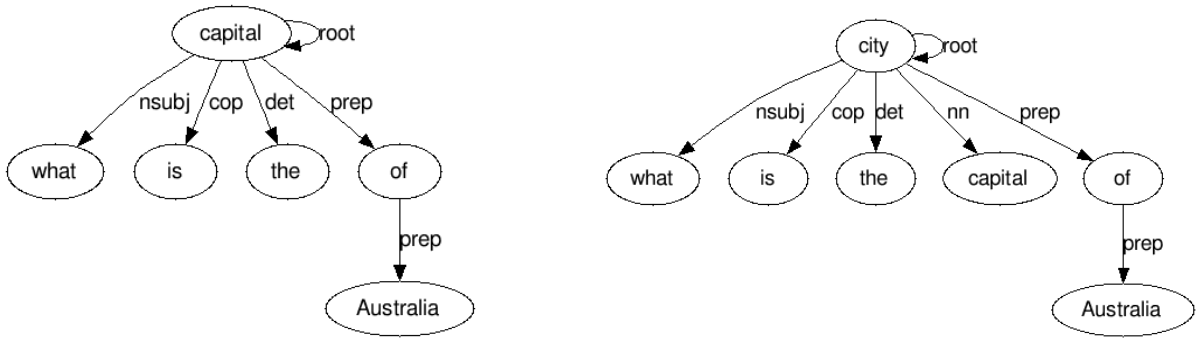
4.1.2 Q-Rel Extraction

The Q-Rel Extraction module receives a Question text in a given natural language (in our context, we use English) and produces a tuple representation of the question containing attributes that would later be used in deriving a similarity score. Questions are often succinct and may lack some distant syntactic and semantic associations that would typically be present in free text. At the same time, questions also inherently contain implicit or explicit characteristics that may not be exhibited in free text. Therefore, we make some assumptions and formulate constraints that would assist in representing a question. We observe that relation extraction for communicating with a KG such as required in the question-answering domain is substantially different from general relation extraction tasks in Open IE. Often, the binary relations extracted from the natural text do not suggest their relation to semantic components in a KG. It is therefore gainful in some cases, to readjust binary relations based on other characteristics within the text. According to [41], a set of phrases within the question can be determined that correspond to semantic components (entity, property and class). In our work, we consider properties as the major semantic component of interest. We assume that a question is either a simple question or is a variably connected set of simple questions. A simple question is a question which exposes only one unique relation [66, 228] and as such the relation can only match one unique property in the KB. Each simple question has a desire, i.e. the type of answer expected [226]. A binary relation can be represented in the logical form $rel(x, y)$ in which rel, and describes the relationship between known or unknown entities x and y [229], and a set of assisted words and symbols. This set of words can be further viewed as named entity nouns, non-named entity nouns and helper words.

Question Relation (Q-Rel)

Definition 4.1.2 (Q-Rel) *In this work, we represent a simple question as a single relation, hereafter referred as Q-Rel. Formally Q-Rel is an octuple* $(\delta,\eta,\alpha,\ell,\gamma,\mathcal{E},\mathcal{N},\Upsilon)$ *where:*
$\delta \leftarrow$ *The question desire*
$\eta \leftarrow$ *The direct helper word to the relation*
$\alpha \leftarrow$ *the relation words in the question*
$\ell \leftarrow$ *The left element in the relation, or the relation head [68]*
$\gamma \leftarrow$ *The right element of the relation or the relation tail [68]*
$\mathcal{E} \leftarrow$ *Possibly empty set of named entities where* $e \in \mathcal{E} \Rightarrow e \notin \{\ell \cup \gamma\}$
$\mathcal{N} \leftarrow$ *Possibly empty set of non entity nouns s.t.* $e \in \mathcal{N} \Rightarrow e \notin \{\ell \cup \gamma\}$
$\Upsilon \leftarrow$ *Possibly empty set of helper words such a dependency preposition.*

Given the simple question: *What is the capital of Australia?*, with the dependency parse tree in 4.2(a) would have the attributes with the values as follows: $\delta \leftarrow$ "location"; $\eta \leftarrow$ "is";$\alpha \leftarrow$ "capital"; $\ell \leftarrow$ **null**; $\gamma \leftarrow$ "Australia"; $\mathcal{E} \leftarrow$ **null**; $\mathcal{N} \leftarrow$ **null**; $\Upsilon \leftarrow$ {of}. For this example, the root *"capital"* of the dependency parse is also the relation word in the Q-Rel. The relation in the question could differ from the root of the dependency tree if the question was asked differently: *What is the capital city of Australia* as shown in 4.2(b). We overcome this difference at the dependency adjustment stage.



(a) (b)

Figure 4.2: Simple question dependency parse trees depicting the difference in dependency structure for the same question asked in different ways. Given the question *"What is the capital of Australia"* with the dependency parse in figure 4.2(a) and question *"What is the capital city of Australia"* whose dependency parse is illustrated in figure 4.2(b). The two questions differ by the word: *"city"* yet the dependency structures are relatively different.

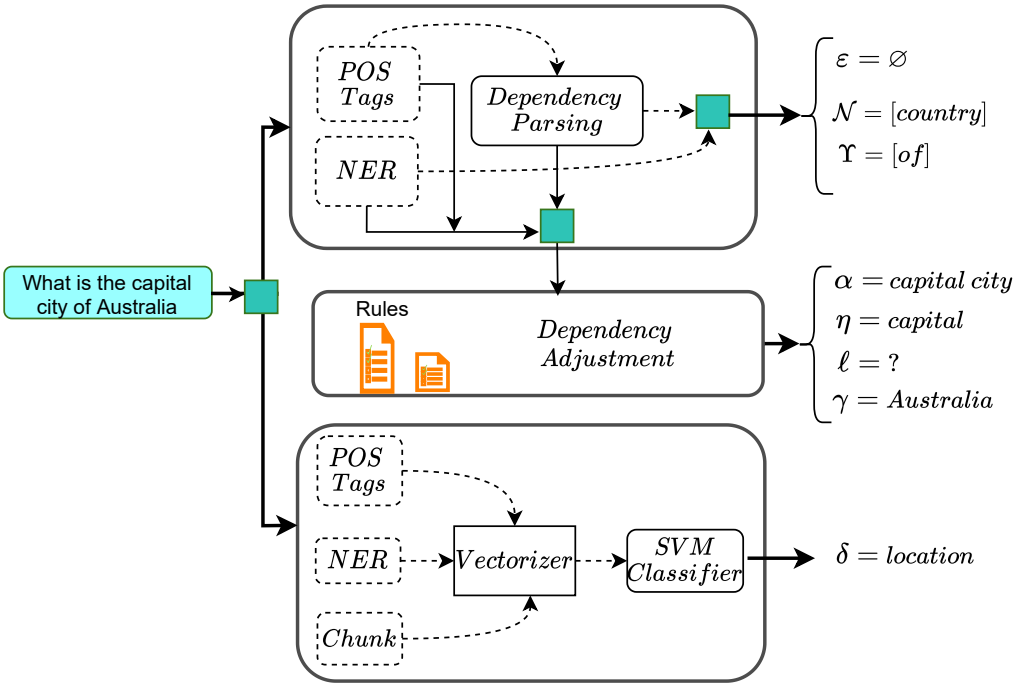


Figure 4.3: Generation of a Q-Rel: The Q-Rel partitions the question into an octuple of it's constituent tokens.

Dependency Adjustment

Rules have been used in several relation extraction tasks for either directly identifying relations [230] or for complementing machine learning algorithms. In this work, we apply rules in two ways namely, i) rules for reducing multi relation questions into constituent single relation questions for ease of processing and ii) for readjusting the relation word in the Q-Rel. To derive simple relations from multi relation questions, we first must partition our question into a simple question that would translate into `Q-Rels`. Based on the initial parse characteristics, we identify the following four elements of complex questions as opportunities for decomposition into the constituent simple questions. Three of these are primarily

inspired by the work of Reddy et al. [231] where they employ linguistic constructs to derive logical forms from dependency parses. Of relevance to our work is their interpretation of adjectival clauses, prepositional phrases and conjunctions. We add extra adjustment consideration based on possessive structures.

Only the relative clauses require recursive processing since the other three lend themselves directly into relations. An adjectival clause, also called relative clause [232, 233] is introduced by the relative pronouns who, whom, whose which, that, etc. Regardless of whether a relative clause is defining or non-defining, they form a separable independent section of a sentence. The relative clause attachment is then considered to be able to prepend the subject of the clause. Taking the question: *"Who was vice president under the president who approved the use of atomic weapons against Japan during World War II?"*, a relative clause begins after *"the president"*, we, therefore, can process this question by analyzing two different statements. i. *"Who was vice president under the president."* and ii. *"The president approved the use of atomic weapons against Japan during World War II?"*.

The first part has only one relation *"vice president"* while the second part of this question produces several relations due to the preposition rule discussed hereafter. All of these prepositions have the same attachment on the verb *"use"* as in *"use of"*, *"use during"*, *use against* which we resolve into one relation with α as *"use"*. Eventually, when we processed this part of the relation, it has no match on any relation in the KG. In this context, this information is contained as a description of an entity rather than a relation. The entity in this question is `dbr:Harry_S._Truman`

For questions with irregular forms such as the form of the verbs *"have"*, *"to be"* and *"to do"* as part-modifiers, the parsers could return these modifiers as the root of the question. We then apply an adjustment rule that seeks the main verb of the question, for example, the question: *"Which movies did Kurosawa direct?"*, the dependency tree returns the token *"did"* as the root. In contrast, the relation word sought is the word *"direct"*.

Prepositional phrase attachments denote a wide range of relations such as time, possession, containment and locality etc. All unique instances of prepositional phrase attachment are considered as instances of Q-Rel. For the question: *How many people live in the capital city of Australia?*, we then derive two Q-rels based on the two prepositions *in* and *of*. *live in(people,X)* and *capital of (X, Australia)*. We add extra complimentary words to the set $\mathcal{N}$ of none named entities according to the type of preposition. For example, the preposition *in* associated with a location or that has a dependency with the word *where* would introduce the two words *location* and *place* if they did not already exist in the set $\mathcal{N}$. Similarly, adjustments are made appropriately if the preposition is of time or positions etc. Also considered are the possessive constructs in which the object of the possession becomes the relation as seen in the question: *What was Brazil's lowest rank in the FIFA World Ranking?* where *ranking* forms α and *lowest* forms η in the *Q-Rel*. A gazetteer of country names and their derived forms is introduced to evaluate all Named entities of type *location*. For those that resolve to country names, we add the word *country* to the set of non-named entity nouns $\mathcal{N}$ as seen in figure 4.3. After producing the Q-Rel we maintain the associated annotations related to the POS sequence and the Bag of words features.

4.2 Similarity Matching

In this section, we take the *Q-Rel* from the *Q-Rel* extractor and match it with the properties in the *EPS* using a set of similarity measures as described below. Four of these similarity measures are applied on the Wordnet Taxonomy graph. The result of the combination of these measures is a value that indicates how similar the *Q-Rel* is to a given property. Every property is then associated with a similarity value which is then used to rank the properties. The result is a list of *top k* ranked property URLs. Figure 4.4

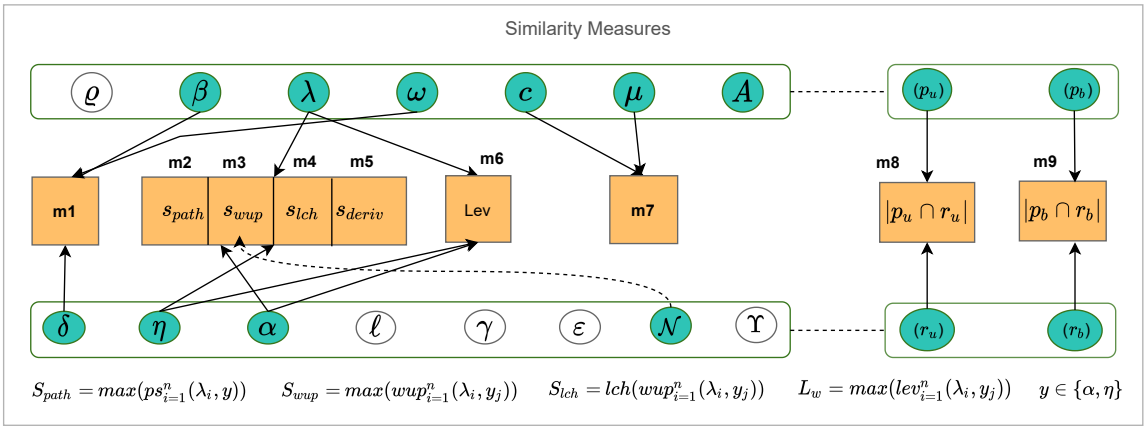


Figure 4.4: Similarity measures: S_{path} - Wordnet path similarity, S_{wup} - Wu-Palmer Similarity, S_{lch} - Leacock-Chodrow similarity, L_w - Levenstein weight obtained from the levestein similarity (Lev), p_u - Property unigrams, r_u - query relation unigrams, p_b - Property bigrams, r_b - query bigrams

indicates which elements from the two tuples are matched against each other. Each similarity measure is numbered in the picture with *m1* to **m9** labels and described as follows.

1. *Wordnet Path Similarity — ps (m1,m2)*:
   The path similarity is a score between 0 and 1 measured according to the behavior of the conceptual distance between two nodes in the taxonomy as factor of the number of edges separating them in the hierarchy [234]. Given two senses the shortest path ($len(r_1, r_2)$) that connects the senses in the is-a taxonomy determines the *ps*, where *ps=1* it implies the two senses are identical. Generally the path similarity (*ps*) is defined as:

   $$ps(r_1, r_2) = 2 * max\_depth - len(r_1, r_2) \tag{4.2.1}$$

   where *max_depth* is a constant representing the maximum depth of the Wordnet graph. In figure 4.4 the *ps* is used to obtain values of $m1$ and $m2$.

2. *Wu-Palmer Similarity (m3)* [235]:
   A measure that takes into consideration the Least Common Subsummer (LCS) of two senses. It is, by definition, the common ancestor deepest in the taxonomy but not necessarily closest to the two senses. If multiple LCS candidates exist, those whose shortest path to the root node is the longest will be selected. Generally, the longer path is chosen for the calculations when the LCS has multiple paths to the root.

3. *Leacock-Chodorow Similarity (m4)* [236]:
   A similarity score in relation to the shortest path connecting two senses and the maximum depth of the taxonomy in which the senses occur expressed as $-log(p/2d)$ where p is the shortest path length and d the taxonomy depth. Since the highest value of this measure is 3.6375, we normalize the value by expressing it as a ration of the $Max\_LCS = 3.6375$.

4. *Derivational_forms (m5)*:
   Derivational forms of a word are terms belonging to different syntactic categories but have the same root form and semantic relation. For example, the word *spouse* is a noun but has a derived

form *espouse* a verb which has a higher semantic relation to the verb *marry*. The other semantic measures miss this relationship. This measure is used to produce the measure *m5* in figure 4.4.

5. *Binarized Levenshtein Similarity (m6)*:
We define our Levenshtein similarity measure as:

$$lev_{sim}(a_1, a_2) = \frac{max(|a_1|, |a_2|) - lev(a_1, a_2)}{max(|a_1|, |a_2|)} \tag{4.2.2}$$

In our work, we employ the Levenshtein edit distance (*lev*) for word similarity on the lemmatized forms of the λ and α as well as the η. In cases where both elements contain values or consist of more than a word token each, we iteratively apply the Levenshtein distance. We represent this distance as either 1 or 0, depending on the nature of the two lemma forms and the extent of the dissimilarity. Take as an example α = "*discovered*" lemma form as "*discover*" against β = "*discoverer*"(*dbo.discoverer*) whose lemma form remains as *discoverer* using the Wordnet lemmatizer. The Levenshtein distance in this case is *2* giving the Levenshtein similarity $\frac{10-2}{10} = 0.8$. In this case, we require the similarity to be 1. Therefore the binarized Levenshtein similarity is given by:

$$lev(a_1, a_2) = \begin{cases} 1\,, \ if\ lev_{sim}(a_1, a_2) > 0.7\ \&\ a_1 \subseteq\supseteq a_2 \\ 0\,, \text{else} \end{cases} \tag{4.2.3}$$

6. *Instances count measure (m7)*:
We define a new measure related to the number of instances in the KG in which the property participates. Given the total number of instances for the property as c, the number of unique subjects in these instances as s and the number of unique objects as o. We first define a ratio $\mu = \frac{s}{o}$. We then use this ratio to penalise a value obtained from the total number of instances as follows:

$$\frac{c * n}{\sum_i c_i} * \mu \tag{4.2.4}$$

7. *Unigrams and Bigrams (m8,m9)*:
This measure obtains a normalised value related to the size of the intersection between two pairs of unigrams as well as bigrams from the question words and the KG properties. From the unigram set, we first remove stop words and require it to contain unique values. The bigrams are derived from the sequence of the POSs in the sentences. The intersection set length is then expressed as a fraction of the length of the question unigram or bigram, respectively.

***Overall aggregation of similarity measures*:**
Taking the similarity measures as a vector m such that m_i refers to the value of a similarity measure at position i in m we define the overall aggregated similarity score as a weighted sum measure:

$$Score_{sim} = wm^T = \sum_{i=0}^{n} w_i m_i \tag{4.2.5}$$

For this work, we assume the measures are all equally weighted but we observe that these weights can be easily learned via a Least Squares Optimisation method.

4.3 Experiments and Results

In this section, we detail our experiments and finding. First, we describe the setup followed by a discussion of the results and notes on possible impact.

4.3.1 Experiment Setup

For our evaluation, we used the QALD-5 dataset [227] which consists of over 400 questions together with the corresponding formal queries (SPARQL) to be applied against the DBpedia ontology [1, 237]. A total of 305 questions were considered, out of which 26 questions were out of scope and had no corresponding SPARQL query. Since for our work, we focus on providing an independent and reusable tool that identifies the URI of a property for pipelining in QA systems, we extract the properties within the SPARQL queries and annotate this against the input questions to form our evaluation dataset.

The 279 viable questions are grouped into three categories based on the number of properties required within the SPARQL queries. A total of 213 questions require only one single property to be matched within the query. A further 61 questions require two properties to be matched, and 5 of the question had three (3) properties. We evaluate against the properties within the SPARQL queries as opposed to the relations extracted from the natural language questions. Running on a 4-core CPU (at 1.7Ghz) with 8GB of memory, each question requires, on average 48 seconds to return an answer. The source code is available on GitHub[11] and a detailed description of practical implementation is online at the project wiki link: `https://github.com/mulangonando/ReMatch/wiki`.

Table 4.1: Performance Evaluation

#Properties in Question	*Total*	**Cumulative Frequency @Rank Position** @1	@2	@3	@4	@5	@10	**Precision** @1	@10	**Recall** @10	**F-Score** @10
1 Property	213	95	108	126	141	150	163	44.6.	53.8.	76.5.	63.2.
2 Properties	61	24 18	26 23	28 31	30 34	31 35	41 44	39.3. 29.5	45.3 41.4	67.2 72.1	54.1 52.6
3 Properties	5	0 2 3	1 2 3	1 2 3	1 3 3	1 3 3	1 3 3	0 40 60	20 45 60	20 60 60	20 51.4 60

4.3.2 Results and Impact

Table 4.1 illustrates our empirical results. For insightful understanding, we have grouped QALD questions into three categories. The first row of the table describes the categories of the questions, which contains only one relation (1 Property), for example, *Who is the wife of Barack Obama*. For such questions, our tool has a precision of 44.68 per cent when the correct result is at the first position in the final list of answers and 53.8 per cent as overall precision for top 10 properties. Recall and F-Measure are also considerably high for such questions, with values equal to 76.5% and 63.25% per cent, respectively.

For questions such as *How many people live in the capital city of Australia*, the expected properties from DBpedia are two: `populationTotal` and `captital`. For such questions (2 Properties), our tool provides overall precision of 45.3 percent for the relation occurred at first instance. In our example

[11] `https://github.com/mulangonando/ReMatch`

question, `populationTotal` represents the first relation of the input question. For questions such as *Which telecommunications organisations are located in Belgium*, which has three properties (3 Properties) namely `rdf:type`, `dbo:industry`, `dbo:location or dbp:locationcountry`, precision and recall values decrease drastically.

We analysed overall precision and recall values of the systems which took part in QALD-5 challenge. We can observe that if our tool were used as a component to identify relations within input questions, it would not decrease overall precision and recall values of many of the systems like SemGraphQA, YodaQA, QAnswer. This is because our tool has higher precision and recall value from many of these systems. Hence, while aiming for a component-based QA process, our tool would not negatively impact the QA system's overall performance for single and double relation questions. However, for the questions with three relations, our tool would negatively impact the QA system's overall performance.

4.4 Summary

In this section, we address our first research question (**RQ1**) by obtaining a unified representation for the relations in text and KGs. We presented an approach, and an independent, reusable tool, for matching natural language relations to KB properties for KG based Question Answering pipelines. This tool employs dependency parse characteristics with adjustment rules then carries out a match against KG properties enhanced with word lexicon Wordnet via a set of similarity measures. Our approach loses precision in cases where the targeted KG property has little textual augmentation. We also observe this drop when the question is too short to represent a considerable amount of information in the `Q-Rel` such as seen with the question: *"Give me all Cosmonauts."* The significant challenges in such scenarios for matching natural language relations to KB properties are the lack of tailored text corpora that can be used to train a learning algorithm. In our note for future work, we targeted to fine-tune the similarity measures by learning the weights through a well known least-squares optimisation approaches and evaluate the results against our results as a benchmark (Since this was the first piece of work in this direction). We also identified the use of embeddings (sentence, word and character level) both on the NLP and the KG side of the NLP-KG divide, coupled with Neural Networks based approaches for deep learning, as a promising avenue for obtaining better precision. A further extension is to determine if a similarity score obtained from the comparison of the Q-Rel embeddings with ESP embeddings would yield better results. In cases where we have a good recall value, but the desired property has not been ranked top of the results, an approach will be determined to better rank the final result set. Other elements still to be considered include experimentation within an existing QA pipeline and evaluating results based on actual question answers.

CHAPTER 5

Knowledge Context Encoding for End to end Entity Linking

Our contribution detailed in chapter 4, concentrates on recognising and linking relations. We describe the entry point to our overall research agenda, where we seek to identify and capture knowledge context from formal KGs for use in NLP models. We achieved this step by modelling text and KG relations into a comparable representation with term graph augmentation. However, we observed that entity linking could contribute richly to the overall understanding of relations in text. We observed from our review of state of the art in Chapter 3 concerning the use of knowledge context in entity linking approaches. Although entity linking is a long-running task in the NLP community, it remains an open research problem. Besides, we noted that most approaches concentrate on improving algorithmic power for efficient capturing of source context. This has resulted in very powerful models, especially with the advent of vector space learning and language modelling. Recently researchers have begun to leverage the power of knowledge context; for example, [24] uses entity aliases in a heuristics approach. Albert Cetoli [73] used one, and two hope KG triples to empower a BLSTM model, while researchers in [28] employ Wikipedia descriptions. However, the specific challenges that exist in end-to-end entity linking

Research Question 2 (RQ2)

How can KG context be effectively encoded in neural network architectures to improve Entity Disambiguation?

Contributing publications : [201] & [79]

All experiments for [201] were designed and carried out by the PhD candidate, who also participated in writing of the paper. Experiments for [79] were partly designed by the PhD candidate and carried out by the first author for Masters Thesis. The PhD candidate participated in writing of the paper.

To evaluate the effectiveness of the knowledge graph context in neural network models and the benefits of efficient encoding, we present the **Arjun** approach. This approach is the first reference implementation encoding attributes of entities from a KG in a Neural network aiming to overcome several specific challenges (see section 5.1). The Arjun Approach aims at providing a vector space representation of entity context specifically to be fed into an attentive neural network. We use Wikidata as the referent

Knowledge Base as well as the source of contextual information for entities. The challenge tackled, the Arjun Approach, and the evaluation results are presented in section 5.2. With the lesson learned from Arjun, we then seek to evaluate our findings in a state of the art language model-based architecture. Thus we present an implementation extension of the Arjun approach 5.3. This extended architecture depicts a modular approach that allows to extend the findings from section 5.2 for new state-of-the-art models. In this step, we apply the representation of KG context to Bidirectional Transformer model. However, we specifically position the candidate generation stage to allow flexibility. Finally, in Section 5.4, we provide a summary of the achieved results and conclude whether **RQ2** holds. This chapter is based on [201], and [79][1].

5.1 The Entity Linking problem

We formally define EL task as follows: given an input sequence of words W = $\{w_1, w_2, w_3, \ldots, w_n\}$, and a Set of entities denoted by $\mathcal{E}$ from a KG/KB. The EL task aligns the text into a subset of entities represented as $\Theta : W \rightarrow \mathcal{E}'$ where $\mathcal{E}' \subset \mathcal{E}$. We formulate the EL task as a three-step process in which the first step is the mention detection (MD). The MD is a function $\theta_1 : W \rightarrow \mathcal{M}$, where the set of mentions is denoted by $\mathcal{M} = (m_1, m_2, ..., m_k)$ $(k \leq n)$ and each mention m_x is a sequence of words starting from i to end position j: $m_x^{(i,j)} = (w_i, w_{i+1}, ..., w_j)$ $(0 < i, j \leq n)$. The next task is candidate generation where for each mention m_x a set of candidates $C(m_x)$= $\{e_1^x, ..., e_n^x | e_i^x \in \mathcal{E}\}$ is derived. Finally, the entity disambiguation (ED) task aims to map each mention $m_x \in \mathcal{M}$ to the most likely entity from its list of candidates. In our case, we model the ED task as a classification task and augment the input with extra signals as context. For every candidate entity $c_i \in C(m_x)$, the model estimates a probability p_i, thus the most likely entity is the one with the highest probability as $\gamma = \arg\max_{p_i}\{\mathcal{P}(p_i \mid m_x, c_i^x, W, C)\}$ where W and C are the input representations respectively for the given sentence (local context) and the context derived from KG/KB. As such the probability of score p_i is conditioned not only on m_x and c_i^x but also on W and C as contextual parameters.

To emphasize on the specialty of Wikidata KG and the challenges involved in EL on Wikidata, we step down our definition of a KG from section 2. Wikidata is an RDF knowledge graph that contains a set of triples $(s, p, o) \in \mathcal{R} \times \mathcal{P} \times (\mathcal{R} \cup \mathcal{L})$, where $\mathcal{R} = C \cup \mathcal{E} \cup \mathcal{P}$ is the union of all RDF resources. ($C, \mathcal{P}, \mathcal{E}$ are respectively a set of classes, properties, and entities), and L is the set of literals ($L \cap R = \emptyset$). An RDF knowledge graph represents a directed graph structure which has the formal definition as:

Knowledge Graph - Entity View

Definition 5.1.1 (Knowledge Graph) *A knowledge graph KG is a directed labelled graph $G(V, E)$, where $V = \mathcal{E} \uplus C \uplus L$ is a disjoint union of entities $\mathcal{E}$, classes C, and literal values L. The set of directed edges is denoted by $E = \mathcal{P}$, where $\mathcal{P}$ are properties connecting vertices. Please note that there is no outgoing edge from literal vertices.*

In this contribution, we target **end to end EL task**. The EL for us is defined as recognising the surface forms of entities in the text and then map them to the entities in the background KG. The EL task can be

[1] In this paper, my contributions include designing and implementing the involved re-engineering the Arjun approach to the fully modular design.

defined as follows:

Entity Linking

Definition 5.1.2 (Entity Linking) *Assume a given text is represented as a sequence of words* $w = (w_1, w_2, ..., w_N)$ *and the set of entities of a KG is represented by set* $\mathcal{E}$*. The EL task maps the text into a subset of entities denoted as* $\Theta : w \rightarrow \mathcal{E}'$ *where* $\mathcal{E}' \subset \mathcal{E}$*. Herein, the notion of Wididata entity refers to the representation of an entity based on the corresponding label because Wikidata might consider a variety of identifiers (called Q values) for the same label.*

The EL task can be divided into two individual sub-tasks. The first sub-task, Surface Form Extraction is recognising the surface forms of the entities in the text. This task is similar to Named Entities Recognition (NER). However, it disregards identifying the type of entities (e.g. person, place, date, etc.).

Mention Detection | Surface Form Extraction

Definition 5.1.3 (Surface form Extraction) *Let* $w = (w_1, w_2, ..., w_N)$ *be a text represented as a sequence of words. The surface form extraction is then a function* $\theta_1 : w \rightarrow \mathcal{S}$*, where the set of surface forms is denoted by* $\mathcal{S} = (s_1, s_2, ..., s_K)$ $(K \leq N)$ *and each surface form* s_x *is a sequence of words from start position i to end position j:* $s_x^{(i,j)} = (w_i, w_{i+1}, ..., w_j)$*.*

The second sub-task, Entity Disambiguation (ED), maps each surface form into a set of the most probable entities from the background KG.

Entity Disambiguation

Definition 5.1.4 (Entity Disambiguation) *Let* $\mathcal{S}$ *be the set of surface forms and* $\mathcal{E}$ *the set of entities of the background KG. Entity Disambiguation is a function* $\theta_2 : \mathcal{S} \rightarrow \mathbb{P}(\mathcal{E})$*, which assigns a set of entities to each surface form.*

Please note that a single surface form might be mapped into multiple, potentially suitable entities.

5.2 Arjun – An Approach for Efficiently Encoding KG Entity Context in Neural Networks

Entity linking (EL) over the web of data, often referred to as Named Entity Disambiguation (NED) or Entity Disambiguation is a long-standing field of research in various research communities such as information retrieval, natural language processing, semantic web, and databases since early approaches in 2003 [169]. In this contribution we concentrate ofn the two main subtasks of EL: *entity recognition* that is concerned with the identification of entity surface forms in the text, and *entity disambiguation* that aims at linking the surface forms with structures and semi-structured knowledge bases (e.g. Wikipedia), or structured knowledge graphs (e.g. DBpedia [1], Freebase [16] or Wikidata [17]).

Research Objectives, Approach and Contribution. The Wikidata KG is unique because the contents are collaboratively edited. As at April 2020; Wikidata contains 83,151,903 items and a total of over 1.2B edits since the project launch[2]. User-created entities add additional noise and non-standard formats since users do not follow a strict naming convention nor a standardized approach; for instance, there are 1788134 unique labels, in which each label matches with at least two different URIs. The previous approaches for EL [24, 34] on the textual content consider the well-established knowledge bases such as Wikipedia, Freebase, YAGO [202], and particularly DBpedia. Thereby, Wikidata as the core background KG along with its inherent challenges, has not been studied particularly for the task of EL.

Besides the vandalism and noise in underlying data of Wikidata, collaborative editing of its content adds several aliases of the entities and its description as entity properties (attributes). This enables Wikidata as a rich source of additional information which may be useful for EL challenges. Thus, in this work, we analyse the impact of additional context from Wikidata on Attentive Neural Networks (NN) for solving its entity linking challenges. We develop a novel approach called Arjun, first of its kind to recognise entities from the textual content and link them to equivalences from Wikidata KG. An important strength of Arjun is an ability to link non-Wikipedia entities of Wikidata by exploiting unique characteristics of the Wikidata itself (i.e. availability of entity aliases as explained in section 5.2.1). Please note; that he focus of this contribution is not to propose a black-box deep learning approach for entity linking using the latest deep learning models such as transformers or graph neural networks. In this section, we hypothesise that even though Wikidata is noisy and challenging, but its special property to provide aliases of entities can help an NN better understand the context of the potential entities. Since the concept of informing a neural network using contextual data from a KG is our proposed-solution in this work, we believe that traditional neural networks make it more transparent to understand the impact of KG context. Hence, our approach contributes to model attentive neural networks respecting the contextual content and trained on a sizable dataset. In particular, Arjun is a pipeline of two attentive neural networks, coupled as follows:

1. In the first step, Arjun utilizes a deep attentive neural network to identify the surface forms of entities within the text.

2. In the second step, Arjun uses a local KG to expand each surface form from the previous step to a list of potential Wikidata entity candidates. Unlike [37], Arjun does not use a pre-computed entity candidate list and search entity candidates among all the Wikidata entities.

3. Finally, the surface forms, coupled with potential Wikidata candidates, are fed into the second attentive neural network to disambiguate the Wikidata entities further.

[2] https://www.wikidata.org/wiki/Wikidata:Statistics

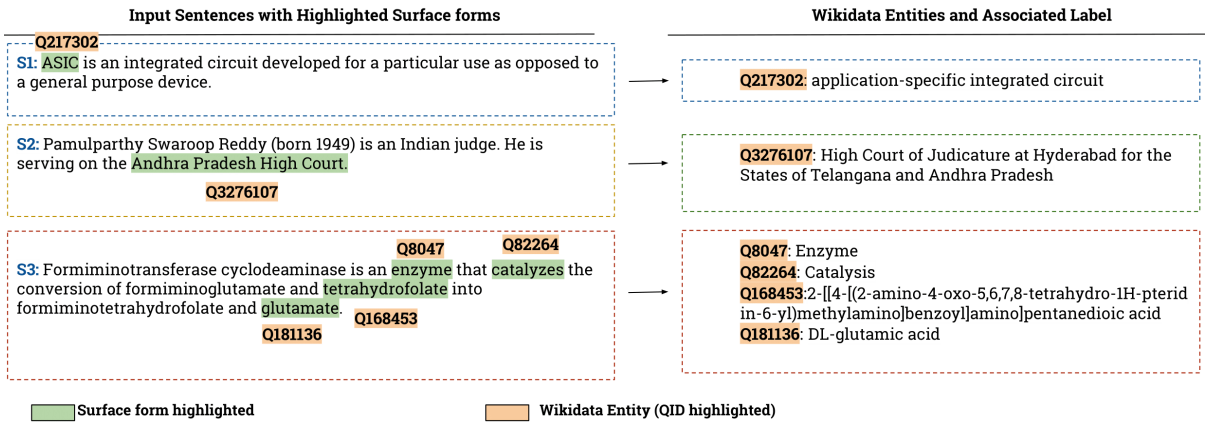


Figure 5.1: Wikidata Entity linking Challenges: Besides the challenge of capitalisation of surface forms and implicit nature of entities, Wikidata has several specific challenges, such as very long entity labels and user-created entities.

5.2.1 Problem and Motivating Examples

We motivate our work by highlighting some challenges associated with linking entities in the text to Wikidata. Wikidata is a community effort to collect and provide open structured encyclopedic data. The total number of entities described in Wikidata is over 54.1 million [17]. Wikidata entities are represented by unique IDs known as `QID` and `QIDs` are associated with entity labels. Figure 5.1 shows three sentences extracted from the dataset released by ElSahar et al. [72] which aligns 6.2 million Wikipedia sentences to associated Wikidata triples (<subject,predicate,object>).

In the first sentence S1, the surface form `ASIC` links to a Wikidata entity `wiki:Q217302` and the entity is implicit (i.e. no exact string match between surface form and entity label). However, ASIC is also known as 'Application Specific Integrated Circuit' or Custom Chip. Therefore to disambiguate this entity, background information about the surface form will be useful. Please note, we will use this sentence as a running example, "Sentence S1". In the second sentence S2, the surface form `Andhra Pradesh High Court` links to `wiki:Q3276107` which has 14 words in the full entity label[3]. It is also important to note here that the surface form `Andhra Pradesh High Court` also contains two sub-surface forms `Andhra Pradesh` and `High Court` which are the entity labels of the two Wikidata entities `wiki:Q1159` and `wiki:Q671721`. An ideal entity linking tool first has to identify `Andhra Pradesh High Court` as a single surface form, then disambiguate the surface form to a long entity label. In Wikidata, entity labels and associated aliases can be long (e.g. `wiki:Q1156234`, `wiki:Q15885502`). In addition there are long erroneous entity labels and aliases, such as entity `wiki:Q44169790`[4] with 62 words in the label and entity `wiki:Q12766033` with 129 words in one alias. The presence of long multi-word entity labels is also specific to Wikidata and poses another challenge for entity linking. Furthermore, in sentence S3 illustrated in the Figure 5.1, the surface form `tetrahydrofolate` is linked to `wiki:Q168453`. The entity `wiki:Q168453` not only has a multi-word entity label and lowercase surface forms but also contains several numeric and special, non-alphanumeric ASCII characters. Such entities are not present in other public KGs. This phenomenon results because, unlike Wikidata, other KGs do not allow users to create new entities, and the entity extraction process depends on unique IRIs of Wikipedia pages, WordNet taxonomy, and GeoNames. A

[3] High Court of Judicature at Hyderabad for the States of Telangana and Andhra Pradesh
[4] https://www.wikidata.org/wiki/Q44169790

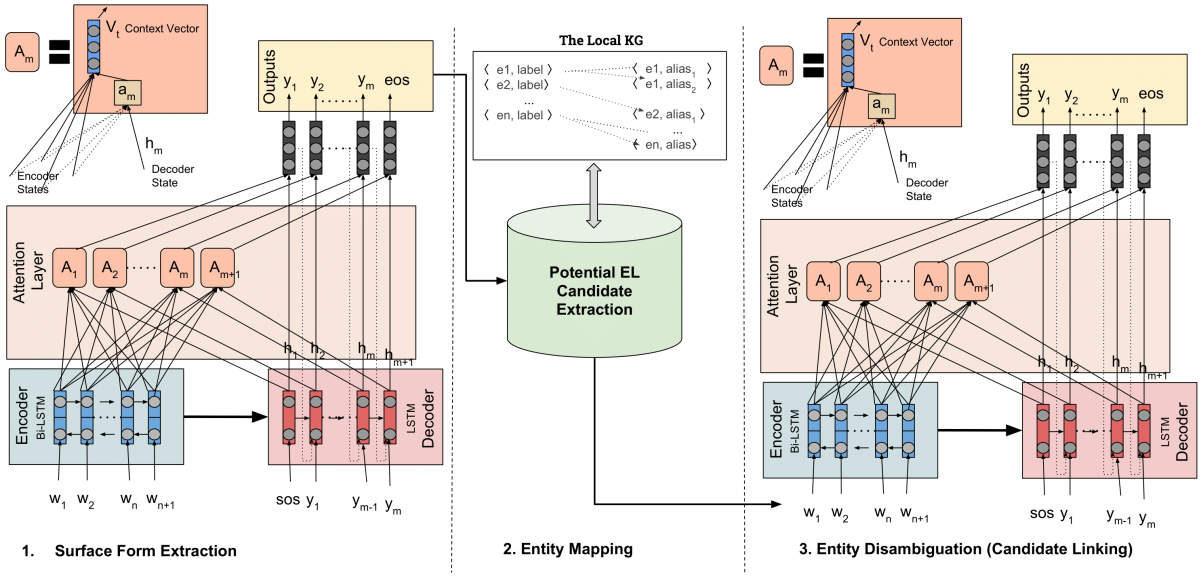


Figure 5.2: Proposed Approach Arjun: Arjun consists of three tasks. First task identifies the surface forms using an attentive neural network. Second task induces background knowledge from the Local KG and associate each surface form with potential entity candidates. Third task links the potential entity candidates to the correct entity labels.

large number of user-created entities poses specific challenges for entity linking. Therefore, it is evident that Wikidata exhibits some specific challenges to the entity linking problem in addition to generic entity linking challenges. Generic challenges such as the impact of capitalisation of surface forms and the implicit nature of entities have been tackled to a certain extent by approaches for entity linking over Wikipedia, and DBpedia [24].

5.2.2 Arjun: Attentive Encoding of KG Context

Arjun is illustrated in figure 5.2. Arjun performs three sub-tasks:

1. surface form extraction which identifies the surface forms of the entities,
2. entity mapping (or candidate generation) which maps the surface forms to a list of candidate entities from the Local KG,
3. entity disambiguation which selects the most appropriate candidate entity for each surface form.

We devise a context-aware approach based on attentive neural networks for tasks (1) and (3). We initially introduce our derived Local KG. Then we present the details of our approach for tasks (1), (2) and (3).

Local KG and Refinement Strategies. Arjun relies on Wikidata as the background knowledge graph. Wikidata consists of over 100 million triples in RDF format. Wikidata provides dumps of all the entities and associated aliases[5]. Although Wikidata has specific challenges for EL, models can utilise its unique characteristic to provide entity aliases in developing entity linking approaches. Since the

[5] https://dumps.wikimedia.org/wikidatawiki/entities/

training dataset is in English, we extracted all 38.6 million Wikidata entities with English labels and 4.8 million associated aliases from the dumps. We use entity labels and aliases as indexed documents in the Local KG, and a large portion of it is reused from Local KG built by Sakor et al. [24]. For example, the entity described in exemplary "Sentence S1" (cf. Figure 5.1), entity `wiki:Q217302` with label *application-specific integrated circuit* is enriched in the Local KG with its aliases: ASIC, Custom Chip, and Custom-Chip.

Model Architecture For task (1) and (3), our attentive neural model is inspired by the work of Luong et al. [120] and consists of an encoder, a decoder, and an attention layer. We don't claim that an extension of Luong's NN architecture used in this work as a novelty. Indeed we experiment with already established concepts of LSTM and attentive Neural Networks. We view our attempt of combining these NNs with *background contextual knowledge from a KG* as an interesting perspective for researchers within the community to solve Wikidata KG challenges and is our main novelty. The task (1) model is used to identify the surface forms of the entities in the input text. The similar attentive neural model used in task (3) selects the most appropriate candidate entity for each surface form (cf. Figure 5.2).

We extended Luong's model by using a Bidirectional Long Short-Term Memory (Bi-LSTM) model for the encoder and a one-directional LSTM model for the decoder. The input of the encoder is the source text sequence $w = (w_1, w_2, ..., w_n, .., w_N)$ where w_n is the n-th word at time step n and N is the length of the text. The encoder encodes the complete sequence, and the decoder unfolds this sequence into a target sequence $y = (y_1, y_2, .., y_m, ..., y_M)$ where y_m is the m-th word at time-step m and M is the length of the target sequence. In our assumption, each target sequence ends with an EOS (end of sequence) token. The N and M values can be considered as the last time steps of the source sequence w and the target sequence y, respectively.

Each word of the source sequence is projected to its vector representation acquired from an embedding model $\mathcal{R}^d$ with dimensionality d. The transformation of the input is represented in the matrix X as:

$$X = [x_1, x_2, ., x_n, ..., x_N] \tag{5.2.1}$$

where x_n is a vector with the size d and represents the low dimensional embedding of the word w_n.

The LSTM Layer: In our model, the encoder and the decoder consist of a single layer of Bi-LSTM and LSTM, respectively. Now we explain the LSTM layer.

We model the first layer of our network using an LSTM layer since it has been successfully applied to various NLP tasks. Each LSTM unit contains three gates (i.e., input i, forget f and output o), a hidden state h and a cell memory vector c. The forget gate is a sigmoid layer applied on the previous state h_{t-1} at time step t-1 and the input x_t at time step t to remember or forget its previous state (eq. 5.2.2).

$$f_t = \sigma(W^f[x_t, h_{t-1}] + b^f) \tag{5.2.2}$$

Please note that W is the weight matrix and b is the bias vector. The next step determines the update on the cell state. The input gate which is a sigmoid layer updates the internal value (eq. 5.2.3), and the output gates alter the cell states (eq. 5.2.4).

$$i_t = \sigma(W^i[x_t, h_{t-1}] + b^i) \tag{5.2.3}$$

$$o_t = \sigma(W^o[x_t, h_{t-1}] + b^o) \tag{5.2.4}$$

The next tanh layer computes the vector of a new candidate for the cell state $\tilde{C}_t$ (eq. 5.2.5). Then the old

state C_{t-1} is updated by the new cell state C_t via multiplying the old state with the forget gate and adding the candidate state to the input gate (eq. 5.2.6). The final output is a filtering on the input parts (eq. 5.2.4) and the cell state (eq. 5.2.7).

$$\tilde{C}_t = \tanh(W^C[x_t, h_{t-1}] + b^C) \tag{5.2.5}$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{5.2.6}$$

$$h_t = o_t \odot \tanh(C_t) \tag{5.2.7}$$

where the model learning parameters are weight matrices W^f, W^i, W^o, W^C and bias vectors b^f, b^i, b^o, b^C. The σ denotes the element-wise application of the sigmoid function, and $\odot$ denotes the element-wise multiplication of two vectors.

The Bi-LSTM of the encoder consists of two LSTM layers. The first layer takes an input sequence in the forward direction (1 to N), and the second layer takes the input in the backward direction (N to 1). We employ same equations 5.2.2, 5.2.3, 5.2.4, 5.2.5, 5.2.6, 5.2.7 for each LSTM Layer. The final encoder hidden state is produced by the sum of hidden states from both LSTM layers ($h_n = \overrightarrow{h_n} + \overleftarrow{h_n}$) at timestep n.

Attention and Decoder layer. The decoder layer takes SOS token (start of the sequence) vector, and the encoder final states (h_N and C_N) as the initial inputs to start decoding the source text sequence into a target text sequence. Here we differentiate between the encoder hidden state and decoder hidden state using the notations h_n at time step n and h_m time-step m, respectively. Below we explain how the decoder generates target text sequence words y_m one by one.

In the attention layer, we define attention weights as $a_m = [a_{m1}, a_{m2}, ..., a_{mN}]$ for a decoder state at time step m which has the size equals to the number of total time steps in the encoder side. The attention weights contain only scalar values which are calculated by comparing all encoder states h_n and decoder state h_m. To calculate the attention weight (a_{mn}) of an encoder state at time step n wrt. a decoder state at time step m, we use the following equation (5.2.8) [120].

$$a_{mn} = \frac{\exp(h_m \cdot h_n)}{\sum_{n'=1}^{N} \exp(h_m \cdot h_{n'})} \tag{5.2.8}$$

Where $(h_m \cdot h_n)$ denotes the dot product. The equation 5.2.9 computes the context vector V_m as weighted average over all the encoder hidden states (h_n) that captures the relevant encoder side information to help in predicting the current target sequence word y_m at time step m and can be defined as:

$$V_m = \sum_{n=1}^{N} a_{mn} h_n \tag{5.2.9}$$

We calculate Attention Vector ($\tilde{h}_m$) using the concatenation layer on the context vector V_m and decoder hidden state h_m for combining information from both the vectors. The equation 5.2.10 represent it mathematically (where tanh is an activation function same as describe in [120]).

$$\tilde{h}_m = \tanh\left(W_v[v_m; h_m]\right) \tag{5.2.10}$$

Finally, we apply softmax layer on the attention vector $\tilde{h}_m$ for predicting a word of a target text

sequence from the predefined vocabulary of the complete target text sequences.

$$p(y_m|y_{<m}, x) = \text{softmax}(W_s\tilde{h}_m) \tag{5.2.11}$$

Where W_s is weight matrix of softmax layer and p is probability. Please note that the decoder stops producing words once it encounters EOS (end of sequence) token or m is equal to M.

5.2.3 Entity Mapping Process

The local KG acts as a source of background knowledge. It is an indexed graph (created using the same methodology proposed by Sakor et al. [24] and reusing a large portion of the indexed graph built by the authors), where each entity label is extended with its aliases from Wikidata. Once Task 1 identifies surface forms in the input sentence, the entity mapping step (Task 2) takes each surface form and retrieves all the entities for which entity label(s) in the local KG matches the surface form. Next, the full list of the entity candidates is then passed into Step 3 of Arjun as input to predict (disambiguate) the best Wikidata entity labels.

Let us trace our approach for the sentence S1 of Figure 5.1 to understand the steps better. The sentence S1 *"ASIC is an integrated circuit developed for particular use as opposed to a general-purpose device"* is fed to the attentive neural model comprises of an encoder (Bi-LSTM), decoder (LSTM), and an attention layer as an input for the surface form extraction task. Thereby, the term `ASIC` is recognised as a surface form. Then, for the entity mapping task, we populate a Local KG to generate candidate entities associated with this surface form. We employ semantic search (reused from Falcon [24]) to identify entity candidate labels for `ASIC` which returns `Application Specific Integrated Circuit`. The last step of Arjun is entity disambiguation. In this step, the surface form `ASIC` along with `Application Specific Integrated Circuit` is fed into the encoder as the input sequence. Here, we utilise an identical attentive neural network used for the surface form extraction task. This attentive neural network decides the context of `ASIC` using extra information in the form of associated alias to correctly link to the Wikidata entity `application-specific integrated circuit` (Q217302).

5.2.4 Experimental Setup

Arju-Dataset

We rely on the recently released T-REx [72] dataset that contains 4.65 million Wikipedia extracts (documents) with 6.2 million sentences. These sentences are annotated by 11 million Wikidata triples. In total, over 4.6 million surface forms are linked in the text to 938,642 unique entities. T-REx is the only available dataset for Wikidata with such a large number of triple alignment. We are not aware of any other dataset explicitly released for Wikidata entity linking challenges. Please note that the popular entity linking datasets (e.g. CoNLL (YAGO) [147]) have linked entities either to Wikipedia, YAGO, Freebase or DBpedia. Work in [73, 185] attempt to develop approaches for EL over Wikidata and align (map using owl:sameAs) existing Wikipedia based dataset to Wikidata. However, our focus in this paper is to solve Wikidata specific challenges and these datasets do no embrace Wikidata specific challenges for entity linking. We divide the T-REx dataset into an 80:20 ration for training and testing.

Baseline

In this work, we pursue the following research question: "How well does the attentive neural network perform for entity linking task leveraging background knowledge particularly for a challenging KG such

as Wikidata?" To the best of our knowledge, it is a pioneering work for the task of entity linking on the Wikidata knowledge graph where it considers the inherent challenges (noisy nature, long entity labels, implicit entities). Therefore, we do not compare our approach to generic entity linking approaches which typically either do not use any background knowledge or employ the well-established knowledge graphs such as DBpedia, YAGO, Freebase. Our approach Arjun comprises all three tasks illustrated in figure 5.2. To elaborate the advantage of inducing additional context post NER step, we built a "baseline" which is an end to end neural model. The "baseline" in our case is the attentive neural network employed in Task 1 without any background knowledge (or can be seen as an end to end EL using an attentive neural network). In fact, in task (1) (cf. Figure 5.2), the baseline directly maps the text to a sequence of Wikidata entities without identifying surface form candidates. Hence, the baseline approach is the modified version of Arjun. With a given input sentence, the baseline implicitly identifies the surface forms and directly links them to Wikidata entities. Unlike Arjun, the baseline does not use any KG context for the expansion of the surface forms. We also compare Arjun with the recently released SOTA for Wikidata entity linking- OpenTapioca [185], which is an end to end EL approach. We are not aware of any other end to end EL tool/approach released for Wikidata.

Training Details

Implementation details We implemented all the models using the PyTorch framework. The local KG and the semantic search is implemented using Apache Lucene Core[6] and Elastic search [238]. The semantic search returns entity candidates with a score (higher is better). We reuse the implementation of Falcon local KG [24] for the same. After empirically observing the performance, we set the threshold score to 0.85 for selecting the potential entity candidates per surface form (i.e. the parameter is optimised on the test set). We reused pre-trained word embeddings from Glove [51] for the attention-based neural network. These embeddings have been pre-trained on Wikipedia 2014 and Gigaword 5[7]. We employ 300-dimensional Glove word vectors for the training and testing of Arjun. The models are trained and tested on two Nvidia GeForce GTX1080 Ti GPUs with 11GB size. Due to brevity, a detailed description of training details can be found in our public Github.

Dataset Preparation We experimented initially with higher text sequence lengths but resorted to 25 words due to GPU memory limitation. In total, we processed 983,257 sentences containing 3,133,778 instances of surface forms (not necessarily unique entities) which are linked to 85,628 individual Wikidata entities. From these 3,133,778 surface forms occurrences, approximately 62% do not have an exact match with a Wikidata entity label.

Results

Table 5.1 summarises the performance of Arjun compared to the baseline model and another NED approach. We observe nearly 8% improvement in the performance over baseline, and Arjun significantly outperforms another end to end EL tool OpentTapioca. Arjun and OpenTapioca generate entity candidates on the fly, i.e., out of Millions of Wikidata entities, the task here is to reach to top-1 entity. This contrasts with other end to end entity linking approaches such as [37], which rely on a pre-computed list of 30 entity candidates per surface form. This translates into extra complexity due to large search space for generating entity candidates in the case of Arjun. Our solution demonstrates a clear advantage of using KGs as background knowledge in conjunction with an attention neural network model. We now detail some success and failure cases of Arjun.

[6] https://lucene.apache.org/core/
[7] https://nlp.stanford.edu/projects/glove/

Success Cases of Arjun Arjun achieves 0.77 F-Score for the surface form extraction task. Arjun identifies the correct surface form for our exemplary sentence S1 (i.e. `ASIC`) and links it to the entity label *Application Specific Integrated Circuit* of `wiki:Q217302`. The baseline can not achieve the linking for this sentence. In the Local KG, the entity label of `wiki:Q217302` is enriched with aliases that also contain `ASIC`. This allows Arjun to provide the correct linking to the Wikidata entity containing the long label. Background knowledge induced in the attentive neural network also allows us to link several long entities correctly. For example, in the sentence "The treaty of London or London convention or similar may refer to," the gold standard links the surface form `London convention` with the label *Convention on the Prevention of Marine Pollution by Dumping of Wastes and Other Matter* (c.f. `wiki:Q1156234`). The entity label has 14 words, and Arjun provides correct linking. OpenTapioca, on the other hand, have a high recall(it has a high number of False Positives). However, the precision is relatively quite low. The limited performance of OpenTapioca was because it finds limitation in linking non-Wikipedia entities that constitute a major portion of the dataset. This demonstrates the strength of Arjun in also linking non-standard, noisy entities which are not part of Wikipedia.

Failure Cases of Arjun Despite the successful empirical demonstration of Arjun, we have a few types of failure cases. For example in the sentence: 'Two vessels have borne the name HMS Heureux, both of them captured from the French' has two gold standard entities (`Heureux` to *French ship Heureux* (`wiki:Q3134963`) and `French` to *French* (`wiki:Q150`)). Arjun links `Heureux` to *L'Heureux* (`wiki:Q56539239`). This issue is caused by the semantic search over the Local KG while searching for the potential candidates per surface form. In this case, L'Heureux is also returned as one of the potential entity candidates for the surface form `Heureux`. A similar problem has been observed in correctly mapping the surface form `Catalan` to `wiki:Q7026` (*Catalan Language*) where Arjun links `Catalan` to Catalan (`wiki:Q595266`). Another form of failure case is when Arjun identifies and links other entities which are not part of the gold standard. The sentence 'Tom Tailor is a German vertically integrated lifestyle clothing company headquartered in Hamburg' has two gold standard entity mappings: `vertically integrated` to *vertical integration* (`wiki:Q1571520` and `Hamburg` to *Hamburg* (`wiki:Q1055`). Arjun identifies *Tom* (`wiki:Q3354498`) and *Tailor* (`wiki:Q37457972`) as the extra entities and can not link `vertically integrated`. For brevity, a detailed analysis of the failure cases per entity type (very long label, noisy non-standard entity), performance loss due to semantic search can be found in our Github.

Limitations and Improvements for Arjun Arjun is the first step towards improving a deep learning model with additional contextual knowledge for EL task. Arjun can be enhanced in various directions considering current limitations. We list some of the immediate future extensions:

1. *Enhancing Neural Network Multiple layers:* Arjun currently has a Bi-LSTM and a single layer LSTM for the encoder and the decoder, respectively. It has been empirically observed in sequence

Table 5.1: Performance of Arjun compared to the Baseline.

Method	Precision	Recall	F-Score
baseline	0.664	0.662	0.663
OpenTapioca [185]	0.407	0.829	0.579
Arjun	0.714	0.712	0.713

to sequence models for machine translations that the models show significant improvements if stacked with multiple layers [239]. Therefore, with more computing resources, the neural network model used in Arjun can be enhanced with multiple layers.

2. *Alternative Models:* In this article, our focus is to empirically demonstrate how background knowledge can be used to improve an attentive neural network for entity linking. Several recent approaches [31, 32, 240] enhance the performance NER and can be used in our models for task (1) and task (3).

3. *Improving NER:* there is a room of improvement regarding surface form extraction where Arjun currently achieves an F-score of 0.77. The latest context-aware word embeddings [241] can be re-used in Arjun or completely replacing NER part with latest language models such as BERT [32].

4. *Replacing Semantic Search:* Another possibility of improvement is in the second step of our approach (i.e., inducing background knowledge). Currently, we rely on very trivial semantic search (same as [24]) over the Local KG to extract Wikidata entity candidates per surface form. Ganea et al. [196] developed a novel method to embed entities and words in a common vector space to provide a context in an attention neural network model for entity linking. This approach could potentially replace semantic search. Classification is seen as one of the most reasonable and preferred ways to prevent out of scope entity labels [37]. On the contrary, Sakor et al. [24] illustrated that expanding the surface forms the way we did, works pretty well for short text. We hypothesised that it should also work for Arjun, which is not completely true if we see our empirical results. Hence, in this paper, we do not claim that every step we took was the best, but after our empirical study, we demonstrate that the candidate expansion by Sakor et al. doesn't work well. However, it solves our purpose of inducing context in the NN, which is the main focus of the paper. It leads to an interesting discussion: what is the most efficient way to induce KG context in a NN, maybe the classification one?- one need to prove empirically, and we leave it for future work.

5. *Coverage restricted to Wikidata:* Effort can be made in the direction to develop a common EL approach targeting multiple knowledge graphs with standard and nonstandard entity formats.

5.3 Extended Arjun Approach for Bidirectional Transformers

In section 5.2 we described our initial attempt at encoding knowledge context in deep learning models. We employed a neural encoder-decoder model with attention mechanism for both the entity recognition (surface form extraction) as well as the disambiguation steps. We extend this concept to allow for a study of the candidate generation (CG) step of the EL process. This is achieved through a modular approach; however, we employ the state-of-the-art Bidirectional Transformer architecture BERT [32]. The problem remains largely the same as defined in section 5.2 above with the caveat that in this contribution, we concentrate on a model's ability to allow the study of the candidate generation step. In this section, we first introduce and motivate our hypothesis in the next section 5.3.1. We then detail the approach in section 5.3.2. This is then followed by a presentation of the results in section 5.3.3.

5.3.1 Idea

Entity Linking approaches are broadly categorised into three categories. The initial attempts [147, 242] solve MD and ED as independent sub-tasks of EL (i.e., a pipeline based system). However, these approaches exhibit a behaviour where errors propagate from MD to ED hence might downgrade the system's overall performance. The second category has emerged in an attempt to mitigate these errors, where researchers focused on jointly modelling MD and ED, emphasising the importance of the mutual dependency of the two sub-tasks [37]. These two EL approaches depend on an intermediate candidate generation step and rely on a pre-computed list of entity candidates. For example, [37] propose a joint MD and ED model and inherits the candidate list from [196]. The third approach combines the three sub-steps in a joint model and illustrates that each of those tasks is interdependent [39, 221].

The recent EL approaches focus on jointly modelling two or three subtasks [193]. Furthermore, the NLP research community has extensively used transformers in end-to-end models for entity linking (broscheit2019investigating, peters2019knowledge, and evry2020empirical). Nevertheless, these works report less performance than [37], a bi-LSTM based model. The observations regarding the limited performance of transformer-based models for the EL motivate our work. In this paper, our focus is to understand the bottlenecks in the entity linking process. We argue that the less studied task in literature, i.e., candidate generation, has an essential role in the EL models' performance, which has not been a focus in the recently proposed transformer-based EL models. We **hypothesise** that the transformer models, though trained on a large corpus, may require additional task-specific contexts. Furthermore, inducing the context at the entity disambiguation step may positively impact the overall performance, which has not been utilised in the state of the art methods due to monolithic implementations [37, 39, 200, 243]. Subsequently, we deviate from the joint modelling of two or three subtasks of the EL and revert to the methodology opted by earlier EL systems in 2011 [147], i.e. treat each sub-task independently. As such, we study the research question: **RQ**: *what is the impact of each sub-task (aka component) on the overall outcome of the transformer-based entity linking approach?* We propose an intuitive extension to the Arjun approach, comprising a modular architecture of two transformer models to solve MD and ED independently. In the first step, a BERT [32] model is employed to identify mentions of the entities in an input sentence. The second step involves expanding each mention with a list of KB entity candidates. Finally, the entity mention, sentence (local context), an entity candidate, and entity Wikipedia description (entity context) are fed as input sequences in the second BERT based model to predict the correct KB entity (cf. Figure 5.3). We train MD and ED steps independently during training, and while testing, we run this pipeline end-to-end for predicting the KB entity. The following are the novel features introduced in this approach:

- The core focus of the approach is to induce external context flexibly and candidate lists in a transformer-based model to improve the EL performance. The idea is to enhance independence from any particular candidate list and additional background context. We study four different configurations to demonstrate the impact of the candidate generation step and background knowledge (i.e. entity and sentential context) induced in the model. We achieves a new state of the art performance on several datasets: T-REx [72] for Wikidata; AIDA-B, MSBC, AQUAINT, and ACE2004 for Wikipedia [147, 244].
- This is the first approach empirically demonstrated to be transferable across KBs with completely different underlying structure, i.e., on semi-structured Wikipedia and fully structured Wikidata.

5.3.2 Transformer based Entity Linking pipeline

The extended Arjun architecture comprises of three main modules as illustrated in figure 5.3.

Mention Detection (MD)

We adapt the vanilla BERT [32] model for the task of entity mention detection in an unstructured text. For each input sentence, we append the special tokens [CLS] and [SEP] to the beginning and end of the sentence, respectively. This is then used as input to the model, which learns a representation of the tokens in the sentence. We then introduce a (logistic regression-based) classification layer on top of the BERT model to determine named entity tags for each token following the BIO format [245]. Our BERT† model is initialised using publicly available weights from the pre-trained BERT_{BASE} model and is fine-tuned to the specific dataset for detecting a mention m_i. Please note that BERT_{BASE} model is the latest approach that successfully outperformed in various NLP tasks, including MD. Thus, we reuse this model for the completion of our approach.

$$m_i = BERT^{\dagger}(w_i) \tag{5.3.1}$$

Candidate Generation (CG)

One of the critical focus of the transformer-based implementation of Arjun is to understand the bottleneck at the CG step. Hence, we reuse the DCA candidate list and propose a novel candidate list to understand the candidate generation impact on overall EL performance.

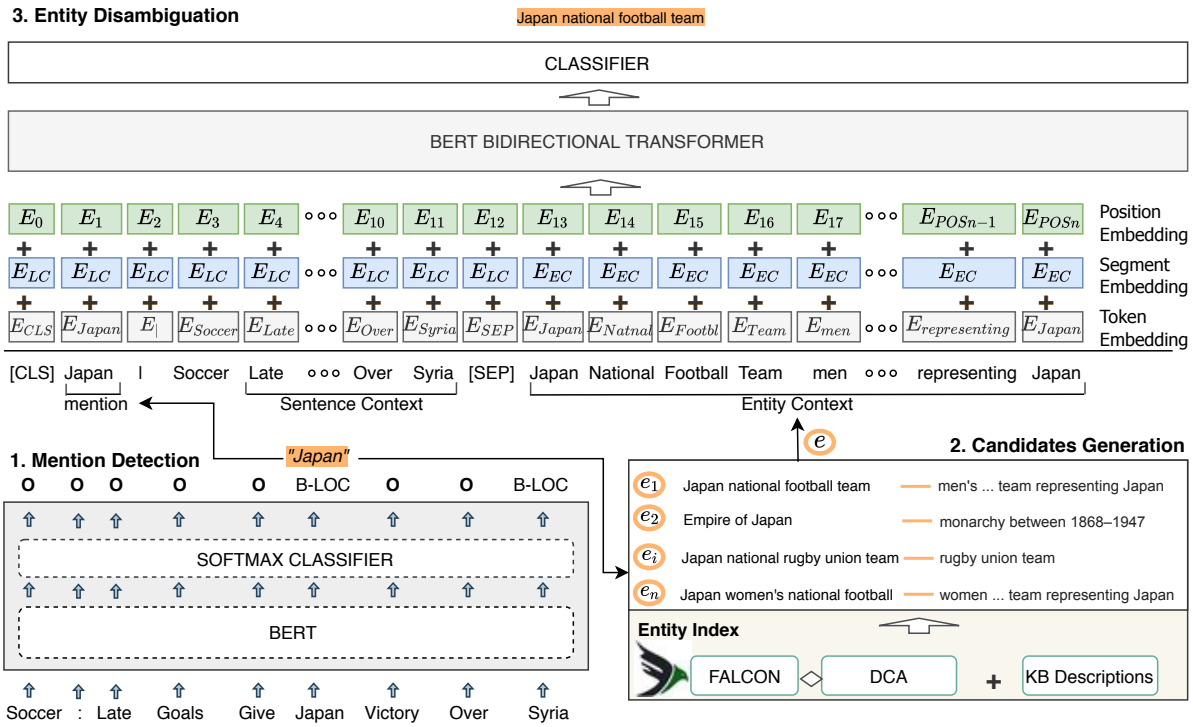


Figure 5.3: Arjun extension has three building blocks: i) BERT-based Mention Detection that identifies entity mentions in the text ii) Candidate Generation that retrieves a set of entities for the mention iii) Entity Disambiguation: employs BERT transformer model powered by background knowledge from KB and local sentential context.

DCA Candidates: [28] adapts the probabilistic entity-map $p(e|m)$ created by [196] to calculate the prior probabilities of candidate entities for a given mention. In the probabilistic entity-map, each entity mention has 30 potential entity candidates. Yang and colleagues also provide associated Wikipedia description of each entity. We *reuse* candidate set $C(m)$ provided by [28] and further consider associated Wikipedia entity descriptions.
Falcon Candidates: [24] created a local index of KG items from Wikidata entities expanded with entity aliases. For example, in Wikidata the entity Q33[8] has the label "Finland". Sakor and colleagues expanded the entity label with other aliases from Wikidata such as “Finlande", “Finnia", “Land of Thousand Lakes", “Suomi", and “Suomen tasavalta". We adopt this local KG index to generate entity candidates per entity mention in the employed datasets. The local KG has a querying mechanism using BM25† algorithm (cf. equation (5.3.2)) and ranked by the calculated score. We build a predefined candidate set using the top 30 Wikidata entity candidates in $C\_Falcon(m)$ for each entity mention. We enrich the candidates set obtained from Wikidata by the correspondence from Wikipedia. We also add the first paragraph of Wikipedia as entity descriptions (only if the Wikidata entity has a corresponding Wikipedia page) to the hyperlinks. By selecting two different candidate list, our idea is to understand the impact of the candidate generation step on end-to-end entity linking performance.

$$e_i = BM25^{\dagger}(m_i) \tag{5.3.2}$$

Entity Disambiguation (ED)

To use the power of the transformers, we propose “WikiBERT" to perform the ED task. In WikiBERT, our **novel methodological** contribution is the induction of local sentential context and global entity context at the ED step in a transformer model, which has not been used in the recent EL models. WikiBERT is derived from the vanilla BERT_{BASE} model and fine-tuned on the two EL datasets (CoNLL-AIDA and T-REx). We view the ED task as sequence classification task. The input to our model is a combination of two sequences. The first sequence S_1 concatenates the entity mention $m \in \mathcal{M}$ and sentence $\mathcal{W}$ where the sentence acts as a local context. The second sequence S_2 is a concatenation of entity candidate $e \in C(m)/C\_Falcon(m)$(obtained from Equation 5.3.2) and its corresponding Wikipedia description (entity context ct_i). The two sequences are paired together with special start and separator tokens: ([CLS] S_1 [SEP] S_2 [SEP]). The sequences are fed into the model which in turn learns the input representations according to the architecture of BERT [32]. Any given token (local context word, entity mention, or entity context words) is a summation of the three embeddings:

i. *Token embedding*: refers to the embedding of the corresponding token. We note here on specific tokens that comprise the input representations for our model more specialised than other fine-tuning tasks. The entity mention tokens appended at the beginning of S_1 and separated from the sentence context tokens by a single vertical token bar |; likewise, for the entity context sequence S_2, we prepend the entity title tokens from the KB before adding the descriptions.

ii. *Segment embedding*: each of the sequences receive a single representation such that the segment embedding for the local context E_{LC} refers to the representation for S_1 whereas E_{EC} is the representation of S_2

iii. *Position embedding*: represents the position of the token in an input sequence. A token appearing at the i-th position in the input sequence is represented with E_i

[8] https://www.wikidata.org/wiki/Q33

To train the model, we use the negative sampling approach similar to [246]. The candidate list is generated for each identified mention. The desired entity candidate item is labelled as one, and the rest of the incorrect candidate items (from the candidate list) are labelled as zero for a given mention. This process iterates over all the identified mentions using Equation 5.3.1.

The training process fine-tunes BERT using the contextual input from the sentence and Wikipedia resulting in the WikiBERT model (Equation (5.3.3)). The model predicts the relatedness of the two sequences by classifying them as either positive or negative.

$$e_i = WikiBERT(m_i, e_i, ct_i) \tag{5.3.3}$$

5.3.3 Experiments and Results

[72]. We adapt the subset of T-REx used by [201] for a fair evaluation setting. The dataset contains 983,257 sentences (786,605 in training and 196,652 in the test set) accommodating 3,133,778 instances of surface forms linked to 85,628 distinct Wikidata entities. T-REx does not have a separate validation set to fine-tune the hyperparameters. Therefore, we further divide the train set into a 90:10 ratio for training and validation. For EL over Wikipedia, we adapt the standard dataset CoNLL-AIDA proposed by [147] for the training. The dataset contains 18,448 linked mentions in 946 documents, a test set of 4,485 mentions in 231 documents, and a validation set of 4,791 mentions in 216 documents. For testing, we use AIDA-B (test) dataset from [147] and MSNBC, AQUAINT, ACE2004 datasets from [244].

5.3.4 Models for Comparison

Baselines over Wikidata

We now briefly explain Wikidata baselines.
1. OpenTapioca [185]: is a heuristic-based end-to-end approach that depends on topic similarity and mapping coherence for linking Wikidata entity in an input text.
2. Arjun [201]: is a pipeline of two attentive neural networks employed for MD and ED. Arjun is the SotA, and we take baseline values from Arjun's paper. For all our experiments (Wikidata and Wikipedia), we only consider peer-reviewed baselines published until 15.09.2020.

Baselines over Wikipedia

1. [147]: build a weighted graph of entity mentions and candidate entities. Then, the model computes a dense subgraph that predicts the best joint mention-entity mapping.
2. DBpedia Spotlight [36] proposes a probabilistic model and relies on the context of the text to link the entities.
3. KEA [7] employs a linguistic pipeline coupled with metadata generated from several Web sources. The candidates are ranked using a heuristic approach.
4. Babelfy [247] is a graph-based approach that uses loose identification of candidate meanings coupled with the densest subgraph heuristic to link the entities.
5. [242]: to solve entity linking, authors focus on mentions recognition and annotations pruning to propose a voting algorithm for entity candidates using PageRank.
6. [37] train MD and ED task jointly using word and character-level embeddings. The model reuses the candidate set from [196] and generates a global voting score to rank the entity candidates.
7. [243] induce multiple KBs into a large pre-trained BERT model with a knowledge attention mechanism.
8. [39] trains MD, CG, ED task jointly using a BERT-based model. Besides, it utilised an entity vocabulary

containing 700K most frequent entities in English Wikipedia.
9. [200] consider large scale pretraining from Wikipedia links as the context for a transformer model to predict KB entities.
In Wikipedia-based experiments, we report values from [200] and [37] for AIDA-B test set. On MSNBC (MSB), AQUAINT (AQ), and ACE2004 (ACE) test datasets, only [37], DBpedia Spotlight [36], KEA [7], and Babelfy [175] report the values and we compare against it.

Hyper-parameters	**Value**
Epochs	4
Batch size	8
Learning rate	$2e^{-5}$
Learning rate decay	linear
Adam β_1	0.9
Adam β_2	0.999
dropout	0.1
Loss Function	Cross-Entropy
Classifier	Softmax

Table 5.2: Hyper-parameters during fine-tuning.

5.3.5 Configurations

We configure our model by applying various candidate generation approaches detailed below.
Arjun-BERT-Wikidata: we train the model using the T-REx dataset and employ $C\_Falcon(m)$ candidate set. The ED model (WikiBERT) is fed with the sentential context but not with entity description as not all Wikidata entities have a corresponding Wikipedia entity.
Arjun-BERT-Wiki+FC: is trained on CoNLL-AIDA [147]. For CG step, we employ Falcon candidate set $C\_Falcon(m)$. Here, the ED model (WikiBERT) is only fed with the sentential context.
Arjun-BERT-Wiki+DCA: We train the MD and ED models on CoNLL-AIDA. The CG step involves DCA candidate set $C(m)$. During the ED step (WikiBERT), Wikipedia descriptions associated with each entity is fed along with sentential context.
Arjun-BERT: inherits **Arjun-BERT-Wiki+FC** but also, Wikipedia entity description is induced into the ED model (WikiBERT).

5.3.6 Metrics and Hyper-parameters

On Wikidata-based experiments, we employ standard metrics of accuracy i.e., precision (P), recall (R), and F-score (F), same as [201]. For Wikipedia-based datasets, we use Micro-F1 score in a strong matching setting [37]. The strong matching needs exactly predicting the gold mention (i.e. target entity mention) boundaries and its corresponding entity annotation in the KB. To compare the recalls of two CG approaches, we report the performance on gold recall. The gold recall is the percentage of entity mentions for which the candidate set contain the ground truth entity [248].
We have implemented all our models in PyTorch[9] and optimized using Adam [135]. We used the pre-trained BERT models from the Transformers library [249]. We ran all the experiments on a single

[9] https://pytorch.org/

GeForce GTX 1080 Ti GPU with 11GB size. Table 5.2 outlines the hyper-parameters used in the fine-tuning of both the datasets. We followed the standard settings suggested by [32].

5.3.7 Results

We study the following research question:*what is the impact of each sub-task (aka component) on the overall outcome of the transformer-based entity linking approach?* We further investigate a sub-research question: how do the external context and the candidate generation step impact the overall performance of Arjun-BERT? Every experiment systematically studies the research questions in different settings.

Results on Wikidata dataset

Table 5.3 summarises the performance of extended Arjun-BERT on T-REx dataset. Arjun-BERT-Wikidata configuration outperforms the baselines. We dig deeper into our reported values. We observe that for the MD task, our F-score is 94.3 (compared to 77 F-score of Arjun [201]). However, the gold recall for the CG step is 81.2. We generate the entity candidates using an information retrieval approach (BM25† algorithm) to get the top 30 candidates based on the confidence score. The Wikidata KG is challenging, and many labels share the same name. It contributes to a large loss in the F-score for the CG step. For instance, the entity mention "National Highway" matches exactly with four Wikidata ID labels while 2,055 other entities contain the full mention in their labels. Please note that we did not perform retraining of [37] (SOTA on Wikipedia EL) on the T-REx dataset since we determined that the model is tightly coupled and relies on pre-computed Wikipedia candidate list from [196].

Model	P	R	F
delpeuch2019opentapioca	40.7	**82.9**	57.9
mulang2020encoding	71.4	71.2	71.3
Arjun-BERT-Wikidata	**75.0**	76.0	**75.4**

Table 5.3: Comparison on T-REx test set for Wikidata EL. Best values in bold.

Ablation Study on Wikidata We study the impact of local context on the performance of Arjun-BERT. Therefore, we exclude the sentence as input in the ED step at training and testing time. Hence, the inputs to the ED model are only entity mention and the entity candidates gained from the CG step. We observe that the performance drops when the local sentential context is not fed (cf. Table 5.4). It justifies our choice to feed the model by the sentence during the ED task.

Model	P	R	F
Arjun-BERT-Wikidata	**75.0**	**76.0**	**75.4**
Arjun-BERT-Wikidata (WLC†)	72.0	73.5	72.7

Table 5.4: The ablation study on T-REx test set for Wikidata EL. Best values in bold. WLC† denotes model without local context. When the local sentential context is excluded from ED, the performance drops.

Results on Wikipedia datasets

Table 5.5 reports the performance of our extended configurations on AIDA-B test set. The first configuration is "Arjun-BERT-Wiki+ FC", in which MD and ED models are trained using CoNLL-AIDA. We notice a clear jump in the performance. We then replaced the Falcon candidate list $C\_Falcon(m)$ with DCA candidates $C(m)$ resulting in "Arjun-BERT-Wiki+ DCA". In DCA candidates, the description of entities is attached. The performance is increased when additional background knowledge as an entity description is fed. Our next configuration is Arjun-BERT, where we attached Wikipedia entity descriptions in Falcon candidate list $C\_Falcon(m)$ (as a modification of "Arjun-BERT-Wiki+ FC"). This setting outperforms all the existing baselines and previous Ajun-BERT configurations. Our experiments illustrate the impact of CG step and background knowledge on end-to-end EL performance. The improvement we observe continues to the other three test datasets where the jump is significantly higher compared to the baselines (cf. Table 5.6). Reported values in table 5.6 also approves the transferability of this approach when we apply cross-domain experiments.

Model	**Micro F1**
Hoffart et.al.,2011 [147]	72.8
DBpedia spotlight [36]	57.8
Steinmetz et.al. [7]	42.3
Moro et.al. [175]	48.5
Piccinno et.al. [242]	73.0
Kolitsas et.al. [37]	<u>82.4</u>
Peters et.al. [243]	73.7
Broscheit et.al. [39]	79.3
Evry wt.al. [200]	76.7
Arjun-BERT-Wiki+ FC	75.1
Arjun-BERT-Wiki+ DCA	77.5
Arjun-BERT-Full	**83.1**

Table 5.5: Comparison on *AIDA-B*. Best value in bold and previous SOTA value is underlined.

Model	**MSB**	**AQ**	**ACE**
DBpedia spotlight [36]	40.6	<u>45.2</u>	60.5
Steinmetz et.al. [7]	30.9	35.9	40.3
Moro et.al. [247]	39.7	35.8	17.8
Kolitsas et.al. [37]	<u>72.4</u>	40.4	<u>68.3</u>
Arjun-BERT-Wiki+ FC	77.8	70.0	85.7
Arjun-BERT-Wiki+ DCA	78.3	75.9	71.3
Arjun-BERT	**83.4**	**76.8**	**86.8**

Table 5.6: The micro F1 scores are listed from the comparative study over three datasets (out of domain). The model is trained over CoNLL-AIDA dataset. Best value in bold and previous SOTA value is underlined.

Ablation Study on Wikipedia

We conducted three ablation studies to understand the behaviour of these configurations over Wikipedia datasets. The first study is to calculate the Gold recall values for various datasets. Full Arjun-BERT uses the candidates from $C\_Falcon(m)$ candidate set for each entity mention. While generating the candidate set from local KG of [24] we observe a drop in the Gold recall as reported in table 5.7. CG plays a crucial role in trading off precision and recall. We conclude that more robust CG approaches likely to impact overall performance. The second ablation study is about to calculate the performance of our configurations for ED step, i.e., running WikiBERT in isolation. Here, we assume that all entities are truly recognised; thus, our focus of the study is the ED model. We report the impact of various candidate generation approaches on the ED model in table 5.8. The significant jump in the performance from "Arjun-BERT-Wiki+FC Vs full Arjun-BERT" contributes to the additional background knowledge provided in full Arjun-BERT as entity candidate descriptions. The third ablation study tests the impact of sentential context fed into two configurations on a Wikipedia dataset. Table 5.9 reports the achieved performance after excluding sentence as the additional context. Obviously, the performance decreases. The model shows similar behaviour on T-REx in table 5.4. These observations confirm our hypothesis as the ED model is enhanced using additional contexts.

Model	**AIDA-B**	**MSB**	**AQ**	**ACE**
Falcon Candidates	94.0	93.8	85.3	97.3
DCA Candidates	98.3	98.5	94.2	90.6

Table 5.7: Gold Recall for Candidate Generation techniques over Wikipedia test datasets.

Model	**Micro F1**
kolitsas2018end	83.8
Arjun-BERT-Wiki+ FC	78.4
Arjun-BERT-Wiki+ DCA	79.1
Full Arjun-BERT	**85.7**

Table 5.8: Comparison on *AIDA-B* for ED. Best score in bold and previous SOTA value is underlined.

Model	**Micro F1**
CHOLAN-Wiki+ DCA	77.5
CHOLAN-Wiki+ DCA (WLC†)	71.2
CHOLAN	**83.1**
CHOLAN (WLC†)	79.6

Table 5.9: Ablation study on *AIDA-B*. We observe that when local sentential context is removed from ED step, the performance drops. Best values in bold. WLC† denotes model without local context.

5.4 Summary

In this chapter, we first focused on introducing the limitations of EL on Wikidata in general, presented the novel approach Arjun in section 5.2, and outlined deficiencies of Arjun, which in particular will guide future work on this topic. In this work, we empirically illustrate that for a challenging KG like Wikidata, if a model is fused with additional context post-NER step, it improves entity linking performance. However, this work was our first attempt towards a longer research agenda. This leaves several directions for possible extension including: (i) extending towards joint entity and predicate linking and use latest language models for NER task, (ii) enriching the background KG to several interlinked KG from Linked Open Data (DBpedia, Freebase, YAGO), (iii) extending Arjun for the learning entities across languages (currently limited to English). In the last two years, the NLP research community has extensively tried transformer-based models for the EL task. However, the performance remained lower than the work by Kolitsas et.al. [37]. We therefore applied the Arjun idea with transformer models into the Arjun-BERT approach 5.3. In this variation, we combine the traditional software engineering principle of modular architecture with the context-induced transformers to effectively solve the EL task. Our reason to deviate from an end-to-end architecture was to provide full flexibility to our system in terms of candidate generation list, underlying KG, and induction of the context at the ED step. We attribute the impressive outperformance of Arjun-BERT to the following reasons: 1) the modular architecture, which brings flexibility and interoperability as Arjun can treat each task independently. [37] reports that shifting towards joint modelling of MD and ED tasks helps mitigate error propagation from MD to ED. However, the performance of BERT_{BASE} for the MD task is significantly high (92.3 on AIDA-B and 94.3 F1-score on T-REX calculated by us), remarkably reducing the errors in MD. This capability is leveraged in the MD subtask, placing more focus on CG and ED tasks. 2) The flexibility in architecture further permits us to induce sentence and entity descriptions as additional contexts. Furthermore, using the candidate list in a plug and play manner has resulted in a significant performance increase. In earlier transformer approaches, the implementation is monolithic, and the context is not utilised. The results achieved by our encoding of knowledge graph context: entity aliases for the original Arjun approach in section 5.2, and entity descriptions for the extended implementation in section 5.3 validate the second research question (**RQ2**).

CHAPTER 6

Generalising Knowledge Context

In the last chapter, we have presented our approaches for capturing knowledge context from KG (using Wikidata as our referent KG). However both the attentive encoder based Arjun approach 5.2 and the bidirection Transformer based implementations 5.3 employ only one or two attributes of entity from the KG namely: entity aliases, and entity descriptions respectively. We distinguish in chapter 3 that several approaches have been introduced for NER and NED tasks. Similarly relation extraction is a long standing task in NLP although the relation linking challenges addressed in chapter 4 are unique. To address these challenges in RL, we employed the Wordnet term graph to assist unify KG and text representations. Moreover we now see approaches such as [35] starting to adopt entity descriptions from KGs to assist in relation linking. Research also shows that Entity Linking (EL) tools have started to appreciate the importance of KG context as background knowledge to power these models. For instance the work in [34] uses entity type characteristics to enhance their models. Likewise, researchers in [28] employ Wikipedia entity descriptions to enhance an attentive neural network for entity disambiguation.

There remains a huge range of unexplored information available about entities that can be obtained from KGs. The intuition of providing semantic input obtained from the structure and representation of entities and relations in a KG is to provide special features specific to the context. Recently, the emergence of pre-trained transformer models as state-of-the-art approaches has interrupted NLP downstream tasks, and most research efforts now tend to follow this direction, for example research in KG completion [250]. The last two years has seen models that learn contextual information from text to improve the performance of several NLP tasks. These models, albeit powerful, still require specialised knowledge in specific scenarios. In this chapter, we take a deeper look into the usefulness of several aspects of contextual information found in a KG.

Research Question 3 (RQ3)

Can the effect of knowledge context be generalised for neural entity linking models?

Contributing publications : Mulang' et.al. [10, 251].

All experiments for both publications [10] and [251]were designed and carried out by the PhD candidate, who also participated in writing of the paper.

We present a detailed evaluation of knowledge context on entity disambiguation models with a view

to observe the value of knowledge context, as well as the extent of generalisation across models. To attain the overall research question targeted in this chapter (**RQ3**) , we investigate three sub research questions: **RQ3-a -** *Impact of knowledge context on transformer models:* How does applying KG context impact the performance of transformers on NED over Wikidata? **RQ3-b -** *Generalisation across models :* How can context be generalised across different knowledge bases (in our case, we consider Wikidata KG and Wikipedia KB). **RQ3-c -** *Relevance of different forms and configurations of knowledge context on NED models:* What is the performance of different forms and configurations of KG context as extra signals for NED ?. The remainder of the chapter is structured as follows. Section 6.1 describes our first set of experiments that studied impact of context on Entity Disambiguation (ED) and how such context information can be generalised across different approaches. Section 6.2 follows with a detailed extension of the evaluation to unearth insights about relevance of different forms of knowledge context. We illustrate the behaviour of several models under different forms of context. We summarise the chapter in section 6.3.

6.1 Evaluating Impact of Knowledge Context on Entity Disambiguation Models

In this contribution, we argue that sufficient context derived from a knowledge graph (KG) provides enough signals to inform pretrained transformer models and improve their performance for named entity disambiguation (NED) on Wikidata KG. We further hypothesise that our proposed KG context can be standardised for Wikipedia, and we evaluate the impact of KG context on the state of the art NED model for the Wikipedia knowledge base. Our empirical results validate that the proposed KG context can be generalised (for Wikipedia), and providing KG context in transformer architectures considerably outperforms the existing baselines, including the vanilla transformer models.

6.1.1 Entity Disambiguation - A Subtask of Entity Linking

Named entity disambiguation (NED) aims to link mentions in text to ground truth entities in a given knowledge base [28]. Research on the learning of contextual data has advanced in two directions. On one hand, the powerful pre-trained transformer models [52, 53] have emerged as state-of-the-art for representing context within text and have seen burgeoning reuse through fine-tuning for several NLP tasks including NED. On the other hand, KGs are increasingly being seen as a source of additional knowledge for Neural Networks. For instance researchers in [252] released an embedding library for the Wikidata KG while the work by [250] introduced an extension of BERT (KBERT) in which KG triples are injected into the sentences as domain knowledge. Specific to the EL task, the work by [201] employs information from a locally derived KG to improve the performance of end-to-end EL using attention-based Neural Networks. The work in [73] fetches a significant amount (as high as 1500) of 2-hop KG triples and used Recurrent Neural Networks (RNN) to encode this information. For a long time, researchers focused on NED tasks over semi-structured knowledge repositories such as Wikipedia[1]. Wikidata [17] has recently attracted the community's attention as a rich source of knowledge, and new approaches have been developed for NED over Wikidata [73]. Wikidata is collaboratively edited and user-created entities add additional noise and vandalism in Wikidata [253] since users do not follow a strict naming convention; for instance, there are 17,88,134 labels in which each label matches with at least two different URIs. This creates additional challenges for Wikidata NED [73].

[1] https://www.wikipedia.org/

Let us consider the sentence from Wikidata-Disamb[73] dataset: *"the short highway in New South Wales and the Australian Capital Territory in Australia, it is part of Sydney-Canberra National Highway link"*. The entity surface form *National Highway* matches four(4) different entities in Wikidata that share the same entity label (i.e., "National Highway") while 2,055 other entities contain the whole mention in their labels. The correct entity `wikidata:Q1967298`[2] refers to `Highway System of Australia`, whereas `wikidata:Q1967342` refers to the `highway system in India`. Having these two entities as candidates may require extra information in addition to the surface form or the sentence context. The pretrained transformer [52, 53], have provided an avenue for encoding the context within text, albeit, in cases such as our example, we postulate that pure textual context may not be sufficient.

6.1.2 Approach: Knowledge Context in Pre-trained Transformers

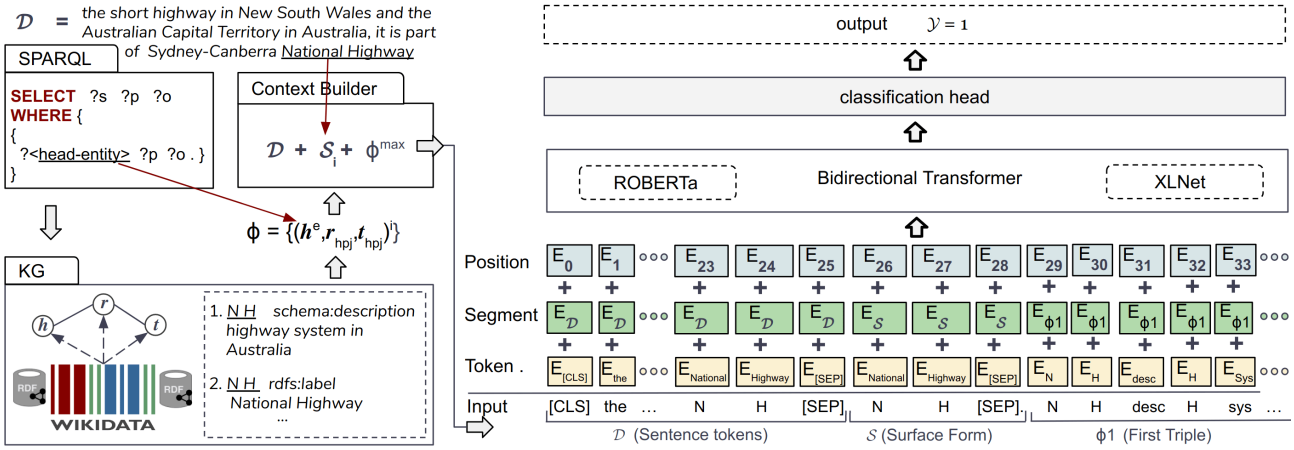


Figure 6.1: *Overall Approach:* Φ *refers to the ordered set of triples from the KG for a candidate entity while* $\Phi^{max} \subseteq \Phi$*, is the maximum number of triples that fits in the sequence length. For brevity:* $N \rightarrow$ *"National",* $H \rightarrow$ *"Highway",* $desc \rightarrow$ *"description"*

Figure 6.1 illustrates the overall approach. For the classification: $f(h(s, e'; \theta)) = y$ such that s, the mentioned surface form, and e', the candidate entity, are known. A set of contextual parameters θ is then provided to the model. By adding the original sentence as part of the input, we let the model learn source context. Such contextual information include the following data indicated in listing below. Our approach then models a set of information from the target KG in the form of KG triples Φ as context. The aim is to maximise both the true positives and true negatives such that, for every input, if $y = 1$ then the e is the ground truth entity of s in the KG. The classifier employs the binary cross-entropy loss.

```
Q1967298:   title <>  National Highway,
            description <> highway system in Australia,
            country <> Australia
Q61669822: title <> National Highway,
            description <> highway system in Taiwan,
            instance of <> highway system,
            country <> Taiwan,
            sub class of <> Highway system in Taiwan
```

[2] `wikidata:Q1967298` binds to https://www.wikidata.org/wiki/Q1967298

```
Q1967342:  title <> National Highway
           description <> network of highways that is managed and
           maintained by the Government of India modified,
           instance of <> highway system,
           country <> India,
           maintained by <> National Highways Authority of India,
           is a list of <> road
```

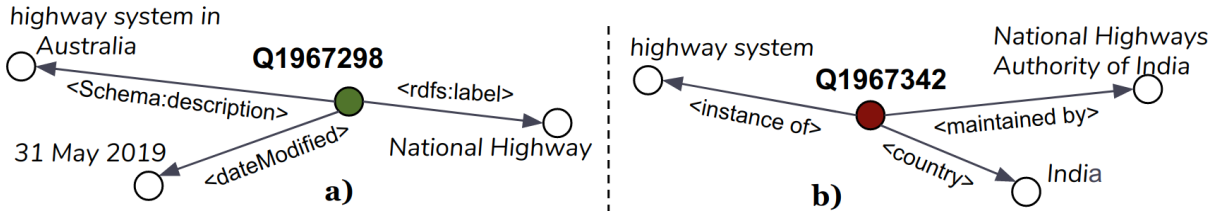


Figure 6.2: KG context: Top three 1-hop triples from Wikidata for the two entities with same label: National Highway.

Knowledge Graph Context: We use a SPARQL endpoint to fetch triples of the identified entity in the sentence. There are two sets of triple configurations considered in our experiments, depending on the hop counts from the head entity. The parameter Φ is therefore an ordered set of triples $(h^e, r_{hp}, t_{hp})^i$ such that h^e, the head (subject) of any triple is the candidate entity to be classified whereas $hp = 1|2$ is the hop count. The i refers to the position of the triple in the set and can range between 1 and over 1000. To formulate our input, we consider the natural language labels of the retrieved triples l_{h^e} , l_r , l_t. A triple is therefore verbalised into it's natural language form: "$l_{h^e}[whitespace]l_r[whitespace]l_t$". The sequence of these verbalised triples are appended to the original sentence and surface form delimited by the [SEP] token. Figure 6.1 shows how the context input is handled such that the Segment Embeddings for every triple is different and provides a unique signal to the tokens at the embeddings layer of the network. When the total number of triples is too many, we use the maximum sequence length to limit the input where the final context representation $\Phi^{max} \subseteq \Phi$.

The values of Φ^{max},for entity: Q1967298 in figure 6.2, is given as:

```
[National Highway description highway system in Australia
           [SEP] National Highway label National Highway
           [SEP] National Highway date modified 31 May 2019.
           [SEP]]
```

6.1.3 Evaluation and Results

Datasets: The first dataset is *Wikidata-Disamb*[73], which aligns Wiki-Disamb30 [170] to Wikidata entities, and adds closely matching entities as negative samples to every entity in the dataset. It consists of 200,000 Train and 20,000 Test samples. We also consider the *ISTEX* dataset introduced by [185], extracted from scientific publications and contains 1000 author-affiliation strings from research articles aligned to Wikidata. For Generalising the impact of KG context, we considered standard Wikipedia dataset: AIDA-CoNLL [147]. We aligned its Wikipedia entities to corresponding Wikidata mentions to

fetch the KG triples.

Baselines: We compare our results with three types of baselines. First is [73], which experimented with numerous configurations of KG context on Long Short Term Memory (LSTM) networks and reported an ablation of these configurations. These models were augmented with a massive amount of 1&2-hop KG triples. We also run the model on the ISTEX dataset to enable performance comparison. We create a second set of baselines by employing the vanilla transformer models of RoBERTa and XLNet(i.e., transformers without extra context from the KG) on Wikidata-Disamb and ISTEX. We fine-tuned vanilla models on Wikidata-Disamb training set. For AIDA-CoNLL, we chose [28] as our underlying model which is the **peer reviewed** SOTA on this dataset. Authors used Wikipedia descriptions as a context for candidate entities, and we replaced this context with our proposed 1-hop KG triple context fetched from Wikidata triples of corresponding Wikipedia entities. We verbalised the fetched triples, as described in our approach.

Model	Prec	Recall	F1
LSTM + RNN-triplets [73]	90.10	92.00	91.10
LSTM+RNN-triplets+ATN[73]	90.20	93.00	91.60
RoBERTa - without KG context	**89.09**	**84.67**	**86.23**
XLNet-without KG context	89.32	87.62	88.46
Our Contextual models			
RoBERTa + 1-hop KG Context	91.48	**93.23**	**92.35**
RoBERTa + 2-hop KG Context	89.88	87.64	88.75
XLNet + 1-hop KG Context	91.55	93.14	92.34
XLNet + 2-hop KG Context	**91.93**	92.36	92.14

Table 6.1: Comparison of our model against the baselines on the *Wikidata-Disamb* dataset. Best Results in dark bold, Worst results in gray bold

Model Parameters: We chose two state of the art transformer architectures: RoBERTa [53], and XLNet [52] and fine-tune them using Wikidata-Disamb30 training set. We report P,R,F values following the baseline of Wikidata-Disamb and ISTEX dataset. For each vanilla Transformer architecture, we add a classification head. The maximum sequence length for the inputs in both models is fixed at 512 tokens, and we use this to limit the amount of KG context to feed. We publicly release code, datasets, training details, and results for reusability and reproducibility:*blind review*. On AIDA-CoNLL, We use open source implementation of [28] for feeding the KG context and report In-KB accuracy as prior work(s).

Vanilla Transformer models perform worse than RNN model with task specific context. However, when provided with sufficient context, performance of Transformer models increase

Model	Prec	Recall	F1
LSTM + RNN of triplets + ATN *[73]*	86.32	**96.38**	90.97
Our Models			
RoBERTa + 1-hop Triples	91.70	91.98	91.84
XLNet + 1-hop KG Context	**96.39**	89.11	**92.61**

Table 6.2: Our models against baseline on ISTEX dataset

Results and Discussion: Table 6.1 shows the results from evaluating our approach against the baselines on the Wikidata-Disamb30 and table 6.2 indicates the performance of the models on the ISTEX dataset. The results in table 6.2 obtained by running the same model trained on the Wikidata-Disamb30 dataset

(also for baseline) with context, but during testing, no more context is provided. Based on our results, we postulate that although the Transformer based language models are trained on huge corpus and possess context for the data, *they show limited performance even against the RNN model. This RNN model [73] uses GloVe embeddings* together with task-specific context (cf. Table 6.1). However, the transformer models outperform the baseline models when fed with our proposed KG context. For instance, (cf. Table 6.1), RoBERTa, with a 1-hop context, can correctly link extra 1127 sample sentences in the test set compared to its vanilla setting. These samples have 997 unique Wikidata IDs. These results also indicate that the transformer models achieve better precision as opposed to recall; this is clear in table 6.2. We can interpret it as follows: our model is more likely to classify an entity as the correct entity only when it is true (few false positives). For brevity, the detailed analysis of each experimental setup and corresponding data can be found in our Github.

Model	**In-KB. Acc.**
Yamada et al. (2016) [29]	91.50
Ganea&Hofmann (2017) [196]	92.22
Yang et al. (2018) [217]	93.0
Le&Titov (2018) [218]	93.07
DeepType (2018) [34]	94.88
Fang et al. (2019) [194]	94.3
Shahbazi et al. (2019) [254]	93.46
Le& Titov (2019) [9]	89.66
DCA-SL (2019)[28]	94.64
Chen et al (2020) [57]	93.54
DCA-SL + Triples(ours)	**94.94**

Table 6.3: Generalizability Study: Comparison of KG Context based model against baselines on the *AIDA-CONLL* dataset. Best value in bold and previous SOTA value is underline.

Concerning **RQ3-b**, our results indicate that including triples from higher hop counts either exhibit an inverse impact on the performance or have minimal effect on overall model behaviour (cf. Table 6.1 RoBERTa vs. XLNet 2-hop values). This signals that the further away we drift from the head entity, the noisier the signal provided by the context added. As such, we did not extend evaluation to higher triple hops. However, we can observe that XLNet shows a more stable behaviour in cases when the excess context is provided as it can preserve already learned information. It is in contrast to RoBERTa, which loses necessary signals in an attempt to learn from the extra context. The amount of data fed as the context in our models is minimal (up to 15 1-hop triples). In contrast, the best performing model from work in [73], was fed up to 1500 1+2-hop triples. Our best performance can then be attributed to the quality of textual context learned by the transformers as well as the optimal choice of KG-triples context.
Generalising KG Context: We induced 1-hop KG context in DCA-SL model [28] for candidate entities. The replacement of the unstructured Wikipedia description with structured KG triple context containing entity aliases, entity types, consolidated entity description, etc. has a positive impact on the performance. Our proposed change (DCA-SL + Triples) outperforms the baselines for Wikipedia named entity disambiguation(cf. Table 6.3). Please note, out of 207,544 total entities of AIDA-CoNLL datasets, 7591 entities have no corresponding Wikidata IDs. Even if we do not feed the KG context for 7591 entities, the performance increases. It further validates our second sub research question (**RQ3-b**), and we conclude KG triple context can be standardised for the NED task for Wikipedia.

6.2 Relevance of Different Forms of Knowledge Context

In the last section, we have presented the insights we observed from evaluating the overall generalisability of knowledge context. This was a scaling step from the approaches we presented on chapter 5 for capturing knowledge context from KG (using Wikidata as our referent KG). To remain consistent with the grand research direction of unearthing the power of knowledge context for the linking tasks, we have taken a journey that passes through aspects of KG context in several approaches. The attentive encoder based Arjun approach 5.2 uses entity aliases and concentrated on proper encoding of this information for an attentive neural network. This is similar to the extrapolated implementation described in section 5.3 that induced entity descriptions into bidirection Transformers. To evaluate the overall generalisation of context from KG, the first part of this chapter, section 6.1 extends the reach of knowledge context to encompass both entity attributes and 1-and-2-hop triples. Although the results in section 6.1 prove that knowledge context generally extend quality features that are inherently model independent for entity disambiguation, there remains a single further sub research question. This concerns whether knowledge context contained in a KG offer consistent performance across different underlying datasets and models. The intrinsic question is rooted on two perspectives namely : i) Whether the nature of the data under study has some influence on the form and type of knowledge context required to improve performance, and ii) a case for model stability in which we interrogate the ability of models to stay stable under different variations of data and the volume of knowledge context.

An elaboration of the forms and volume of information represented in KGs is given in section 2.1. Such contextual data can be captured both from the structure of the KG and the literal data types. Chapter 3 discusses several approaches that have been introduced for NER and NED tasks and indicates increasing progress towards use of knowledge context in linking models. KGs are mainly rich knowledge repositories containing semantic information that define the represented entities and their relations. This provides a huge avenue to access and infer knowledge of various forms.

6.2.1 Richness of Knowledge Graphs

For expressive semantics, the KGs have exhibited different representation approaches. On the one side, KGs such as DBpedia and YAGO employ the Resource Description Framework (RDF) triples model in which an edge between two entities can exhibit only one relation. There is often a need to express multiple values of any given relation concerning an entity which is achieved through the RDF-Reification [255]. On the other hand, Wikidata utilises the Hyper-Relational [203] KG representation, in which a relation can be represented using several attributes. This is made possible via the Wikidata Statements, whereby items whose properties naturally pose multiple values (e.g., heads of states of a country at different times, or the head coach of a football club) are acceptable to portray each of these multiple values. The ability to express relations as statements allows Wikidata to accept claims from the crowd, and thus avail both challenges and opportunities. Figure 6.3 shows an example of representation with two entities from Wikidata KG and a subset of their relations. We can observe the depth of information represented to form contextual signals to Artificial Intelligence (AI) algorithms and methods for several tasks. The entity Q974 with label: *"Democratic Republic of the Congo"*, has a multivalued property *"also known as / alias"* with over 10 different forms including the former name *"Zaire"*, and the short form *"DRC"*, likewise the relation *"head of state"* links to more than 2 other named entities (Joseph Kabila, Félix Tshisekedi). The concept "Human" and the entity "Hewa Bora", are in the 2-hop from the head entity Q974. Given this amount of readily available information, KGs are recently utilised to provide additional contextual signals to the deep learning models. For instance, researchers in Ji et al. [158], RESIDE [35], and RECON [22], and Wang et al. [256] utilise varying KG information (entity type,

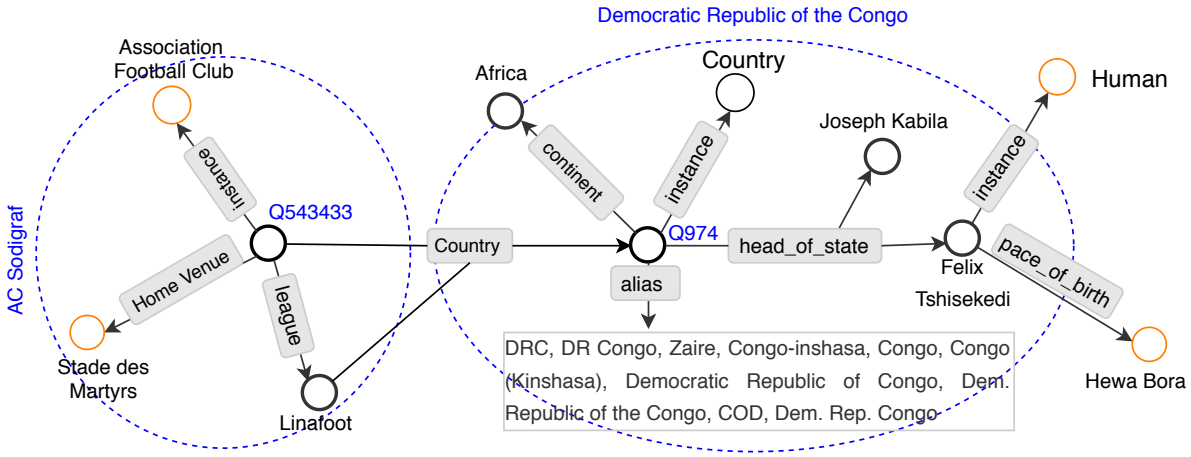


Figure 6.3: KG Entity representation : indicating 1-hop, and 2-hop triples - A set of readily available information from Wikidata KG concerning the mentioned named entities in our example text.

triples, etc.) to improve deep learning models aiming for various tasks such as entity linking, relation extraction, and recommendation system.

Entity Disambiguation (NED) is the last step in the Entity Linking (EL) task. EL majorly consists of three stages: mention detection, candidate generation, and named entity disambiguation (NED). A mention refers to a single sequence of tokens that denote an entity in the text. The candidate generation process seeks to identify a set of entities in the KG that probably refer to the given mention. The NED step scores the candidates against the given mention to select the best match. Systems that perform end-to-end EL, e.g., [37], train a single model that learns to provide the final entity scores from the text (performing all the steps in a single learner). In another direction, these tasks have been independently studied and evaluated, see for example Named Entity Recognition (NER) [201, 257, 258], and Named Entity Disambiguation (NED) [73, 259, 260]). As KG contexts are increasingly becoming popular to augment deep learning models, we examine the KG context's effect on entity disambiguation models in this paper. In our earlier work, we introduced Arjun [201]. Arjun focuses on inducing Wikidata entity aliases as the context in an attentive neural network for entity linking. However, one of Arjun's findings was establishing that context derived from the KG can improve deep neural network task-specific performance. However, Arjun limits the KG context to entity aliases. Our hypothesis in this paper is to understand which semantically encoded information of an entity has the most positive impact on the NED task. Let us take a look at the following sample sentence obtained from the CONNL-AIDA test-B dataset.

> *Result of the second leg of the African Cup Winners Cup final at the National stadium on Friday: Arab Contractors - Egypt 4 Sodigraf Zaire 0, halftime 2:0 Scorers: Aly Ashour 7', 56'(penalty), Mohamed Ouda 24' 73'. Contractors won 4-0 on aggregate*

The mentions *Sodigraf* and *Zaire* refer to the entities Q543433 with label *AC Sodigraf* and the African country Q974 *Democratic Republic of the Congo* respectively. These entities fall under the category of long-tail entities since they exhibit low out-degree and in-degree (edges/relations connecting from and to the entities) in the knowledge graph, which indicates that they are either emergent entities or scarcely used in the Web. Except for the single mention *"Zaire"*, the sentence exposes very little about

the referent entity. In the KG, this entity's label has no syntactic similarity relation to the mention in the text. Therefore, to help in disambiguation, we need an approach to encode different KG context that can provide semantic similarity, such as the fact that Q974 contains "Zaire" in the aliases which are in a 1-hop triple from the entity, and the relationship "content" connecting with entity Q15 "Africa." Such semantic information in the KG can be leveraged in the disambiguation process.

Given that the KG represents several attributes of an entity and the connections with other entities, research has already shown the power of using such information for underlying deep learning models. [10, 73] demonstrate that adding 1 and 2 hop triples from the KG increases the performance for the NED task. However, there is no indication of which of these forms of context impacts which kinds of datasets. In this work, we seek to empirically determine which semantically encoded information about an entity positively impacts NED. Further, we also desire to understand if the KG context shows homogeneous behaviour across different neural models and the impact across other underlying datasets. For the final sub resuch question - RQ3-c, we address the following aspects:

- Which forms of KG context provide sufficient signal to improve the performance of NED tasks: in the KG, there are several forms of information about entities such as the entity attributes (label, description, aliases, and the entity type), as well as triples from relations with other entities. Information regarding two linked entities can be obtained by following the graph's path from the head entity (the entity under observation) and taking the N number of hops to any given nodes. As such, we view the triples as N-hop triples. For instance, in our running example, we see that the textual mention *"Zaire"* appears in the aliases of the entity. Suppose we added KG context to the model but hold out the aliases. Does the model still predict the right entity? To validate this research question, carry out exhaustive experiments based on different entity context configurations in section 6.2.3.

- What is the behaviour of different Deep Neural models under various forms of KG context? Do the several elements of these entities and their triples exhibit similar behaviour across neural models (For example, attention-based networks vs. fine-tuned transformer architectures)? To validate this research question, we employ two variants of neural network models. First, the DCA (Dynamic Context Augmentation) model by Yang et al. [28], which uses an attentive neural network, and the transformer-based XLNet [52].

- What is the role of the underlying data on the performance of context-induced models? Is the role played by any form of context generalisable across different datasets, or is the behaviour specific to the dataset.

6.2.2 Relevance in Entity Disambiguation Task Definition

In this section, we continue our experiments tackling the Entity disambiguation (ED) problem where a natural language (NL) sentence, is pre-annotated with entity mentions. The ED task aims to link each mention to its corresponding gold entity from a given knowledge base (KB). Consider a sequence of natural language words and symbols denoted as:
$\mathcal{W} = \{w_1, w_2, w_3, \ldots, w_{|\mathcal{W}|}\}$. The set of entity mentions in $\mathcal{W}$ is represented by $\mathcal{M} = (m_1, m_2, ..., m_k)$ $(k \leq |\mathcal{W}|)$ where each mention m_x is a sequence of words starting from a begin position i to end position j: $m_x^{(i,j)} = (w_i, w_{i+1}, ..., w_j)$ $(0 < i, j \leq |\mathcal{W}|)$. Given the set of entities $\mathcal{E}$ from the KB, the entity disambiguation (ED) is therefore a function $\Theta : \mathcal{M} \rightarrow \mathcal{E}$ such that, the output tuple (m_x, e^*) matches the mention m_x to it's ground truth entity e^* in the KB. In the first step, the task is to reduce the search space for finding the correct entity by generating a reduced set of possible (highly likely) entities to match the

identified mention, called the candidate set $C(m_x) = \{e_1^x, ..., e_n^x \mid e_i^x \in \mathcal{E}\}$. A disambiguation model fits a similarity score function on each of the candidate entities e_i^x against the corresponding mention m_x as a probability measure p_i. The probability score is thus employed into a ranking or classification objective.

$$\gamma = \arg\max_{p_i}\{\mathcal{P}(p_i \mid m_x, e_i^x)\}$$

Source Context: The entity mention often offers very little semantics when used in isolation. The major challenge in entity disambiguation is to obtain high-quality features of each entity mention for accurate entity prediction. The *local context* refers to the immediate neighbourhood (commonly the sentence) in which the mention appears. More features can be captured in several cases by considering other entity occurrences mentioned in the whole document. The *global context* of an entity is the discounted representation by considering all sentences in which the entity appeared in a whole document. The *local* and *global* contexts form the *Source Context* of an entity mention. In this work, we consider the sentence $\mathcal{W}$ in which the mention is detected as the source context. therefore source context $\mathcal{SC} = \mathcal{SC}_L \cup \mathcal{SC}_R \cup \Gamma$ where $\mathcal{SC}_L = \{w_1, \ldots, w_{x-1} | w_i \in \mathcal{W}\}$ $\mathcal{SC}_R = \{w_{j+1}, \ldots, w_{|\mathcal{W}|}\}$ refer to the left and right context context respectively, and Γ refers to extra features derived from the source sentence.

Algorithmus 2 : BUILDENTITYCONTEXT

Input : $\langle e, KG, N\rangle$; $e \leftarrow$ mentioned entity; $KG \leftarrow$ knowledge graph; $N \leftarrow$ max number of hops
Output : a double: set of entity attribute triples, and set N-hop triples

$A^e, \mathcal{T}^e \leftarrow []$
Q $\leftarrow \emptyset$
Q.enqueue(e)
$i \leftarrow 0$
while $Q \neq \emptyset$ ***and*** $i \leq N$ **do**
  v = Q.dequeue
  for ***each*** $u \in KG.Adj[v]$ **do**
    if $u \in \mathcal{L}$ ***and*** $v = e$ **then**
      $\mathcal{A}^e$**.append**$(F(v), F(vu), F(u))$
    else if $u \notin \mathcal{L}$ **then**
      $\mathcal{T}^e$**.append**$(F(v), F(vu), F(u))$
      Q.enqueue(u)
  $i \leftarrow i + 1$
return $(A^e, \mathcal{T}^e)$

KG Entity Context : From the preliminaries discussion in section 2.1, there is a rich semantic representation of entities in KBs, especially the structured KGs. General KBs offer a textual description of the entity that contain relevant relations with the source context. KGs on the other hand provide well structured relationships that are classified as entity attribute context $\mathcal{A}^{e_i^x}$ and entity triple context $\mathcal{T}^{e_i^x}$ where $e_i^x \in C(m_x^{(i,j)})$ is a candidate entity for mention $m_x^{(i,j)}$. Thus $(\mathcal{A}^{e_i^x}, \mathcal{T}^{e_i^x})$ = `BuildEntityContex`(e_i^x, KG, N).

$$\gamma = \arg\max_{p_i}\{\mathcal{P}(p_i \mid m_x, e_i^x, \mathcal{SC}, C \subseteq \mathcal{A}^{e_i^x} \cup \mathcal{T}^{e_i^x})\}$$

Models : Assuming a model agnostic approach, any given model Ψ fits that fits the mention and source context to the candidate entity and we seek to evaluate how well the model fits the data and the behaviour under extra entity specific contextual information. For experimentation, we derive different forms of the KG context.

$$\gamma_k^{\Psi_m} = \arg\max_{p_i}\{\mathcal{P}(p_i \mid \Psi_m(m_x, e_i^x, \mathcal{SC}, C_k)\}$$

Building the entity context To build an entity context, we perform a breadth first traversal on the subgraph rooted at the mentioned entity (i.e. the entity whose context is to be fetched). In this case, when we visit the root node, we retrieve the entity attribute context ($\mathcal{A}^e$), and depending on the number of hops, we recursively visit the adjacent vertices. However for every subsequent node, we only retried the E2E triples as described in algorithm 2.

6.2.3 Approach : Evaluating Knowledge Relevance

The section described the approach for KG-enhanced contextualised entity disambiguation. Figure 6.4, shows the overall process to modelling KG context for the entity disambiguation task. In this work, we conceive a model-agnostic view and define an approach consisting of three major stages: Input representation, Sentence-pair encoding, and scoring. Only the second stage requires a design choice of a specific model for learning a combined representation of the source and entity KG context. This section describes the steps involved in our approach, followed by a summary of the two models we selected for our experiments.

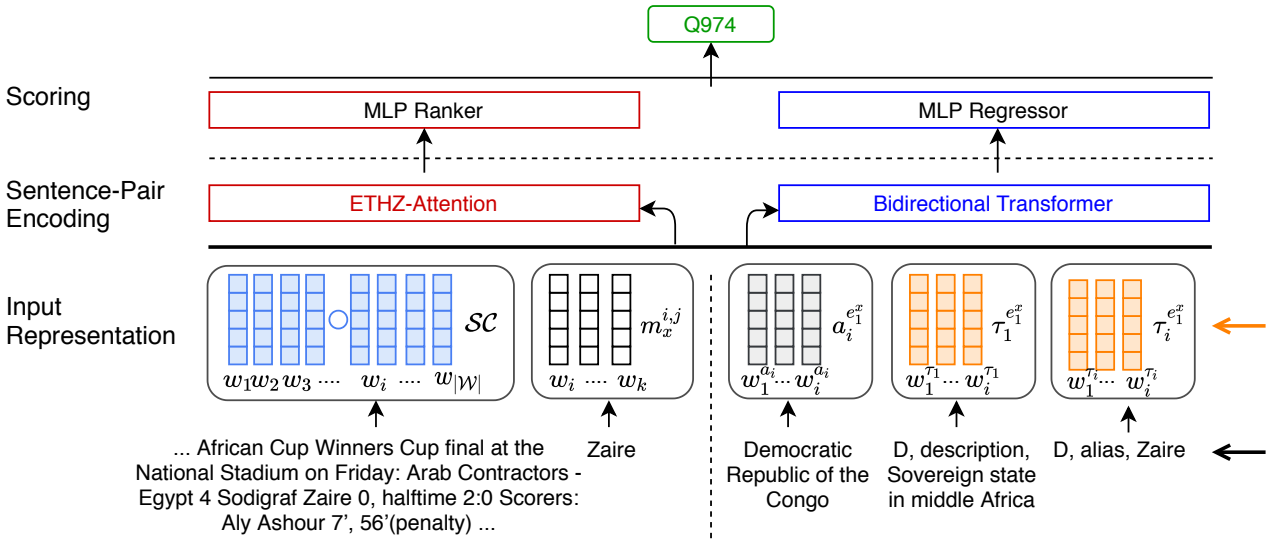


Figure 6.4: Approach : Entity-Context-Enhanced Disambiguation portraying the input representation and the selected models. The abbreviation D. refers to "Democratic Republic of the Congo"

Context Enhanced Disambiguation

Input Representation Text inputs for deep learning-based NLP models generally take the form of a sequence of word vectors. Such vectors are either trainable embedding learned within the first layer of the model or are fetched from static pre-trained vectors in a given vocabulary. Regardless of whether the vectors are trainable or static, a context-enhanced disambiguation model requires that the input tokens'

vector representation encapsulate both the semantics and meaning expressed by the token as the relative meaning defined by its partition (segment). Our approach's input consists of the four partitions illustrated in section 6.2.2. Therefore, the representation must capture the singular token semantics and the semantics expressed by its relative partition. For instance, the word "Zaire" may have the same meaning when it appears in the mentioned segment of figure 6.4 and when it appears in the third triple segment. Let's represent the embedding vector for a given token in the input as $\vec{w}$, and the vector representation for the input partition as $\vec{o}$. In contrast, the elective absolute positional embedding is denoted as $\vec{p}$. The final token representation is given by : $Combine(\vec{w}, \vec{o}, \vec{p})$. The function $Combine()$ is a model-dependent choice between concatenation, addition, averaging, or multiplication.

Sequence-pair Encoder The sequence-pair encoding aims to obtain a single unified vector representation that combines the semantics and expresses the proximity of the two sequences (source and mention sequence vs. candidate entity and entity kg-context sequence). With the progress made towards sequence encoding through attention mechanism, the intuition obtains different initial contextual word representation for the two sequences, followed by extrapolated, selective inter-matching of tokens across the two sequences. Depending on the attention mechanism's granularity designed in a model, the final representation often involves the aggregation of several attention functions or units' outputs. Our decision to use attention-based approaches in the sequence-pair encoding is informed by the tremendous success achieved in previous researches such as [31, 32, 261]. As indicated in similar to works [261, 262], attention take two representation and defines a weighted representation that entails their compatibility. Let us take a hidden layer representation of a given attention unit as h_L if the unit exists in the left sequence. It is h_R if the unit exists in the right sequence (a unit can be a token, token block or the whole sequence). An attention function is performed as follows:

$$a = \alpha(W^L h_L \ (op) \ W^R h_R)$$

Where a is the attention vector, α is a nonlinear function such as $tanh$ or $ReLU$, W^L, W^R are the trainable weight matrices for the left and right units, and (op) is an operation depending on the type of attention used (concatenation, addition, and multiplication). If we then take several of such units respectively, the final value of a unit representation is given by combining the attention vector about other units with the unit's original vector, e.g., for h_L .

$$h'_L = h_L softmax(w^T a_i)$$

For final sentence-pair representation, a non-linear function is used to combine the first sequence's attention weighted representation with the last output vector encoding the two input sequences. We generate another representation based on the second sequence and concatenate the two in a two-way attention mechanism.

Scoring At the scoring stage, we take the vector representation from the sequence-pair encoder and fit it into an objective that produces the final prediction. The common scoring objectives include classification, regression, or ranking. This entails passing the encoding through one or two fully connected layers with the selected scoring objective. If we assume the feature vector representation for the source and candidate entity, the ranking objective takes the max-margin loss while the classification employs the cross-entropy loss function.

Models

In our approach, we chose two different models that explicitly influence the behaviour of the three stages described before. The first model is the DCA model [28] based on the ETHZ-attention, and the second is the pre-trained transformer architecture XLNet [52]. In the following, we briefly describe each.

DCA Model The ETHZ-Attn was introduced in work by Ganea et al. 2017 [196] in which a 2-layer feed-forward network is used to aggregate scores obtained from 3 different local features : (1) A prior probability score for the mention-entity pair distribution empirically drawn from a large corpus. (2) A score that estimates the similarity between the the source context and the entity context, and (3) Type similarity score that relates the type of the candidate entity to the mention. The model then seeks to amass information from entities that have been already linked. Such information becomes a dynamic context to boost subsequent linking decisions. For removing irrelevant entities in the dynamic context, an attention mechanism is again applied [28]. The model employs a ranking objective to select the right candidate finally.

Adding KG Context to DCA: The attention mechanism employed in the DCA model allows extended range matching of tokens and segments. The original model was therefore trained using Wikipedia paragraphs as entity context. However, Wikipedia paragraphs are not adequately structured, hence do not provide concise entity context representation. In our experiments, we replace this context with the Wikidata entity context. The initialisation used in this model is drawn from the static Glove embeddings. We feed the triples in the form of word embeddings for the natural language forms of the triples. For the DCA model, the KG entity context input takes a sequence similar to the example below.

> *Democratic Republic of the Congo, alias DRC, alias DR Congo, alias, Congo-Kinshasa, alias Zaire, alias Dem. Republic of the Congo, alias Dem. Republic of Congo, alias Dem. Rep. Congo, alias Congo (Kinshasa), alias COD, description sovereign state in Central Africa, instance of country, instance of sovereign state, part of Middle Africa, ...*

Fine-tuning XLNet Model Language models such as BERT[32], RoBERTa[53]yang2019learning, and XLNet[52] have become state of the art for several tasks. Mulang' et al. [10] experimented with RoBERTa and XLNet for context representation and determined that XLNet is a more stable model under extra context in the input sequence. We, therefore, borrow this insight and implement our approach using a fine-tuned XLNet model. We add a regression layer to obtain the scores shown in figure 6.4.

Adding KG Context to XLNet: The architecture for pre-trained transformer models lends easily to our need to represent separate segment embeddings for each input portion. We employ the segment separator token *[SEP]* between each triple, which allows the embedding for each triple to be unique from the rest of the input. From figure 6.4, each $a_i^{e_1^x} \in A^{e_1^x}$ refers to a triple derived from the entity attribute context set $A^{e_1^x}$ of the candidate entity e_1^x, while each $\tau_i^{e_1^x} \in \mathcal{T}^{e_1^x}$ refer to triple connecting the entity to other entities. For our running example, the following represents the entity context.

> *[SEP] Democratic Republic of the Congo alias DRC, DR Congo, Congo-Kinshasa, Zaire, Dem. Republic of the Congo, Dem. Republic of Congo, Dem. Rep. Congo, Congo (Kinshasa), COD, [SEP] description sovereign state in Central Africa [SEP] instance of country, sovereign state [SEP] part of Middle Africa, ...*

6.2.4 Implementation and Evaluation

Implementation

Our implementation involves retrieving the entity context from Wikidata. We use SPARQL[3] queries to fetch 1-and-2-hop triples from the Wikidata SPARQL endpoint[4] similar to the queries used by Cetoli et. al.[73]

```
PREFIX wikibase: <http://wikiba.se/ontology#>
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?item0 ?rel ?item1 WHERE {
  ?item rdfs:label "%s"@en .
  ?item ?r ?item0 .
  ?item0 ?rel ?item1 .
  FILTER regex (str(?item0), '(statement)') . #2-hops only for inline
  FILTER regex (str(?item1), '^((?!statement).)*$') . #statements
  FILTER regex (str(?item1), '^((?!https).)*$') .
} limit XXX
```

The DCA model is obtained from the shared code by Yang et al. [28] under their GitHub link[5]. We train the supervised learning model (DCA-SL) in which we replace the Wikipedia descriptions with the entity context from Wikidata. This model is trained up to 400 epochs, achieving convergence between epochs 290 and 297. For the second model, we employ the XLNet based model, with implementation from Huggingface[6]. The model is trained for three epochs with a learning rate of 2e-5, per GPU batch size of 4 on 8 NVIDIA GeForce GTX 1080 and max sequence length at 512.

Dataset	# mentions	Gold recall
AIDA-train	18448	-
AIDA-A	4791	97.3
AIDA-B	4485	98.3
MSNBC	656	98.5
AQUAINT	727	94.2
ACE2004	257	90.6
CWEB	11154	91.1
WIKI	6821	92.4

Table 6.4: Dataset Statistics. Gold recall is the percentage of mentions that are aligned to ground truth entities

[3] https://www.w3.org/TR/sparql11-query/
[4] https://query.wikidata.org/
[5] https://github.com/YoungXiyuan/DCA
[6] https://github.com/huggingface/transformers

Evaluation Setup

Datasets. For our evaluation, we use six (6) standard datasets studied in the community namely: CoNLL-AIDA datatset [147], MSNBC [25], AQUAINT [263], ACE2004 [148], CWEB [244], and WIKI [244] datasets. Similar to previous studies, we train all models only on the AIDA-CoNLL training set and perform both on the CoNLL-AIDA test set (in-KB evaluation) and across the other domain dataset (Out-of-KB evaluation). This evaluation setting has been utilised by most researchers in this task [9, 28, 196]. Some statistics of these datasets is give in table 6.4.

Metrics: We use the Accuracy metric for our evaluation, where the AIDA-testA and AIDA-testB are in the same domain as the training set (AIDA-train). For these datasets, the metric is referred to as the In-KB accuracy. Evaluation done on the rest of the datasets out of the AIDA-train domain is referred to as out-of-KB accuracy. For a given component, the correct mention entity ratio is linked to the overall number of mentions in the dataset.

Approach	**In-KB. Acc.**
Local Models	
Lazic et. al. (2015) [264]	86.40
Globerson et. al. (2016) [265]	87.20
Yamada et al. (2016) [29]	87.20
Ganea&Hofmann (2017) [196]	88.80
BERT-Entity-Sim (Chen et.al.'20) [57]	90.06
Local and Global	
Huang, Heck, and Ji (2015) [266]	86.60
Ganea et al. (2016) [267]	87.60
Chisholm and Hachey (2015) [268]	88.70
Guo and Barbosa (2016) [244]	89.00
Globerson et al. (2016) [265]	91.00
Yamada et al. (2016) [29]	91.50
Ganea&Hofmann (2017) [196]	92.22
Yang et al. (2018) [217]	93.00
Le&Titov (2018) [218]	93.07
DeepType (2018) [34]	94.88
Fang et al. (2019) [194]	94.30
Le& Titov (2019) [9]	89.66
Context Augmented Models	
DCA-SL (2019)[28]	94.64
Chen et al. (2020) [57]	93.54
DCA-SL+WDT (ours)	**94.94**
XLNet-WDT (ours)	86.78

Table 6.5: IN-KB: scores on AIDA-B (test set).

Approach	MSNBC	AQUAINT	ACE2004	CWEB	WIKI	Avg.
Milne and Witten [263]	78.00	85.00	81.00	64.10	81.70	77.96
Hoffart et al. [147]	79.00	56.00	80.00	58.60	63.00	67.32
Ratinov et al. [148]	75.00	83.00	82.00	56.20	67.20	72.68
Cheng and Roth [33]	90.00	90.00	86.00	67.50	73.40	81.38
Guo and Barbosa [244]	92.00	87.00	88.00	87.00	84.50	85.70
Ganea and Hofmann [196]	93.70	88.50	88.50	77.90	77.50	85.22
KB Context Augmented Models						
DCA-SL	94.57	87.38	89.44	73.47	78.16	84.60
DCA-SL-WDT	94.41	88.25	89.33	74.55	78.60	85.03
XLNet-WDT	86.78	87.42	74.52	89.2	62.56	68.25

Table 6.6: Out-of-KB Accuracy scores for out-of-KB test sets.

Model Config	AIDA-B	MSNBC	AQUAINT	ACE2004	CWEB	WIKI
DCA-SL-WDT+labels	94.56	94.41	87.97	88.93	74.23	78.44
+aliases	94.65	94.41	87.55	88.13	74.55	78.35
+description	94.69	94.41	87.55	89.34	74.38	78.41
+instance	94.64	94.41	87.55	89.74	74.44	78.68
+e2eTriples	94.94	94.41	88.25	88.93	74.55	78.10
XLNet-WDT-labels	83.05	75.93	72.02	82.5	61.5	64.93
+aliases	83.06	84.47	72.02	82.5	61.5	66.56
+description	85.56	88.20	75.85	87.92	64.67	70.99
+instance	86.36	88.35	77.91	87.5	65.34	71.72
+e2eTriples	86.78	87.42	74.52	89.2	62.56	68.25

Table 6.7: Ablation study : Impact of different KG Context configuration on different datasets

Evaluation Results

Table 6.5 shows our two models against the state-of-the-art models on this dataset. We categorize the models according to their use of contextual information: 1) Models that utize only the local context in the sentence. 2) Models that employ both the local and global contexts (source context - according to the definition in section 6.2.2). And 3) models that utilise both source context and entity context (both general knowledge-based and knowledge graphs). We can observe that the DCA-SL-WDT and XLNet-WDT models trained with Wikidata context exhibit better performance than those trained without context. We also observe that the bidirectional attention-based method (ETHZ-Att) employed in the DCA model achieves higher accuracy than a fine-tuned XLNet model. This explains why the other models including [34, 57, 194, 196, 218] perform better than the fine-tuned model. However, comparing the minimalistic context model's performance that includes only KG-labels (cf. Table 6.8), We see that the full context model obtains a near 9% jump in performance. This is an empirical validation of the first aspect of RQ3-c which presumes that different forms of knowledge context have different impact on different models. Table 6.8 indicates the level of influence incremental addition of several forms of context has. Overall, the highest jump is seen when the entity to entity triples are added, and the second

most impacting form of triples are the descriptions.

Concerning the last aspect of our experiments, we observe that entity-specific context positively influences both the DCA and the fine-tuned XLNet models. However, the jump in performance is more pronounced in the fine-tuned model. This is partly because the original DCA model was already trained with entity-specific context from Wikipedia, close to Wikidata. Albeit, the model, still benefits from the better structured and more specific information from Wikidata. On the other hand, the DCA model is trained from scratch with a profound attention mechanism. As such, the model can cross-reference similarities appearing over a long-range, as opposed to the XLNet model that is fine-tuned on a mere three epochs. We still observe some noteworthy improvement with the fine-tuned model too.

Model Config	**CoNLL-AIDA Test-A Recall @**					**CoNLL-AIDA Test-B Recall @**				
	1	2	3	4	5	1	2	3	4	5
XLNet-WDT										
+labels	90.56	95.53	96.77	97.55	97.94	83.05	89.43	91.94	92.43	93.19
+aliases	90.56	95.53	96.77	97.55	97.94	83.05	89.43	91.94	92.43	93.19
+description	92.91	96.60	97.81	98.09	98.32	85.56	92.07	94.31	95.55	96.47
+instance	92.24	97.25	98.60	98.97	99.03	86.36	92.85	94.34	95.07	95.46
+e2eTriples	91.87	96.56	97.66	98.17	98.6	86.78	92.5	94.4	95.5	95.97

Table 6.8: Recall @N : Analyzing the accuracy at given top N recall values for the fine-tuned XLNet dataset

Table 6.8 shows the models' performance using different configurations and how they perform on different datasets. We can observe that the performance varies depending on the underlying dataset. Whereas in AIDA-B (in-KB evaluation), the incremental addition of new forms of contextual information leads to a concordant increase in performance. The out-of-KB datasets do not exhibit the same behaviour. We can deduce a response to our final research concern from this table. For example, the addition of descriptions shows a drop in performance in the MSNBC dataset. In contrast, in all other datasets, there is a general increase in performance.

6.2.5 Discussion and Insights

Our empirical studies studied three research questions to cover various dimensions of the KG context effect on entity linking models. Across datasets and forms of context, we conclude that attention-based models can express better granular targeted matching of information from both sequences. From the evaluation results, we can observe that fine-tuning does not allow enough learning of features and parameters compared to the attention-based model. According to table 7, when there is more data, the fine-tuning process slightly drops performance in the dev set yet performs better in the test set. This indicates that the fine-tuning process does not allow the model to capture enough signals when the sequence contains more context. Due to very few training steps, the model has not converged. However, we can not increase the number of fine-tuning steps; otherwise, we will overfit the data. The observed behaviour can also explain why the DCA model outperforms the XLNet fine-tuned model. We believe that a pre-trained XLNet model with task-specific contextual information should be able to capture enough signals. We also conclude that the recall at position 2 surpasses all existing models, indicating that the model can sift out all dissimilar candidates and struggles only with very closely related candidates. Another significant insight is that impact of context is particular to the underlying dataset. For example, on the AQUAINT dataset, adding e2e triples in XLNet harms the overall performance compared to when a combination of label, alias, description, and entity type is added as context. The observed behaviour

could be interpreted as follows: adding additional triples may have made the model confused to fetch the correct context per entity mention. Hence, it can be concluded that there exists no specific form of KG context whose impact will increase the model. Based on our analysis, we leave readers with the following open questions for future research:

- Considering the impact of context is dependent on the underlying data, it remains an option question that how one can intelligently Select the context even before feeding into the model?
- As Wikidata is hyper-relation KG, how can the edges' properties also be utilised to provide additional task-specific context?
- The context from the previous sentences have also been utilised in the literature [35]. It would be interesting to see if the document context can be utilised along with the KG context.
- Not all KG contexts always improves the performance of the underlying deep learning model [10]. Hence, it remains an option question in which case KG context negatively impacts the overall entity linking performance.

6.3 Summary

In this chapter, we addressed the gap that exists concerning the overall value of KG context. Due to the partial or none use of knowledge context in NED models, an open question has remained. Invariably, concerning the generalisability and the complete use of information from KGs, research has thus far lagged behind. For instance, several models have employed only the descriptions [28, 35], others have chosen to include the type information [9, 34], while still others have chosen entity aliases as the KG context [24, 201]. In chapter 5 we described our approach to encode information from KGs for neural network-based approaches. As discussed in section 5.2 and section 5.3, we employed only a portion of the available information on entities. Due to this research gap, we set out to evaluate the overall impact of KG context considering the whole extent of entity attributes and triple relationships. For our first set of experiments (section 6.1), we define three closely related sub-search questions to our running research question (RQ3). Of interest in our evaluation, was also to understand the depth of knowledge encoded in the pre-trained transformer models, especially given their excessive use in SOTA models. We demonstrate that pre-trained Transformer models, although powerful, are limited to capturing context available purely on the texts concerning the original training corpus. This emanates from an observation we made when we added extra task-specific context from the KG that improved the performance. However, there is a limit to the number of triples that can improve performance. We note that 2-hop triple resulted in a negative or little impact on transformer performance. Our triple context can be generalised (for Wikipedia) and shows a positive effect on the NED model for Wikipedia, leading into a new SOTA for the AIDA-CoNLL dataset.

In the second part of our research question, we demonstrate the value of the knowledge graph context in entity disambiguation. Our work proves the general hypothesis that KG context improves the performance of deep learning-based NLP models. The caveat, therefore, is the necessity of appropriate representation. We employ the verbalised form of the KG triples and incorporate this as part of the input sequence to the models in our work. With proper demarcation tokens, current models can attend to specific, insightful portions of the input. Our experiments indicate that various forms of information from the KGs exhibit varying behaviour with different models and other datasets. Therefore, it is paramount to consider what aspects of the KG context would be relevant for a given dataset. Moreover, we determine that the representation of the context is as important as the context itself. For instance, the attention-based

DCA model can capture contextual signals from a long-range sequence, while the fine-tuned XLNet model is limited in this respect. We, therefore, recommend the use of the pre-training or more elaborate attentive model. Furthermore, we see the KG embedding avenue as a rich representation to capture the entity context into encoding vectors for input into the model. Experimenting with various KG embedding approaches is a logical next step for this work. We also view the ability to select the optimal contextual information for any given model dynamically, which would tremendously improve performance.

CHAPTER 7

Application of Knowledge Context to Explanation Regeneration

The previous chapters in this thesis have traversed the landscape of methods and approaches for entity and relation linking. Chapter 2 presented concepts and underlying techniques that act as background basis for the challenges and research questions described in chapter 1. Chapter 3 discusses existing research findings and tools that have been developed in the community for entity and relation linking tasks. We then took a look into the opportunity presented by the knowledge context through term graphs by providing an approach to unify textual and knowledge representations of relations (described in the Rematch approach of Chapter 4). We determined that we can overcome the semantic gap related challenges existing between NL and KGs by articulating comparable structures from the text and KG to enhance similarity matching. Since relation linking has been reported to tremendously improve when entities are known, we embarked on the second task of identifying and linking named entities. Our Arjun approach in chapter 5 seeks to address the challenge of effectively encoding knowledge context for Neural Network (NN) models. Two implementations are thus presented that are based on different techniques: an attentive neural network based encoder-decoder architecture, and a transformer based model. Both of these approaches outperform state-of-the-art models on end to end entity linking. Subsequently, we tackle the question of generalisability of knowledge context in chapter 6. Initially we tackle the sub research question concerning impact of the volume of knowledge context used (6.1) taking into account a complete view of context that encompasses entity attributes as well as 1-and-2-hop triple relations, and a model agnostic behaviour of context for entity disambiguation. The evaluation in section 6.2 tackles the sub research question, and evaluates the relevance of different forms of knowledge context.

Generalising knowledge context in other domains : In this chapter, we look to reuse the experience gained from the approaches and experiments in the previous chapters into an emerging task in NLP. The explanation generation aims at improving the *interpretability* of the machine learning process (i.e., to reveal how a system arrives at the prediction) [269, 270], or *trustworthiness* of the result (i.e., to make the prediction more believable as correct by justifying it) [271, 272]. On the one hand, interpretability enables better comprehension of the so-called black box machine learning, while on the other hand, trustworthiness determines usefulness of the technology in decision critical domains such as in medicine and law. In other words, machine learning predictions cannot be acted upon in such domains without additional supporting evidence, as the consequences may be dramatic. Explanation regeneration for elementary science task takes a corpus of elementary science question and correct answer pairs ('QA pairs' hence) taken from standardised tests, to automatically justify the correct answer with an explanation generated from science and commonsense facts. As a continuation to the overall agenda of explanation

regeneration, [74] introduced the TextGraph-13 shared task involves performing multi-hop inference for justifying the correct answer choice in multiple-choice QA. The current state-of-the-art performances on the task are still low (below 60% in F-score), indicating its challenging nature and the need for avenues to address the challenges for better performance. This chapter describes our solutions that rely on deriving features form knowledge context to enhance machine learning models for this task as an application of our findings in previous chapters .

Knowledge Context Application

This chapter presents an evaluation of knowledge context in a different domain and task from Entity and Relation Linking?

Contributing publications : Mulang' et.al. [273], D'Souza et.al [274].
All experiments for [273] were designed and carried out by the PhD candidate, who also participated in writing of the paper.
Experiments for [79] were only partially designed and carried out by the PhD candidate in collaboration with the first author.
The PhD candidate also participated in writing of the paper.

The rest of this chapter is structured as follows. In section 7.1 we first introduce the task by discussing a brief history of it's origins followed by a definition of the problem and description of the corpus we use. Our linguistic features based approach that derives knowledge context from knowledge bases for a SVM based solution is presented in section 7.2. After that, we present a fine-tuned transformer based approach infused with contextual information from textual characteristics in section 7.4. Finally we summarise in section 7.7.

7.1 The Explanation Regeneration Task

Generally, in multiple-choice QA exams, a student selects one answer to each question from among typically four choices and can explain why they made that particular choice based on their world and

Question Granite is a hard material and forms from cooling magma. Granite is a type of
Answer igneous rock
Explanation
(f1) igneous rocks or minerals are formed from magma or lava cooling;
(f2) igneous is a kind of rock;
(f3) a type is synonymous with a kind;
rock is hard;
to cause the formation of means to form;
metamorphic rock is a kind of rock;
cooling or colder means removing or reducing or decreasing heat or temperature;

Table 7.1: Example depicting lexical hop between Question and Correct Answer pair not just with correct facts, but also with incorrect fact candidates.

commonsense knowledge. For a machine, on the other hand, constructing an explanation for the correct answer can be challenging for the following reasons: 1) It can be a multi-step process since some facts may directly relate to the question and correct answer, but there may be others that build on the earlier facts provided as explanation. Consider in table 7.1, facts f1 and f2 directly relate to the question and

Question A student put 200 milliliters (mL) of water into a pot, sets the pot on a burner, and heats the water to boil. When the pot is taken off the burner, it contains only 180 milliliters (mL) of water. What happened to the rest of the water?
Answer it turned into water vapor
Explanation
(f1) to turn means to change
(f2) water is in the gas state, called water vapor, for temperatures between 373 or 212 or 100 and 100000000000 k or f or c
(f3) boiling or evaporation means change from a liquid into a gas by adding heat energy
(f4) water is a kind of liquid
(f5) evaporation causes amount of water to decrease
(f6) a burner is made of metal
(f7) a burner is a part of a stove
(f8) a stove generates heat for cooking usually
(f9) pot or pan or frying pan is made of metal for cooking
(f10) metal is a thermal or thermal energy conductor
(f11) a thermal energy conductor transfers heat from warmer objects or hotter objects to cooler objects
(f12) if a thermal conductor or an object is exposed to a source of heat then that conductor or that object may become hot or warm
(f13) a source of something emits or produces or generates that something
(f14) if one surface or one substance or one object touches something then one is exposed to that something
(f15) being on something or placed in something or placed over something means touching that something
(f16) heat energy is synonymous with thermal energy
(f17) transferring is similar to adding
(f18) conductivity is a property of a material or substance
(f19) if an object is made of a material then that object has the properties of that material
(f20) metal is a kind of material
(f21) a burner is a kind of object or surface

Table 7.2: Example Instance in the WorldTree Corpus [272]. A Question and Correct Answer pair (QA pair) with its Explanation comprising 21 logically ordered facts (f1, f2,..., f21). In the WorldTree, explanation lengths vary between 1 and 21 facts. This selected example Explanation with 21 facts is the longest in the corpus. Characteristic of the data design, facts in explanations lexically overlap (shown as underlined words) with the question or answer or other facts.

correct answer, however, fact f3 is an elaboration for f2. This phenomenon is even more prevalent in longer explanations. Consider the example in table 7.2, where facts f6 to f14 are indirectly related to the question or correct answer, nonetheless are essential to the logical sequence of facts to explain the phenomenon of "heating of water caused by the pot on the burner." And, 2) this multi-step inference is highly amenable to the phenomena of *semantic drift*, i.e. the tendency of composing spurious inference chains leading to wrong conclusions [275, 276]. This is depicted by the facts in red in table 7.1, that on the surface are linguistically related to the question and correct answer, but are not semantically relevant to the explanation for the correct answer.

In this work, we address the aforementioned machine learning challenges by simultaneously expanding both the linguistic and conceptual vocabulary of the question, correct answer, and explanation fact words, in a domain-targeted manner as features for machine learning. By expanding the vocabulary, we aimed to obtain greater number of lexical matches between the QA pair and explanation facts. In this way, we also indirectly aimed to facilitate improved semantic relatedness between the QA pair and their explanation facts via their expected greater number of lexical matches. In all, six differing and novel information categories were leveraged to represent the instances for learning. While in an earlier system [277], we have similarly employed a features-based approach for this task, in our new version presented in this paper, the generic features of that system are replaced by a domain-targeted set.

With respect to the machine learning strategy, we adopt the *learning how to order* problem formulation since the annotated explanations in the WorldTree corpus [272] are made up of logically ordered facts in discourse. Specifically, in the context of the WorldTree, the automatic task entails learning and predicting preferences over candidate facts per QA pair explanation. Generally, learning a preference function involves ranking facts from a candidate set, i.e. the relevant facts before the irrelevant facts, and the relevant facts in order w.r.t. each other. Further, it also includes an implicit "abstaining" from making ranking decisions between the irrelevant facts. Then during testing, new QA pair explanations are generated by predicting the order for the facts per the trained preference function. Since the problem doesn't involve a total ordering of all facts in the tablestore for the explanations, but only the relevant facts, we adopt the preference learning approach [278, 279] rather than a ranking approach, where the latter entails a total ordering. Nevertheless, preference ranking are a class of problems that subsume ranking functions. In fact, among the problems in the realm of preference learning, the task of "learning to rank" has probably received the most attention in the literature so far, and a number of different ranking problems have already been introduced. In this work, we compare a pointwise preference learning approach versus the pairwise ranking approach. Further, the scoring and loss functions for both pointwise and pairwise ranking is from the support vector machine class of learning algorithms. Support vector machines are preferred by many as a strong classifier needing less computation power than neural models. Although we are not the first to contrast pointwise and pairwise learning, our study offers new observations on the comparison of these two techniques on a new problem, i.e. the ranking of facts to construct explanations. In this way, we build on our earlier system [277] that tested only the pairwise ranking approach with its generic features set. The main contributions discussed in this chapter include:

- A domain-targeted space of representative knowledge context features derived from world and commonsense knowledge, utilised to assist associate a QA pair with the candidate explanation facts both linguistically and semantically.

- A unique contextual feature characterising concepts in text that is able to improve performance of transformer based deep learning models.

7.1.1 Problem Definition

Given a question $q = \{w_1, w_2, .., w_{|q|}\}$, it's correct answer $a = \{w_1, w_2, ..., w_{|a|}\}$, and a set of explanation facts $\mathcal{E}$ s.t. every $e \in \mathcal{E} = \{w_1, w_2, ..., w_{|e|}\}$ where $w_i \in V$ for some vocabulary V. Following the definition for the TextGraphs-13 MIER task [74], the aim is to obtain, for every question and its correct answer, a ordered list of facts from the explanations set that are coherent in discourse. By definition, for a question-answer pair q, a, there exists a set $\mathcal{R}_{q,a} \subseteq \mathcal{E}$ called the relevant set. The task aims to generate an ordered list of explanation facts $\mathcal{E}^o$ such that $\forall e^o, e \in \mathcal{E} : e^o \in \mathcal{R}_{q,a} \wedge e \notin \mathcal{R}_{q,a}\ ,\ pos(e^o, \mathcal{E}^o) < pos(e, \mathcal{E}^o)$. We define, for any given q,a pair the ordered list as $\mathcal{E}^o_{q,a} = Reorder(\{(e_k, \gamma_k) \mid e_k \in \mathcal{E}\})$ where γ_k is an associated relevance score obtained by predicting a proximity value given as $\gamma_k = \Phi(q, a, e_k, \theta)$. Φ is a regression function and the optional θ, represents any extra input parameters to the model to enhance prediction performance. In our work, we consider two avenues for knowledge context to assist this task namely: inducing focus words from both the question-answer side, and the explanation side. Adapted from [77], a focus word $v \in V$ is a word with a concreteness score of between the values 3.0 and 4.3.

7.1.2 The Corpus

The data used in this study comes from the WorldTree corpus[1] [272]. The WorldTree corpus [272] newly released a manually authored knowledge base of semi-structured tables (also called 'a tablestore') of nearly 5,000 elementary science and commonsense facts. These facts were then used to construct explanations of varying lengths as justifications for correct answers in a multiple-choice QA setting. As an example, consider a QA pair and its explanation data instance from the corpus in table 7.2. This corpus, inadvertently and similar to other tests for machine intelligence such as the Turing test [280], presents itself as another credible test for evaluating the *intelligence* of natural language inference systems but in the framework of standardised elementary science tests. Thus the systems could then be evaluated with respect to their language understanding, reasoning, and use of common-sense knowledge capacities via their generated explanations. It comprises a portion of the standardised elementary science exam questions, 3rd to 5th grades, drawn from the Aristo Reasoning Challenge (ARC) corpus [281]. The questions have multiple choice answers with the correct answer known. Each question-correct answer pair (QA pair) in the WorldTree corpus [272] has detailed human-annotated explanations, consisting of between 1 to 21 facts that are arranged in logical discourse order w.r.t. each other. The QA pair instances are divided by the standard ARC train, development, and test sets. The WorldTree corpus then is provided as 1,190 training, 264 development, and 1,248 test instances where each instance is a QA pair and its explanation.

Explanations for Correct Answers to Elementary Science Questions

As alluded to above, QA pairs in the WorldTree corpus [272] are annotated with explanations of up to 21 facts (see in Fig. 1 the distribution of facts in the explanations in the training and development sets).

Total unique explanation facts:	4,789
Seen in training data:	2,694
Seen in development data:	964
Seen in training and development data:	589

[1] We use the TextGraphs2019 Explanation Reconstruction Shared Task Data Release available at http://cognitiveai.org/explanationbank/

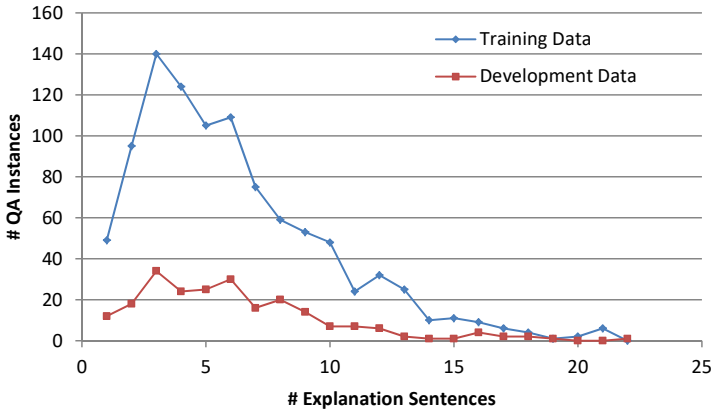


Figure 7.1: Facts in explanations per question-answer pair in the training and development datasets.

Based on corpus design decisions, the inclusion criteria for facts in explanations were: *lexical overlap*—facts lexically overlap with the question or answer, or with other facts in the explanation; and *coherency*—the explanation facts form a logically coherent discourse fragment. As a consequence of the *lexical overlap* characteristic, a traversal path can be traced between each QA pair and its explanation facts via multiple lexical hops (depicted in tables 7.2 and 7.1 via the underlined words). Further, as an additional annotation layer, facts in each training and development set explanation were categorized as one of three classes. These classes were determined by the role played by the fact in the explanation. Specifically, the classes were *Central*, *Grounding* and *Lexical Glue*. Central facts were defined as core scientific facts relevant to answering the question. E.g., facts such as "as the amount of rainfall increases in an area, the amount of available water in that area will increase." Grounding facts were those which connected to other core scientific facts present in the explanation. E.g., "rain is a kind of water" would connect "rain" and "water" present across two or more Central facts in the explanation. And lexical glue facts expressed synonymy or definitional relationships. E.g., "rainfall is the amount of rain an area receives." table 7.3 offers statistics on the overall prevalence of explanation facts across QA pairs in the training and development sets, and also per explanation fact category.

Total QA pairs		1,213
Total facts used		7,448
Facts per QA pair		6.14
Central	Total facts used	3,705
	Facts per QA pair	3.05
Grounding	Total facts used	2,131
	Facts per QA pair	1.76
Lexical Glue	Total facts used	1,612
	Facts per QA pair	1.32

Table 7.3: Corpus statistics for QA pairs w.r.t. their explanation facts from the WorldTree [272] training and development corpora combined

Next, additionally released with the corpus was a tablestore of 4,789 human-authored candidate facts from which the explanations were constructed. The tablestore facts were authored based on the elementary science themes of the ARC question-answering data. They are organized in 62 tables representing relation predicates such as *kind of* (e.g., an acorn is a kind of seed), *part of* (e.g., bark is a part of a tree), *cause*

(e.g., drought may cause wildfires); or the *actions* of organisms (e.g., some adult animals lay eggs); or the *properties of things* (e.g., an acid is acidic); or *if-then* conditions (e.g., when an animal sheds its fur, its fur becomes less dense). In table 7.4, we depict the table types whose facts belonged to at least 1% of the explanations in the training and development sets.

KINDOF	25.22	REQUIRES	2.87	ATTRIBUTE-VALUE-RANGE	1.53
SYNONYMY	14.27	PARTOF	2.74	CHANGE	1.53
ACTION	6.48	COUPLEDRELATIONSHIP	2.67	CHANGE-VEC	1.43
IF-THEN	5.31	SOURCEOF	1.89	EXAMPLES	1.43
CAUSE	4.17	CONTAINS	1.79	PROPERTIES-GENERIC	1.21
USEDFOR	4.17	AFFECT	1.73	TRANSFER	1.11
PROPERTIES-THINGS	3.58	MADEOF	1.69	AFFORDANCES	1.08

Table 7.4: Fact table types sorted by the proportion of their occurrence in explanations, for only 21 tables of 63 total that had facts participating in at least 1% of the training and development explanations.

7.2 Knowledge Context for Explanation Regeneration

In this section, we elaborate on our feature function ϕ introduced with our formal models used to transform a (q, ca, e) triplet to a one-hot encoded feature vector x as data instances for the learning algorithms. Our motivation in selecting these features was to encode linguistically the necessary world and commonsense knowledge required for unifying facts as explanations to Elementary Science QA. Further, with clear emphasis on *lexical overlap* as a characteristic association between QA pairs with the facts in explanations, we have features for this criteria including the following group.

7.2.1 Bags of Lexical Features

This feature group most generically encodes the lexical overlap criteria by including features as lemmas of $q/ca/e$; lemmas shared by q and e, ca and e, and q, ca and e; 5-, 4-, and 3-gram prefixes and suffixes of $q/ca/e$; 5-, 4-, and 3-gram prefixes and suffixes shared by q, ca, and e; and e's table type from the provided annotated tablestore data. In our experiments we use 70,949 total features from this category.

7.2.2 ConceptNet

We hypothesise that semantic features, in particular commonsense knowledge, could be useful for the explanations to elementary science QA pairs. IN this feature category, we use 294,249 total features. Since elementary science questions query general knowledge about common nouns like animals, planets, occupations, etc., we find that ConceptNet [75] as a resource with its focus on the general meanings of all words, whether they be nouns, verbs, adjectives, or adverbs, and less on named entities, is perfectly suited to our task. Let us illustrate with an example.

In this example, ConceptNet tells us that the answer "rabbit" is an "animal" and a "herbivore", among other things. Extending the answer with this knowledge enables better semantic connection between the q, ca, and all three explanation facts, in the absence of which, the ranking could experience a semantic drift toward irrelevant explanation facts such as "long ears are a part of a rabbit" or "a jackrabbit is a kind of rabbit". Given the potential usefulness of ConceptNet for our task, we create conceptualisation features as follows. The top 50 conceptualisations of $q/ca/e$ words; top 50 conceptualisations shared by q and e, ca and e, and q, ca and e words; and commonsense ConceptNet facts that $q/ca/e$ words participate

Question Which animal eats only plants?
Answer Rabbit
Explanation
herbivores only eat plants;
a rabbit is a kind of herbivore;
a rabbit is a kind of animal;
ConceptNet conceptualisations for "rabbit"
animal, herbivore

in relations such as FormOf, IsA, HasContext, etc. with other terms (e.g., for word 'tea' in $q/ca/e$, the ConceptNet facts are 'tea ReceivesAction brewed', 'tea HasA caffeine', 'tea IsA beverage', etc., from which the features are 'ReceivesAction brewed', 'HasA caffeine', and 'IsA beverage').

7.2.3 OpenIE Relations

We introduce features computed as open information extraction relation triples using OpenIE [282], motivated by our observation that better connected inter-sentence units ground other lexical or conceptual information units between the question or correct answer or the explanation facts. We use a total of 36,989 total features derived from this feature group. Let us illustrate with an example:

Question Which of the following properties provides the BEST way to identify a mineral?
Answer Hardness
Explanation
hardness is a property of a material or an object and includes ordered values of malleable or rigid;

In the example, the given fact is top-ranked in the explanation. For it, from OpenIE we get the relation triple (hardness → is a property of → material). Further, ConceptNet tells us that the answer Hardness is related to concepts "property," "material property," etc. We see how pooling these information units together enables a coherent word cloud involving the question, correct answer, and explanation fact for the terms "hardness," "property," and "material." Features that enable grounding externally computed terms to the lexical items given in the QA pair or explanation facts create a tighter overlap improving task performance. Given the potential usefulness of inter-sentence OpenIE triples for explanation generation, we create features as follows. For each of triple produced by the parser, the features are: the $q/ca/e$ lemmas in the relation subject role, shared q, ca, and e subject lemmas, $q/ca/e$ lemmas in the relation object role, shared q, ca, and e object lemmas, and $q/ca/e$ lemma as the relation predicate.

7.2.4 Multihop Inference Specific Features

These features are a more selective bag of lexical features for obtaining matches with a positional emphasis. We identify that adding positional information for lexical matches is a useful heuristic to identify the concepts that are the focus of the (q, ca) and explanation facts. For this feature category, we collected 2,620 total features. Consider the underlined words in the two subsequent examples in this section.

As shown in the examples, often the focus word of the (q, ca) are at the start or end and also at the start and end of the e. Further, for one- or two-word ca, we can directly infer it as a focus concept, in

Question There are different types of desert. What do they all have in common?
Answer low rainfall
Explanation
a desert environment has low rainfall

Question Sonar helps people find which information about an object?
Answer Location
Explanation
sonar is used to find the location of an object;
the location of an object can be used to describe that object;

which case we try to find a match with e where they are the first or last word. And for focus words that are verbs, they tend to occur in the middle.

Length of q and ca; positions of q/ca verbs in the phrase (as 0 if it is the first word, 1 if it is the second word, and so on); of the verbs shared by q and e do they occur among the first few words or middle or last words.[2]; if ca is a uni- or bigram, does e contain all its words/lemmas?; does e contain the last q lemma/word?; is the last q lemma/word in the first position of e?, is it in the last position of e?; is the first q lemma/word in the first position of e?

7.2.5 TF-IDF Ranking

The explanation regeneration task performance via ranking based on cosine similarities between TF-IDF weighted (q, ca) appended text and each fact candidate proves surprisingly effective for the task (see scores in Evaluation section). We use the *TF-IDF Iterated* variant by [283] to encode the text. For this feature category, we employ 750,283 total features. The ranks obtained by cosine similarity on these instances are then used as features for the SVM learner. We hypothesise that employing the TF-IDF-based cosine similarity ranks as features will provide a baseline ordering signal to the learning algorithm. Our TF-IDF features per (q, ca, e) are the following: e's rank; e's binned rank in bins of 50; e's binned rank in bins of 100; whether e is in top 100 or 500 or 1000?

7.2.6 BERT Embeddings

BERT-based [32] context embeddings are our last features category. The out-of-box BERT model is pretrained on millions of words from Wikipedia which as a commonsense knowledge source is already pertinent to elementary science QA. Thus, we simply query the BERT embeddings from the pre-trained model using the bert-as-a-service library. Thus, for each data instance word, we extract their BERT embedding features that can easily can be combined with the other linguistic features. This can be viewed as a semantic projection of an elementary science concept in the Wikipedia encyclopedia space. Specifically, we query the BERT_{Base} UNCASED ENGLISH model: 12 layers, 768 hidden units, 12-heads, with 110M parameters that outputs a 768 dimensional vector for a given input text. We treat each dimension of this context vector as a separate feature for representing the instance. While the earlier five feature categories enabled extending the (q, ca, e) vocabulary beyond the given words both lexically and conceptually, with BERT embeddings we aim to leverage semantic abstractions as features. We hypothesise such features would be useful in creating semantic associations between the elements in

[2]For first, middle, and last words, using a window 1/4 the size of the total words, centered on the middle, we find the middle portion of the sentence, at its LHS, the first portion, and at the RHS, the last portion of the sentence.

the (q, ca, e) triple which are topically similar based on knowledge from Wikipedia. As in the following example.

Question Diamonds are formed when carbon is placed under extreme heat and pressure. This process occurs
Answer beneath the surface of Earth.
Explanation
the formation of rock is a kind of process;
diamond is a kind of mineral;
rock is made of minerals;
the formation of diamonds occurs beneath the surface of the Earth by carbon being heated and pressured

In the example, considering the focus words "diamonds," "earth," and "minerals" that reflect the topics of the QA pair, the word "minerals" in the fact is neither present in the q or ca, but is poignant to the semantic topic of the (q, ca). We hypothesise that BERT features will help capture such topicalised semantic abstractions of similarity. We tested two ways of obtaining BERT features for (q, ca, e) triples: i) query BERT separately for the question, correct answer, and fact embeddings, respectively, obtaining three 768 dimensional feature sets and resulting in 2,304 additional features from BERT per instance; and ii) query BERT for aggregate 768-dimensional embedding features for the (q, ca, e) triple. Experiments indicated that the latter method is a better-suited representation for the task while the former method is ineffective.

7.3 Knowledge Context Enhanced Support Vector Machines.

We add that all the six feature categories considered to represent (q, ca, e) triples, when taken together, should readily address the multi-step inference process between (q, ca) and e candidates. Since we have features extending the given information in the (q, ca) with world knowledge generically (e.g., ConceptNet, BERT) with other features providing lexical glue (at generic and task-specific levels) enabling traversing the (q, ca, e) via multiple hops (e.g., multihop inference lexical features, openIE relations). Therefore, for themes such as about vehicles, as an example from [284], describing its mechanisms, its purpose, its needs, and its functions, this information is easily available to the learning algorithm from our features group. Figure 7.2 we depict our overall approach including the feature modules and the two SVM-based machine learners we employed.

As described in Section 3, explanation regeneration for Elementary Science QA pairs is posited as a ranking task given a collection of candidate facts, where for each QA pair explanation, the number of valid facts can vary up to 21 and the desired result is to have all the valid facts top-ranked. Formally, let (q, ca, e) be a triplet consisting of a question q, its correct answer ca, and a candidate explanation fact e that is a valid or invalid candidate from the given unordered facts tablestore e_{uno}. Our task is, for each (q, ca) given e_{uno}, to rank the generated (q, ca, e) triplets such that the group (q, ca, e^c) is top-ranked to produce an ordered tablestore e_o, where e^c stands for the group of relevant facts in the explanation and $e^c \subseteq e_{uno}$. Within a preference-based object ordering formalism [285], the candidate facts e_{uno} comprise the reference set of objects. Training data consists of a set of rankings $\{O_1, ..., O_N\}$ of facts for N (q, ca) training instances, respectively, where for $(q, ca)_i$, the ordering is:

$$O_i : e_a^c \succ e_b^c \succ ... \succ e_g^c \tag{7.3.1}$$

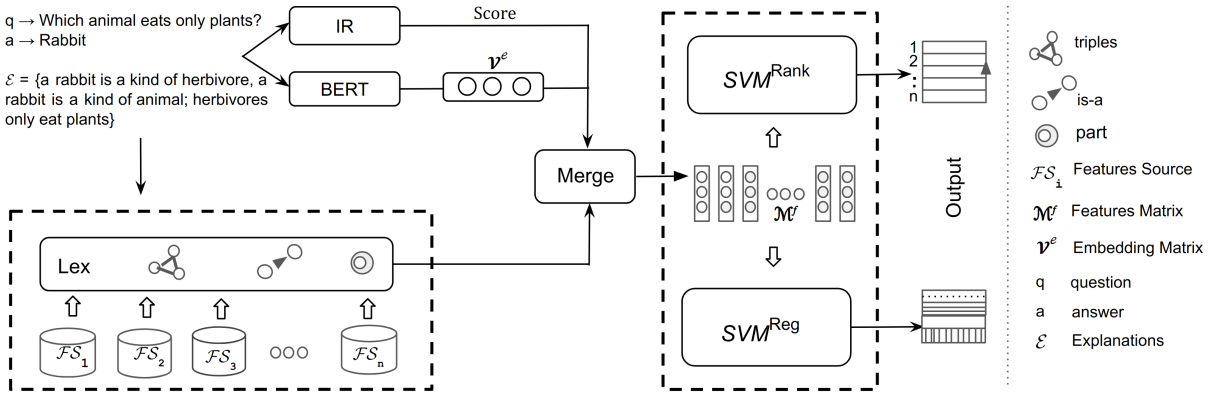


Figure 7.2: Overall representation of our approach. For representing (q, ca, e) triples, the feature categories used include Lex (lexical features), IR (information retrieval features), and BERT-based features among others. The data instances are then represented as a features matrix, separately as training data, development data, and testing data. Two variations of the SVM algorithm (SVM^{Rank} and SVM^{Reg}) are then used to learn Explanation Regeneration models.

such that O_i is an ordering of only the valid facts e_i^c for a $(q, ca)_i$ instance where $|O_i| < |e_{uno}|$. The order relation $\succ$ is interpreted in terms of preferences, i.e., $e_a \succ e_b$ suggests that e_a is preferred to e_b in terms of logical discourse. And the remaining $e_{uno} \setminus e_i^c$ are assigned a uniform least rank.

The next natural question is which functions do we choose to learn the set of orderings for (q, ca) pairs. In particular, two such approaches are prevailing in the literature. The first one reduces the original ordering problem to *regression*: it seeks a model that assigns appropriate scores to individual items and hence is referred to as the *pointwise* approach. The second idea reduces the problem to *binary classification*; the focus is on pairs of items, which is why the approach is also called the *pairwise* approach. Next, we briefly introduce these models in the context of the support vector machine (SVM) class of algorithms and describe how we train them. At a high-level, the objective of the SVM is to find the optimal separating hyperplane in an N-dimensional space (where "N" is the number of features) which maximises the margin of classification error on the training data. The margin is defined in terms of certain select training data points that influence the position and the orientation of the hyperplane such that it is at maximal separating distance from the data points in the various classes. These points then constitute the support vectors of the trained SVM. The support vectors lie on boundary lines that run parallel to the classification hyperplane but at the maximal computable distance. Obtaining a maximal margin produces a more generalisable classifier to unseen data instances. Note also that in real-world problems, the boundary lines are more practically considered soft boundaries with an error allowance defined by a slack variable ξ, that allows classifications to fall somewhere within the boundary margin from the classification hyperplane. Formally, as an optimisation problem, the SVM classification objective is to:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \\ \text{s.t.} \quad & y_i - w \cdot \phi(x_i) - b \leq \xi_i \\ & w \cdot \phi(x_i) + b - y_i \leq \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{7.3.2}$$

where $i = 1, ..., N$ for N training instances, ϕ is a feature transformation function for input x_i, w is the features' weight vector over all instances, and y_i is either +1 or -1. The constant $C > 0$ determines the

trade-off between the norm of the weight vector and error margin defined by slack variable ξ.

7.3.1 Pairwise Learning-to-Rank (LTR) for Preference Ordering

The next question is how can our preference ordering problem be formulated in terms of binary classification. This is possible by the pairwise LTR transformation. Vaguely, this is done by modeling: 1) whether a candidate fact is a valid candidate or not; and 2) for the collection of valid explanation facts, the logical precedence of one fact over another. Thus, these decisions are identified in a relative sense, that is to say, by determining the pairwise preferences between facts in the explanation compared w.r.t. each other and w.r.t. the remaining facts in the tablestore.
Our dataset originally is:

$$S = \{x_{ij}, y_{ij}\} \text{ where } x_{ij} = \phi((q_i, ca_i), e_j)$$

(q_i, ca_i) is the i-th QA pair instance, e_j is the j-th explanation fact from the tablestore where the ordering between facts is known during training and is unknown during development and testing. ϕ is a feature transformation function, and $y_{ij} \in \{1, 2, 3, ...K\}$ denotes an order between the (q_i, ca_i) pair and the explanation fact e_j as a graded order w.r.t. the other relevant and irrelevant explanation fact candidates. By the pairwise LTR transformation, our original dataset S then becomes:

$$S\text{'} = \{(x_{ij} - x_{il}), (y_{ij}\theta y_{il})\}$$

where θ is the rank difference so that $(y_{ij}\theta y_{il}) = 1$ if x > y and -1 else, resulting as a binary classification task. The goal of the LTR algorithm is to acquire a ranker that minimizes the number of violations of pairwise rankings provided in the training set which is attempted as the above classification problem. Essentially, since pairwise LTR only considers the labels where $y_{ij} > y_{il}$ or $y_{il} > y_{ij}$ between relevant candidates and between relevant and irrelevant candidate pairs, while transforming S to S ', the relevance between the (q, ca) and the correct candidate explanation facts must be indicated as graded relevance, while all the incorrect candidates are relegated to a uniform least rank. This is done as follows. A training instance x_{ij} gets label y_{ij} in a descending rank order starting at $rank$ = number of valid explanation facts+ 1 for the first fact and ending at $rank = 2$ for the last relevant fact in the explanation sequence, if x_{ij} corresponds to $\phi((q_i, a_i), e_j)$ with e_j as a correct explanation fact; otherwise, the uniform least $rank = 1$ if e_j is an irrelevant explanation candidate.

Training LTR for QA Pair Explanation Fact(s) Preference Ordering

We use the SVM LTR learning algorithm as implemented in the SVM^{rank} software package [286]. To optimise ranker performance, we tune the regularisation parameter C (which establishes the balance between generalising and overfitting the ranker model to the training data). However, we noticed that a ranker trained on the entire tablestore set of facts is not able to learn a meaningful discriminative model at all owing to the large bias in the negative examples outweighing the positive examples (consider that the number of relevant explanation facts range between 1 to 21 whereas there are 4,789 available candidate facts in the tablestore). To overcome the class imbalance, we tune an additional parameter: the number of negative facts for training. Every (q, ca) training instance is assigned 1000 randomly selected irrelevant explanation facts. We then tune the selection of the number of irrelevant explanation facts ranging between 500 to 1,000 in increments of 100.

Both the regularisation parameter and the number of negative explanation facts are tuned to maximise performance on development data. Note, however, that our development data is created to emulate the

testing scenario. So every (q, ca) instance during development is given all 4,789 facts to obtain results for the overall ordering task.

7.3.2 Pointwise Preference Ordering by Regression

SVM regression differs from the SVM classification objective in that instead of optimising over binary targets, the optimisation is performed for real-valued targets. To facilitate this, regression is then defined in terms of an ϵ-precision objective. In other words, we do not care about training errors as long as they are less than ϵ. Further, as in the classification objective with soft decision boundaries, similar allowances are made with slack variables in the regression context, but defined over targeted regression precision. Formally, the regression optimisation problem is defined as follows:

$$\begin{aligned} \min_{w,b,\xi,\xi^*} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - w \cdot \phi(x_i) - b \leq \epsilon + \xi_i \\ & w \cdot \phi(x_i) + b - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0 \end{aligned} \tag{7.3.3}$$

where, $i = 1, ..., N$ for N training instances, ϕ is a feature transformation function for input x_i, w is the features' weight vector over all instances, and y_i is a real-valued target, and ϵ is the regression targeted precision. The constant $C > 0$ determines the trade-off between the norm of the weight vector and error margin defined by slack variables ξ, ξ^*.

Next, an important question is, how to represent our ordering problem in terms of a regression objective. We do this by defining regression targets in terms of the preference ordering expectations [279, 287] rather than true regression quantification. In our dataset $S = \{x_{ij}, y_{ij}\}$ where $x_{ij} = \phi((q_i, ca_i), e_j)$, the label y_{ij} for the correct candidate explanation facts are indicated as unit graded relevance in order of their preference, while all the incorrect candidates are relegated to a uniform least rank. This is done similarly to the pairwise LTR setting. More specifically, given an explanation ordering O_i of facts of length n_i for (q_i, ca_i), an instance x_{im} for a valid fact e_m ranked on position r in O_i is assigned the score $y_{im} = n_i + 2 - r$. This is justified by assuming a uniform distribution of ranks. Roughly speaking, facts in explanation orderings are assumed to be distributed uniformly among the whole spectrum of explanations. All facts not in O_i paired with (q_i, ca_i) are assigned $y_i = 1$. With testing the regression formulation for the preference ordering of facts in explanations, we make the assumption that all facts can be treated independently w.r.t. each other. Such assumptions are highly contingent on the properties of the underlying dataset and may not apply in all preference ordering or ranking scenarios. In contrast, the pairwise LTR is, in principle, applicable in any ordering scenario.

Training SVR for QA Pair Explanation Fact(s) Preference Ordering We use the SVR learning algorithm [288] as implemented in the SVM^{light} software package [289] (hence called SVM^{reg} since we employ its regression setting). Similar to the ranker system, to optimise regression performance, we tune the regularisation parameter C on the development set with all the other parameters at their default values. Again like in the ranking training setup, we randomly select a smaller set of irrelevant explanation facts to learn a meaningful discriminative model which are tuned on the development set to range between 500 to 1,000 in increments of 100. Note that our development data is created as usual to emulate the testing scenario given e_{uno}. So every QA pair instance during development is given all 4,789 candidate facts for regression predictions.

7.4 Finetuning BERT with Focus Words

The core task in explanation regeneration essentially entails two main steps: the identification of relevant explanation facts from a given knowledge base, followed by ranking the selected facts as a logically coherent paragraph. Figure 7.3 shows an example data instance from the WorldTree corpus [272] that defines this task. It is basically a question, its correct answer, and a set of ordered facts that justify the correct answer choice. Depicted in the figure, as a subgraph, is a crucial characteristic feature of the data: that there are lexical overlaps between the question, the correct answer, and explanation facts. In this respect, however, there are two notable caveats: 1) distractors—the lexical overlaps can also exist with irrelevant facts to the QA. E.g., given KB fact *a decomposer is usually a bacterium or fungus*, it has a lexical match to the answer, but it is not relevant to the explanation. Similarly, at least 13 other such matching irrelevant facts can be found in the WorldTree corpus [272] knowledge base. And 2) multi-hop inference of valid explanation facts—not all the relevant explanation facts have a direct lexical match to the QA pair, some of the facts are lexically connected to the other valid explanation facts. E.g., the fact *a plant is a kind of an organism* has no lexical relation to the question or to the answer but to the first explanation fact, hence this entails multihop inference from the QA to the explanation fact to another explanation fact. As such, selecting the set of relevant explanation facts, demands extra effort beyond direct lexical matches with the QA.

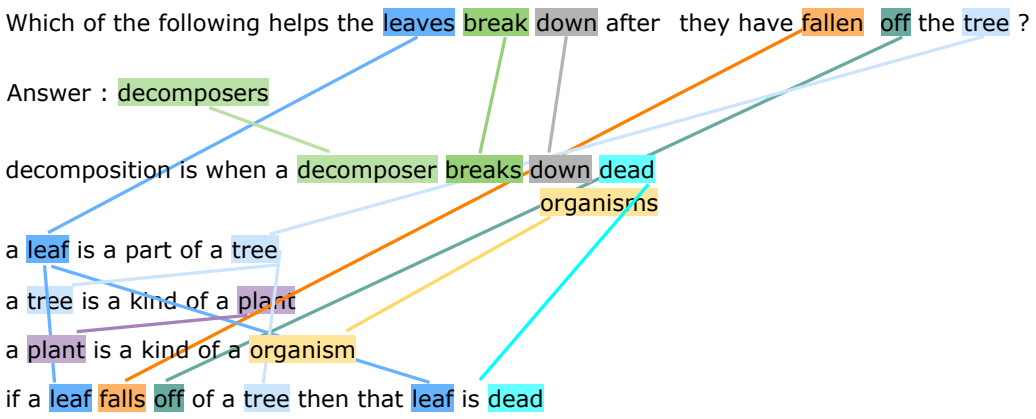


Figure 7.3: A elementary science question, its correct answer, and the ordered set of justification facts for the answer in the WorldTree corpus [272] depicted as a subgraph of lexical matches.

In light of these caveats in the data, the task presents itself as a fairly complex inference task, where traditional methods for QA that are based on simple fact matching have proved inadequate [284, 290]. Indeed on this task, the pure *tf-idf* baseline performs poorly (28% *mAP*). The system by Chia, Witteveen and Andrews employed an improved form of the *tf-idf* algorithm to achieve over 14 points improvement on the baseline. D'Souza, Mulang and Auer hand-craft a set of linguistic features to train a Support Vector Machine (*SVM*) for ranking. On the other hand, this traditional model is competed by the vast amount of research concerning machine reading comprehension and vector space representation of text [32, 52] that rely on neural attention mechanisms. These models commonly referred to as *"Pretrained Transformers"* are capable of demonstrating state-of-the-art results on multiple flavors of downstream tasks such as "Question Answering," "Entity Linking and Disambiguation" on different ends of the spectrum of IE tasks. In the TextGraph-13 MIER task, Das et al. employed a fine-tuned BERT-based model as the basis for their approach, achieving SOTA performance. However, the performance of any fine-tuned model depends on how related the data is to the original pretraining data and how best the input representation

is formulated. In this work we employ a fine-tuned transformer model by augmenting our input with focus word information to enhance the attention capability of the model and report an improvement on the pure transformer model. Specifically, we investigate two research questions: RQ1: What is the effect of number of negative training examples on model performance? RQ2: What is the impact of refocusing during fine-tuning on model performance?.

7.4.1 Approach: Encoding Focus Words in BERT

Our approach consists of two main sub-tasks as illustrated in figure 7.4. In this section we expound on the details of each sub-task.

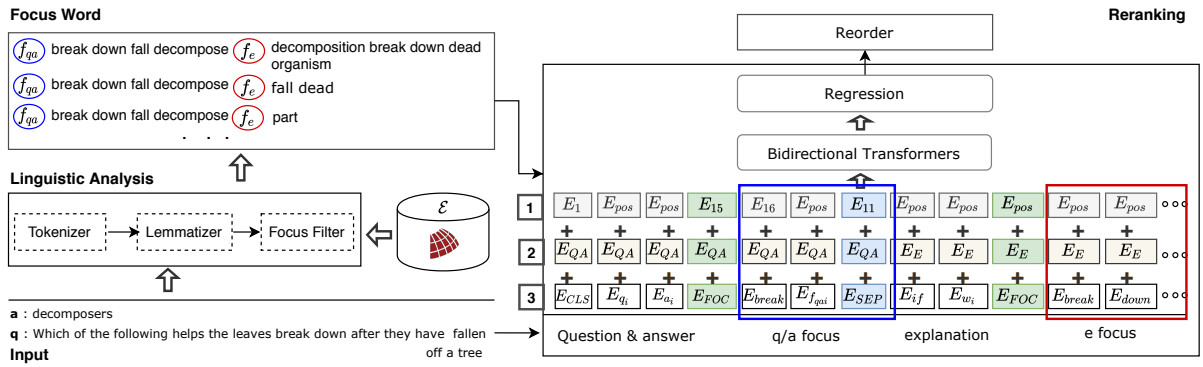


Figure 7.4: Approach: Refocused Fine-tuning of Transformers. The q,a are paired with each explanation in the set. This is then passed through the Focus word extractor that identifies the focus words based on concreteness scores. Input representation layers: 1) Position Embeddings. 2) Segment Embeddings. 3) Word Embeddings.

Linguistic Analysis

The word concreteness rating scores [77] forms a good source of information to determine the semantic weight of a term in a sentence. Jansen et al. jansen2017framing employ the word concreteness scores to determine the focus words in the question and explanations. This in turn is used as the pivot to aggregate relevant explanation sentences. We borrow this insight and apply these concreteness scores during fine-tuning process. The ratings reference list contains lemma forms of 40,000 generally known English words. Words with a concreteness rating below 3.0 are abstract words while those with a rating score above 4.2 are concrete words, and any word with score in between is considered focus words.

The intuition behind our approach is that these words, if used to augment the input, refocuses the model on the most relevant terms. We use the spaCy[3] NLP library for tokenization and lemmatization of the text before retrieving concreteness scores from the [77] list.

Reranking

We utilise the pretrained BERT [32] model and fine tune it on the sentence pair scoring task with a regression score function. For each input sentence, we append the special tokens $[CLS]$ and $[SEP]$ to the beginning and end of the sentence, respectively. Additionally (for the refocused model), we introduce one special token $[FOC]$ to separate the focus words segment from the main text. The sample input for our running example is represented as:

[3] Available from https://spacy.io/

```
[CLS] Which of the following helps leaves break down after they have
fallen off a tree decomposers.
        [FOC] break fall decompose [SEP] decomposition is when a
        decomposer breaks down dead organisms.
        [FOC] decomposition decompose break down organism [SEP].
```

This is then used as input to the model which learns representations for both the tokens and the sentences (loosely used here to refer to the question-answer pair and the explanation sentence). Figure 7.4 shows how the input representations are handled at the embeddings layer. for instance to obtain a representation for a focus word at position i in the input, from the q-a side: the word embedding E_{qai} - layer 3, segment embedding E_{QA} - layer 2 and the position embedding E_{pos_i} - layer 1; are summed up into a single embedding vector. This output is passed to the Bidirectional Transformer layer and finally through a regression layer to produce the score γ for final reordering. The reordering is a sort algorithm that assigns explanations with higher γ values lower position in descending order.

Training and Hyperparameters

Our BERT model is initialised using publicly available weights from the pretrained BERT_{BASE} model available in the Python package Pytorch-Transformers[4]. We use the default learning rate of 2e -5, a batch size of 32 and maximum sequence length of 512. This batch size and the sequence length stay the same both for training and testing. The model was fine-tuned for 3 epochs using Adam optimiser [135].

7.5 Experimental Setup

Dataset As mentioned earlier, the experimental corpus employed in this study is the Worldtree corpus [272], which is a subset of the ARC QA corpus [281] extended with explanations for QA pair. We relied on the provided corpus partitions comprising 1,190 and 264 elementary science QA pairs for system training and development, respectively. For testing, we used the 1,247 instances released as test data. As explanation candidates, we used the tablestore of the 4,789 facts which are the same candidates for training, development, and test data.

Evaluation Metrics We report one set of results in terms of the mean average precision *mAP* metric which is a standard in IR ranking tasks. With the *mAP* metric score we see to what extent our system returns the relevant explanation facts as top-ranked. To evaluate our system for ordering the relevant explanation facts w.r.t. each other, we employ the Precision@k and Recall@k metrics, where k is the group of top-ranked facts ranging between 2 to 50 facts in increments of 2. With these latter evaluations, we see how the top-ranked facts returned by our system are ordered w.r.t. each other satisfying the logical ordering of facts in the WorldTree corpus [272].

Parameter Tuning For the SVM^{rank} and the SVM^{reg} systems described in section 7.2, we jointly tune the C and the number of negative training instances parameters on development data. Our best SVM^{rank} model when evaluated on development data was obtained with C = 0.8 and 1000 negative training instances, while our best SVM^{reg} model was obtained with C = 0.005 and 900 negative training

[4] Accessible at https://github.com/huggingface/transformers

instances.[5] On the other hand, for the fine-tuned BERT with focus word as knowledge context approach in section 7.4, we initialised the BERT model using publicly available weights from the pretrained BERT_{BASE} model available in the Python package Pytorch-Transformers[6]. We use the default learning rate of 2e -5, a batch size of 32 and maximum sequence length of 512. This batch size and the sequence length stay the same both for training and testing. The model was fine-tuned for 3 epochs using Adam optimiser [135].

We compare our models with nine existing systems as reference performances (see section 7.5.1), where the systems we compare with have varying degrees of complexity from simple IR approaches to neural-based machine learning approaches. Further, as we show in the results eventually, the systems also have varying degrees of performances not necessarily correlated with the system complexity. For instance TF-IDF approaches prove surprisingly effective on this task. In the following section, the nine systems we evaluate against are briefly described.

7.5.1 Reference Baselines for Evaluations

TF-IDF Baseline Facts are ranked by cosine similarity of their TF-IDF representation with the TF-IDF representation of the query string composed of the question and all the available answer choices.

TF-IDF Baseline features + SVM^{rank} [74] Facts are ranked by SVM^{rank} [286] using the *TF-IDF Baseline* output for each (q, ca, e) as feature representations.

Generic Feature-rich SVM^{rank} [277] In this system, 76 features categories including OpenIE [282], ConceptNet [47], Wiktionary, and FrameNet [292] representations for each (q, ca, e) triple are employed, which are then ranked by SVM^{rank} [286].

Rules + Generic Feature-rich SVM^{rank} [277] In this hybrid model, the *Generic Feature-rich SVM^{rank}* system output is corrected for obvious errors by a set of 11 re-ranking rules applied sequentially, pipelined to the SVM system output. As an example of a rule consider: all facts that contain the bigram or unigram correct answer word are to be top-ranked.

BERT Iterative Re-ranking [293] The system models explanation regeneration using a re-ranking paradigm, where BERT [32] transformer models are used to provide an initial ranking, and the top 15 facts output by the BERT model are re-ranked using a custom-designed relevance ranker to improve overall performance.

Optimised TF-IDF [283] This system differs from *TF-IDF Baseline* in the following ways: all incorrect answer choices are dropped from the query string; the query and the fact strings are additionally preprocessed by lemmatization and the removal of their stopwords.

Iterated TF-IDF [283] Where in the *Optimised TF-IDF* system, the query string consists of only the question and the correct answer, in this system, the query string is iteratively expanded to include the top-ranked fact. After each expansion step, cosine similarity is rerun on the remaining facts to obtain the next top-ranked fact. This process is iteratively repeated until all facts are ranked.

[5] For parameter tuning, C is chosen from the set {0.005,0.05,0.1, 1,10,50,100} and the number of negative training instances is chosen from the set {500,600,700,800,900,1000}.

[6] Accessible at https://github.com/huggingface/transformers

BERT Re-ranking with Iterated TF-IDF scores [283] A BERT regression module is trained to predict the relevance score for each $(q + ca, e)$ pair, where the relevance score is the *Iterated TF-IDF* system rankings; and in the interest of lowering the computational complexity and runtime, the model is trained and tested to rerank only the top 64 of the *Iterated TF-IDF* system output.

BERT Re-ranking with inference chains [291] It is an ensemble model composed of BERT-based path ranker and a more advanced reranking system [294] which they employ as a ranker. The BERT-based path ranker uses a sophisticated multi-step design. The initial step involves obtaining the top 50 facts based on TF-IDF similarity with (q, ca) query. In the next step, 1-hop lexical similarity paths are traced from each fact in the retrieved 50 facts list to the remaining facts in the tablestore. Finally, the BERT path ranker is trained on pairwise fact instances. Instances are formed by exhaustively pairing each fact in the top 50 TF-IDF list with all the corresponding retrieved facts at a 1-hop lexical distance from it such that the pairs where both facts constitute the explanation for the given query are a true instance for the BERT path ranker, and others are false. The overall ensemble system then relies on this BERT-based path ranker output for a score threshold of above 0.5, else it uses a reranker [294].

With its chaining of facts, this system models a vital aspect of the corpus: i.e., some valid explanation facts directly lexically overlap with the QA pair, and others lexically overlap with other valid facts. As we will see next, this system has the overall best performance, and is the only system we do not outperform. We note, however, that it also presents a high degree of computational complexity. In our system, during training, each QA pair is linked with only roughly 1000 explanation facts; whereas the team4 system, would construct training instances as follows: for each QA pair, given top 50 ranked facts, assuming each fact has a 1-hop chain to at most 200 other facts, this would result in nearly 10,000 instances. A larger training dataset generally implies a larger training time, a fact that is particularly true in the case of BERT models as the team4 system, while also true for SVMs, although in the SVM case, the number of features would also matter. Further, in the test scenario, while the team4 system would still evaluate for 10,000+ odd chains, we would merely check for all the facts in tablestore which presently is about 5,000. Thus, the [283] system is the most effective and at the same time the most computationally intensive of all the systems for this task including ours.

7.6 Results and Discussion

Table 7.5 shows the elementary science QA pair explanation fact preference ordering results in terms of *mAP* with best results from the reference system and two implementations (last two paired rows of results) in bold. **For the feature-rich SVM models:** we find that SVM^{reg} is significantly better[7] than SVM^{rank} by applying the paired t-test to their adjacent scores in the table. Thus, given our underlying dataset, a pointwise learning approach proves better suited to it than a pairwise learning approach. Nevertheless, the latter is still a valid model, in principle, for the task as it does not rely on the strong, and seeming unrealistic, independence assumption between instances made by the SVM^{reg} model. However, since SVM^{reg} significantly outperforms SVM^{rank} at $p < 0.05$, it proves practically better suited on this dataset implying that the non-independence assumption between facts by SVM^{rank} is not a crucial factor in learning the task defined in the data.

Compared with the nine reference systems, our SVM^{reg} approach significantly outperforms eight of the models. This set of systems also includes the neural ranking model by [293], which our system surpasses by +9.4/+10.9 points in *mAP*; as well as a neural re-ranking model by [283] which we surpass by +3 *mAP*. Although we observe lower performance when compared to the best-performing approach

[7] Unless otherwise stated, all statistical significance tests are paired t-tests with $p < 0.05$.

(-5.6/-5.3) by [291], theirs is a significantly computationally complex system than ours as explained earlier (Section 7.2). Finally, in terms of scalability, next to our feature-rich SVM^{reg} are the *Iterated TF-IDF* or *Optimised TF-IDF* models by [283]. Where *Optimised TF-IDF* is far simpler of the two than our model, nonetheless, is significantly underperforming, however its ranking output as features in our system proves effective as we will see in the ablation analysis results (table 7.7). With our re-engineered system leveraging domain-targeted features, we have significantly outperformed our earlier system that was based on generic linguistic features [277]. Our SVM^{rank} is at +9.2/+8.8 compared to the system without rules and at +3.9/+1.5 compared to the hybrid system; and our SVM^{reg} is at +16.6/+16.1 to the without-rules system and +11.3/+8.8 to the hybrid system. Thus, we see in contrast the task impact obtained from an effective learning algorithm and a set of features that specifically models the domain in our new system version.

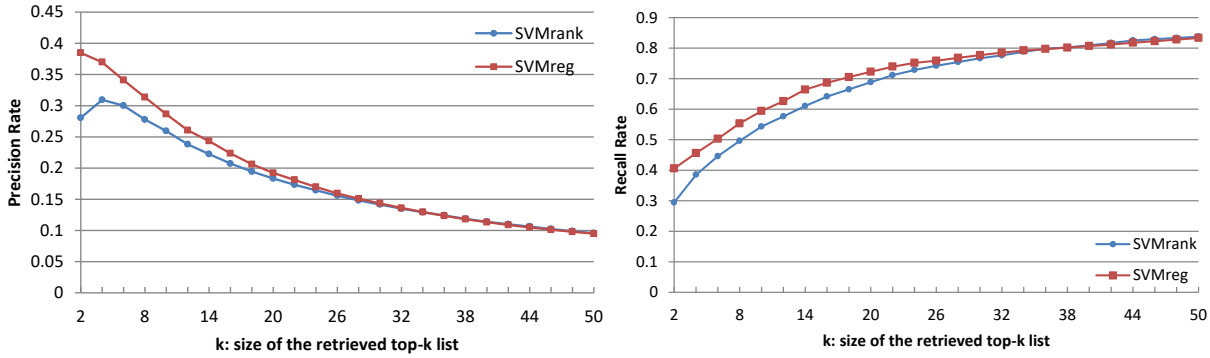
	Approach	***mAP***	
		Test	Dev
1	BERT Re-ranking with inference chains [291]	**56.3**	**58.5**
2	BERT Re-ranking with Iterated TF-IDF scores [283]	47.7	50.9
3	Iterated TF-IDF [283]	45.8	49.7
4	Optimised TF-IDF [283]	42.7	45.8
5	BERT iterative re-ranking [293]	41.3	42.3
6	Rules + Generic Feature-rich SVM^{rank} [277]	39.4	44.4
7	Generic Feature-rich SVM^{rank} [277]	34.1	37.1
8	TF-IDF Baseline features + SVM^{rank} [74]	29.6	–
9	TF-IDF Baseline	24.8	24.4
	Knowledge Context Based Linguistic Features NLE		
	Targeted Feature-rich SVM^{rank} (Ours)	43.3	45.9
	Targeted Feature-rich SVM^{reg} (Ours)	**50.7**	**53.2**
	Knowledge Context for Bert Finetuning [273]		
	Plain BERT + Optimised Neg Example (Ours)	**54.1**	**52.6**
	Refocused BERT + Optimised Neg Example (Ours)	**55.6**	**53.8**

Table 7.5: Mean Average Precision (*mAP*) percentage scores for Elementary Science Q&CA Explanation Regeneration by our systems (last two rows) compared with nine reference systems on testing (Test) and development (Dev) datasets, respectively.

The Fined BERT model with focus words: addresses two aspects concerning the number of negative examples in the training set, and the effect of the focus words feature in performance of the transformer model. for the first question, we experiment with different number of negative examples starting with the whole dataset containing approx. 4,770 negative explanation per question. Table 7.6 shows that too many negative examples for training had a negative impact. We reach an equilibrium between 600 and 900 negative explanation sentence per question. Table 7.6 shows the performance of the best configuration of number of negative explanation sentences per question (900) trained with refocusing against the plain BERT model. There is an improvement of over 1 percentage *mAP* in both Dev and Test sets. We ran these experiments 3 times and each version of experimentation settles at these numbers ±0.2. Likewise, we compare our results against 9 different baseline systems that were submitted at the TextGraph-13

workshop [74]. Our results are second to the best performing system from [291] who employ BERT as a part of a chains of reasoning system. We view that our single model approach performs competitively.

As a summary statement for our results presented in this section, we see that at 50.7% *mAP* of ranking the valid facts as top-ranked, only a small proportion of the predictions are in exact order based on the gold standard.



(a) (b)

Figure 7.5: Fact ordering evaluations of our system for top-ranked explanation facts in terms of Precision Rate @ k (subfig. a) and Recall Rate @ k (subfig. b) on the test data.

Feature Ablation Results

To provide further insights on the impact of adding different feature groups, we show ablation analysis results in table 7.7. Our ablation analysis strategy is to append each of the six feature groups, one at a

#Neg. Examples	Dev *mAP*	Test *mAP*
~ 4770	43.21	40.11
1000	53.12	50.42
900	54.14	**52.57**
800	54.88	52.26
600	54.88	52.26

Table 7.6: Comparing Mean Average Precision (mAP) percentage scores of fine-tuned BERT model based on number of negative examples in training set.

	Feature Type	SVM^{rank} *mAP* Dev	SVM^{rank} *mAP* Test	SVM^{reg} *mAP* Dev	SVM^{reg} *mAP* Test
1	Bag_of_Lex	34.01	30.44	39.57	37.43
2	Bag_of_Lex+ConceptNet	36.10	32.72	41.47	39.10
3	Bag_of_Lex+OpenIE	32.61	30.59	39.83	37.12
4	Bag_of_Lex+Multihop	35.33	32.46	41.90	39.43
5	Bag_of_Lex+TF-IDF	40.53	38.18	51.24	46.99
6	Bag_of_Lex+BERT	40.90	39.89	47.89	46.93

Table 7.7: Ablation Results of SVM^{rank} and SVM^{reg} on the dev and test sets, respectively, in terms of percentage *mAP* with feature groups (from six feature types considered) added one at a time to the bag of lexical features. TF-IDF and BERT features have highest impact.

time, to the baseline features as individual ablation experiments.

From the reported scores in the table, we observe that on both the SVM^{rank} and SVM^{reg} learners, respectively, that TF-IDF and BERT features show highest impact; multihop and ConceptNet features were the second most impactful feature types. And the least impact was from the OpenIE features which showed no improvement with SVM^{reg} and a minor improvement with SVM^{rank}. Nonetheless, we retain this feature group since its ablation analysis doesn't show a negative impact on system performance.

A qualitative examination of our results in the light of ConceptNet features shows that its added commonsense world knowledge prevented semantic drift in several cases. We demonstrate this with the help of one selected example below. In it, with additional knowledge such as that "plant" has a ConceptNet class "photosynthetic organism" enables achieving a higher ranking for the 2nd and 3rd explanation facts since one of the focus concepts in the question is "photosynthesis."[8]

Question In which part of a tree does photosynthesis most likely take place?
Answer leaves
Before
[r_p 1 & r_g 1] a leaf performs photosynthesis or gas exchange
[r_p 5 & r_g 2] a leaf is a part of a green plant
[r_p 10 & r_g 3] a tree is a kind of plant
After
[r_p 1 & r_g 1] a leaf performs photosynthesis or gas exchange
[r_p 3 & r_g 2] a leaf is a part of a green plant
[r_p 7 & r_g 3] a tree is a kind of plant

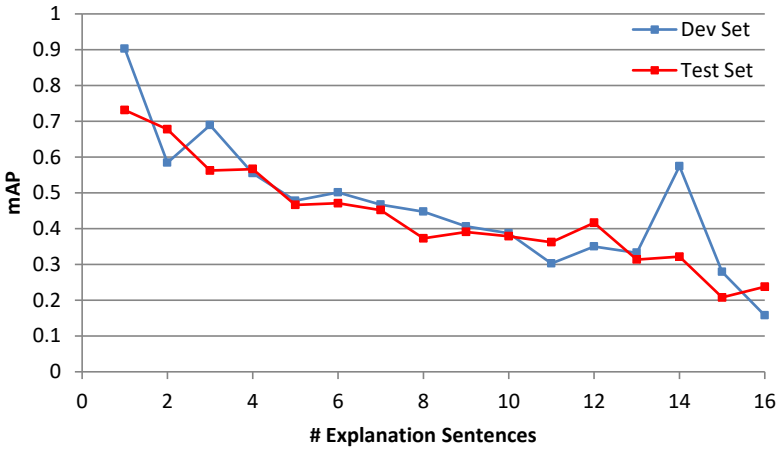


Figure 7.6: Percentage *mAP* of the SVM^{reg} system on Development data (in blue) and Test data (in red), respectively, on different length explanation sentences.

And with the help of the following example, we qualitatively depict the impact of adding BERT features. We glean the theme of the QA pair as "falling under gravity." While the dotted phrases "gravitational force," "fall," and "falling" encompass the theme, they are not directly present in the (q, ca) pair unlike the underlined phrases. The third fact, i.e. "come down is similar to falling," that contains one of the (q, ca) absent thematic phrases, viz. "falling," after adding BERT features, attains a significant performance boost by 7 ranks. We posit this is by the better abstract theme modeled by BERT features.

[8] r_p is the predicted rank and r_g *3* is the gold rank

Question If you bounce a rubber ball on the floor, it goes up and then comes down. What causes the ball to come down?
Answer gravity
Before
[r_p 2 & r_g 1] gravity or gravitational force causes objects that have mass or substances to be pulled down or to fall on a planet
[r_p 1 & r_g 2] a ball is a kind of object
[r_p 12 & r_g 3] come down is similar to falling
After
[r_p 2 & r_g 1] gravity or gravitational force causes objects that have mass or substances to be pulled down or to fall on a planet
[r_p 1 & r_g 2] a ball is a kind of object
[r_p 5 & r_g 3] come down is similar to falling

7.7 Summary

In this chapter, we have investigated two direction of applying knowledge context for the Explanation Regeneration task. i) The first direction includes deriving a rich amount of features from several knowledge sources including commonsense knowledge bases (KBs) as well as term graphs and triple KBs (cf. section 7.3). Since the target is to provide a unified representation for the (q, ca) and the e/f to enhance the machine learning algorithm (SVM), this approach is likened to the work described in chapter 4. ii) The second approach tackles the question of representing a specific knowledge context feature derived from text, that describe their relative weight in the overall semantic context. The focus word feature used in section 7.4 is induced into a bidirectional transformer model (BERT [32] in this case). This application borrows from insights learned from section 5.3 and 6.1.2.

Our first approach 7.3 explores knowledge-rich features-based approach for preference ordering of facts to explain the correct answer to elementary science questions. With the goal of creating meaningful unification of (q, ca, e) triples, we have investigated six different feature categories targeted to the domain at hand at varying lexical and semantic information representations. Further, our evaluations of regression versus learning-to-rank machine learning systems for preference ordering offers a new observation of the applicability of pointwise versus pairwise approaches [278, 279, 285, 287]. Further, we have reported a detailed empirical analysis of our system performances on the task of explanation regeneration against nine existing reference systems. We have found that when provided with domain-targeted features, SVMs can outperform BERT-based neural approaches [293], however, the neural models applied in computationally complex task formulations far surpass the SVM performance. The trade-off then is computational complexity, implying very low practical viability, versus a performance compromise. It remains to say that designing features for SVMs requires linguistic insights of the practitioner to hand-craft features that model the dataset well and hence are avoided in light of recent performance boosts obtained from black-box neural models, but perhaps they merit reconsideration in some newer tasks.

Four our second implementation in section 7.4 we integrated knowledge context attained through Focus Words. We empirically determine that the number of negative examples in the training set has an impact on the fine-tuning process for the explanation regeneration task. Since fine-tuning is a transfer learning method, these models are heavily reliant on the original training data. In this sense, however powerful the bidirectional attention mechanism used, fine-tuning does not allow deeper attention focus on input data especially if the data is from a different domain. In this work we present an approach that targets

to overcome this limitation through refocusing where the input data is intelligently augmented with focus words to highlight points of attention. We observe a considerable improvement in performance. Subsequently we consider this as an avenue to explore especially in scenarios where pretraining is possible, or by training with simpler attention-based models.

CHAPTER 8

Conclusion

Linking of elements in text to referent knowledge bases (also referred to as disambiguation) is a fundamental step in Natural Language Understanding (NLU). Aspects of textual communication that often require disambiguation include: named entities, relations, concepts, word senses, and topics etc. The NLP community has defined two major linking tasks for named entities (Entity Disambiguation/Linking) and relations (Relation Extraction/Linking). Researchers have worked on these two tasks for more than a decade, for example: the RE task begun as early as 1998 [149], being studied throughout the early years of the century [42], while the EL task picked up around 2010 [147]). Notwithstanding, these tasks still remain largely unsolved, evidenced by continued interest from the community, and the performance of the available systems [21, 22, 57, 200, 295, 296]. In chapter 3, we discussed the state-of-the-art literature concerning RL (section 3.1) and EL (section 3.2) and determined the challenges facing linking tasks and the opportunities available to assist tackle these challenges through knowledge context. Over the surveyed literature, there exists tangible evidence that applying extra signals derived from curated knowledge bases helps improve performance [28, 73]. Such curated knowledge bases have largely been enabled through the work on semantic web that allows description of data to facilitate machine-to-machine communication and automatic machine understanding and processing of information. Knowledge Graphs (KGs) are special forms of KBs that have seen tremendous increase in research interest, development, as well as applicability in NLP systems over the last decade. Chapter 2 offers an elaborate definition of KGs, the underlying techniques for their creation, and the methods for access. This rise in KGs can be attributed to their ability to represent world facts in simple triple-based assertions of the form <subject, predicate, object> (s,p,o). Despite the efforts in KG research, linking approaches appear to remain relatively oblivious of the mammoth opportunity afforded by these structured KGs. Table 3.1 summarises use of knowledge context in EL systems. We observe that these approaches attempt to bridge the performance gap in the linking tasks by increasing the algorithmic power of the solution. There is a recent breakthrough in deep learning for NLP that has led to powerful text representation approaches [31, 50] and language modelling approaches [32, 52].

8.1 Research Contributions

This thesis recognised an opportunity to harness the power of these models to assist in capturing knowledge context from KGs and learn latent features for the linking tasks. This therefore, laid the foundation for our overall research problem definition of the thesis as follows:

Research Problem Definition

How can knowledge context be leveraged to improve performance of entity and relation linking?

This thesis takes into consideration the utility value of semantics exposed by knowledge repositories by leveraging the power of machine learning to encode contextual information (knowledge context) to enhance performance of linking approaches. The work in this thesis unearths aspects of knowledge graphs and the power it possesses to influence models used in several NLP tasks. As such, some of the contributions are intended as first attempts in this direction and have already influenced research within the community. Our overall research question is apportioned into three major constituent sub-research questions. First we tackle the challenging problem of relation linking where we operate in the short text environment that targets end-to-end RL [11]. Our RL tool ReMatch [11] denotes in similar constructs, both the properties in a KG and the relations in a given question as comparable tuples, then leverages both synonyms and semantic similarity measures based on graph distances from the lexical knowledge base - Wordnet [46]. Chapter 4 documents the ReMatch approach [11] to relation linking. This was the first attempt to perform zero shot relation linking in short text environment (vis a vis Question Answering). Furthermore, we tackled relation linking with non labelled data and defined a systematic approach that can be replicated in several research endeavours. Additionally, we offered a tool that can be reused in Question Answering pipelines together with other tools. This approach and tool to leverage graph networks to unify representations of relations in text KGs for end-to-end relation Linking contributes towards our approach for addressing the first research question:

RQ1

How can we achieve a unified representation of both knowledge graph and textual relations to enhance similarity matching?

8.1.1 Impact and Research Influence of RQ1 Contributions

The following are key contributions of the work presented in chapter 4 to the overall research community and a note on the impact this work has had since it was published:

- *A pioneer baseline for Relation Linking:* Being the first work that attempted relation linking in short text scenario (in our case the Question Answering task), this work provided a concrete baseline for comparison for subsequent researcher. A number of our suggested future work has since now been tackled by other researchers in the community. The EARL [38] entity and relation linking tool extends the research in this direction by jointly linking the entities and relations in a question. table 8.1 shows the comparison made between ReMatch (our work in chapter 4) and the later tool EARL. Other researchers in [297] worked under the short text environment but assume that entities have been linked. They extend these ideas by introducing ontological reasoning for inference concerning the candidate entities. 8.1 documents the comparison with this work. Still other researchers
- *A tool for use in Question Answering Pipelines:* Since our work produced a Relation Linking tool that performs a vital task in the QA process, researchers have found this tool relevant to QA

RL Tool	Dataset	Precision	Recall	F-Measure
SIBKB [298]	QALD-5	0.27	0.34	0.29
ReMatch [11]		**0.36**	**0.39**	**0.37**
EARL [38]		0.17	0.21	0.19
EERL [297]		0.43	0.49	0.45
SIBKB [298]	QALD-7	0.33	0.35	0.34
ReMatch [11]		**0.35**	**0.38**	**0.37**
EARL [38]		0.30	0.31	0.30
EERL [297]		0.42	0.46	0.43
SIBKB [298]	LC-QuAD	0.15	0.18	0.16
ReMatch [11]		**0.18**	**0.20**	**0.19**
EARL [38]		0.20	0.25	0.21
EERL [297]		0.53	0.58	0.55

Table 8.1: Comparing performance of RL tools inspired by ReMatch. Figures obtained from [297]

pipelines. Singh et.al. [18] build a question answering system that dynamically select best fit components. They report that our tool (ReMatch) reported best results for relation linking when combined with most dynamically chosen pipelines.

- *An approach to overcome several relation linking challenges:* Operating in a no labelled data scenario, we are able to define an approach that intuitively identifies patterns in data. With such patterns, we then describe the natural language relations into a construct the can be comparable to similar representation from the KG.

Thereafter, we approached the entity linking task from the perspective of enhancing EL models using knowledge context. This emerged from our observation that Relation Linking highly depends on entities and identifying named entities can greatly improve RL. The same observation has been made by other researchers [38, 297] leading to tackling RL as a joint task together with EL. The Arjun approach (chapter 5) and its two implementations [79, 201] contribute towards addressing the second research question:

RQ2

How can KG context be effectively encoded in neural network architectures to improve Entity Disambiguation?

8.1.2 Impact and Research Influence of RQ2 Contributions

Chapter 5 describes our approach named: Arjun. The Arjun approach has 3 major aspects:

- *Encoding of Knowledge context for Neural Network models:* Both of our implementations of the Arjun approach discussed in section 5.2 and section 5.3 utilise a local derived knowledge graph that infuses either the entity aliases or descriptions from DBpedia and Wikidata. We then define an

encoding system empowered by the work on deep neural networks. Hence, the first implementation uses an attention based NN encoder-decoder [55] to capture features from KG labels and aliases of entities, while the later implementation employs a transformer-based finetuned model based on BERT [32].

- *Modular Approach to EL:* that allows study of different sub-tasks of the overall EL task. e.g. our transformer based implementation in section 5.3 envisions a 3-module system that specifically targets to study the behaviour of the candidate generation step. The candidate generation (CG) step of EL has receive little attention from the community, yet when we experimented in our work, we identified that our model improved performance when extra context is infused at this stage of EL process.

- *Pioneer work for EL on Wikidata:* Wikidata is emerging as a vital source of information due to it's constant update rates, as well as the open community publishing policy. However, for so long, the research community has not developed datasets targeting this KG nor utilising the knowledge represented. Our model in [201] is the first attempt at end-to-end EL based purely on data generated for the Wikidata KG. As such we layout numerous challenges that are uniquely exposed when performing EL in Wikidata.

After our research results for the Arjun approach [79, 201], we observed that the volume of information encapsulating different forms of context, has not been explored in the research community. Subsequently we set out to perform an evaluation study to uncover the insights about the depth and form of KG-based knowledge context. Our evaluation work in chapter 6 provide two major observations: i) that knowledge context can be generalised for many machine learning models (e.g. we observed improved performance for the transformer models: XLNet [52] and RoBERTa [53]), as well as the attention based model in DCA [28]. We also observed that knowledge context can be a trade-of for algorithm complexity. This is portrayed by the fact that a simple model such as the LSTM network performs better than the extremely intricate and resource intensive RoBERTa model without context. Our contributions derived from the evaluation described in chapter 6 and reported in [10] directly apply to our third research question:

RQ3

Can the effect of knowledge context be generalised for neural entity linking models?

Closely related to our generalisation approach in section 6.1 is the question of identifying the relevant forms of knowledge context for any model. We therefore further extend our evaluations in section 6.2 of chapter 6. In this second phase of our work, we look at information from KGs (in this case, the Wikidata KG) in 4 different forms: entity titles / labels, entity attributes (aliases and descriptions), entity ontological type (instance of), and triples. In cognisance that not all this information is relevant for any specific linking instance. for example in our motivating example 4 - *"Result of the second leg of the African Cup Winners Cup final at the National stadium on Friday: Arab Contractors - Egypt 4 Sodigraf Zaire0, halftime 2:0 Scorers: Aly Ashour 7', 56'(penalty), Mohamed Ouda 24' 73'.Contractors won 4-0 on aggregate"*. The mention *"Zaire"* refers to the entity `wd:Q` - *"Democratic Republic of the Congo"*. From the Wikidata KG, on the aliases of the entity has the form *"Zaire"*, therefore this is essentially the only required piece of information for disambiguation. We take a look at these pieces of information and how much impact to specific disambiguation scenario they have.

8.1.3 Impact and Research Influence of RQ3 & RQ4 Contributions

The last two research questions are futuristic in nature and we have addressed them only in a one direction. way in this thesis. This thesis has laid a foundation for a discussion in the research community not only on how to capture relevant information from KGs, but to design approaches for encoding this information for optimal use with machine learning models.

8.2 Limitations and Future Directions

This thesis presented our findings concerning avenues for representing knowledge context and combing these representations with the power exposed by machine learning algorithms. Although we achieved good success, and are able to demonstrate clear validation of our research questions, these contributions are meant to ignite a new research conversation. Likewise, there are few limitations of this research which have not been covered in the scope of the thesis. We therefore, view that our contributions presented are initial steps in a larger research agenda that will foster further research. A few of such directions are enumerated below:

- **Extension of data:** Our work on relation extraction can be extended in multiple directions. Firstly, there is need for more datasets in the community that encourage end-to-end RL. The dataset used in our work are derived from Question Answering datasets hence limiting the scope of our work. Second, the incorporation of multiple relations and entity from the text during linking is a direction that can still be explored.

- **Leveraging power of KG-Embeddings:** Attaining more expressive embeddings for entities and relations from the KG to be used as features in linking models. This can be achieved by extending the work from KG Completion [19, 299]. RECON [22] is a relation linking operating under the sentential RE variant [61] of relation extraction. Inspired by our work on leveraging knowledge context and work on KG embeddings. The RECON approach attains vector representations of entity attributes and source sentence. In a further step, RECON performs entity and relation embeddings ins separate spaces to enhance expressively. All embeddings are then combined in a Graph Attention Network (GNN) to classify the desired relation. They show that the knowledge context enable model significantly outperforms the SOTA models. This direction of KG embeddings is still at an early stage and more research is required to achieve ultimate potential.

- **Cross-KG Linking:** Our models have been evaluated on community datasets that inherently target one knowledge base or the other (e.g. the QALD datasets and LC-QUAD are targeted for the DBpedia; the AIDA-CoNNL, ACE2004, MSNBC, and AQUAINT on Wikipedia; and finally the ISTEX and the T-Rex data are purely based on Wikidata). We believe that models that can perform disambiguation across several different knowledge repositories would provide better performance while enabling robustness.

- **Cross-Lingual Linking:** Our models are based on the English language. This is partly because of the underlying datasets. However, use of a single language limits the reach of our models to other data since there is a large volume of data in other languages (e.g. German, French or even low resource languages like native African languages). Several knowledge Graphs such as Wikidata and DBpedia already incorporate multiple languages (e.g. in entity labels and descriptions). This multilingual attribute can assist future researchers to attain models that incorporate multilinguality.

- **Incorporation of inferential knowledge from KGs:** The power exhibited by semantic web representations such as OWL and RDF has not been fully harnessed. In our work, we have been able to evaluate features obtained from the directly modelled data. However these tools expose more power that include inference of new knowledge. The open question is therefore whether research can induce ontology reasoning approaches as a form of knowledge context?

- **Dynamic selection of knowledge context:** From our findings in chapter 6, we conclude that only part of the available information is relevant to specific tasks, dataset or even piece of text, depending on the granularity. It is however difficult to pre-select such information as we may not tell before hand what kind of entities and relations exist in text. In future work, it would be interesting to understand which triples negatively impact the context and how to select the "optimal choice of KG-triples context," considering we rely on the triple in the same order of the SPARQL endpoint returned results. All these pieces of information can be treated as single feature points in a Graph to allow use of graph algorithms in dropping irrelevant points. This dynamic selection can also be achieve via machine learning approaches such as those employed in the work by Singh et. al. [2, 18].

In summary, we view that this work will result in a new variety of models for NLU research and systems. This emerging direction is expected to trigger questions concerning representation of knowledge context and use of the same as features in AI. Immediate questions include how to incorporate more expressivity in models such that more semantics including ontological reasoning and KG literals are captured in models. In another direction, there is a need for models that can discriminate information captured from KGs so that only relevant and concise features are retrieved.

Bibliography

[1] S. Auer et al., "DBpedia: A Nucleus for a Web of Open Data", *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.* 2007 (cit. on pp. 1, 19, 41, 42, 44, 59, 64).

[2] K. Singh, *Towards Dynamic Composition of Question Answering Pipelines*, PhD thesis: Universitäts-und Landesbibliothek Bonn, 2019 (cit. on pp. 1, 40, 132).

[3] H. Altwaijry, S. Mehrotra and D. V. Kalashnikov, *Query: A framework for integrating entity resolution with query processing*, Proceedings of the VLDB Endowment **9** (2015) 120 (cit. on p. 1).

[4] A. Mohamed et al., *RDFFrames: Knowledge Graph Access for Machine Learning Tools*, arXiv preprint arXiv:2002.03614 (2020) (cit. on p. 1).

[5] A. Kiryakov et al., *Semantic annotation, indexing, and retrieval*, J. Web Sem. (2004) 49 (cit. on pp. 1, 42).

[6] L. Dietz, A. Kotov and E. Meij, "Utilizing knowledge graphs for text-centric information retrieval", *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018 1387 (cit. on p. 1).

[7] N. Steinmetz and H. Sack, "Semantic multimedia information retrieval based on contextual descriptions", *Extended Semantic Web Conference*, Springer, 2013 382 (cit. on pp. 1, 76, 77, 79).

[8] C. B. Jones et al., "Spatial information retrieval and geographical ontologies an overview of the SPIRIT project", *SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Finland*, 2002 (cit. on p. 1).

[9] P. Le and I. Titov, "Boosting Entity Linking Performance by Leveraging Unlabeled Documents", *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Association for Computational Linguistics, 2019 1935 (cit. on pp. 1, 3, 46, 88, 97, 100).

[10] I. O. Mulang' et al., "Evaluating the impact of knowledge graph context on entity disambiguation models", *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020 2157 (cit. on pp. 1, 43, 45, 46, 83, 91, 95, 100, 130).

[11] I. O. Mulang, K. Singh and F. Orlandi, "Matching natural language relations to knowledge graph properties for question answering", *Proceedings of the 13th International Conference on Semantic Systems*, 2017 89 (cit. on pp. 1, 51, 128, 129).

[12] S. Dill et al.,
"SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation",
Proceedings of the 12th international conference on World Wide Web, 2003 178 (cit. on p. 1).

[13] K. Höffner et al., *Survey on challenges of Question Answering in the Semantic Web*,
Semantic Web (2017) (cit. on p. 1).

[14] F. M. Suchanek, G. Kasneci and G. Weikum,
YAGO: A Large Ontology from Wikipedia and WordNet,
Web Semantics: Science, Services and Agents on the World Wide Web (2008) (cit. on p. 1).

[15] F. Mahdisoltani, J. Biega and F. M. Suchanek,
"YAGO3: A Knowledge Base from Multilingual Wikipedias",
CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings, 2015 (cit. on pp. 1, 44).

[16] K. D. Bollacker, R. P. Cook and P. Tufts,
"Freebase: A Shared Database of Structured General Human Knowledge", *AAAI 2007*, 2007
(cit. on pp. 1, 41, 64).

[17] D. Vrandecic, "Wikidata: a new platform for collaborative data collection",
Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume), 2012 1063 (cit. on pp. 1, 19, 41, 44, 64, 65, 84).

[18] K. Singh et al., "Why reinvent the wheel: Let's build question answering systems together",
Proceedings of the 2018 World Wide Web Conference, 2018 1247 (cit. on pp. 1, 42, 129, 132).

[19] A. Bordes et al., *Translating embeddings for modeling multi-relational data*,
Advances in neural information processing systems **26** (2013) 2787 (cit. on pp. 1, 9, 131).

[20] S. Wu, K. Fan and Q. Zhang, "Improving distantly supervised relation extraction with neural noise converter and conditional optimal selector",
Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019 7273 (cit. on p. 2).

[21] W. Zhou et al., "Nero: A neural rule grounding framework for label-efficient relation extraction",
Proceedings of The Web Conference 2020, 2020 2166 (cit. on pp. 2, 127).

[22] A. Bastos et al.,
"RECON: Relation Extraction using Knowledge Graph Context in a Graph Neural Network",
Proceedings of The Web Conference (WWW) (long papers)- to appear, 2021 :N/A
(cit. on pp. 2, 9, 47, 89, 127, 131).

[23] J. Mayfield, P. McNamee and C. Costello, "Language-Independent Named Entity Analysis Using Parallel Projection and Rule-Based Disambiguation",
Proceedings of the 6th Workshop on Balto-Slavic Natural Language Processing,
Association for Computational Linguistics, 2017 (cit. on p. 2).

[24] A. Sakor et al.,
"Old is gold: linguistic driven approach for entity and relation linking of short text",
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),
2019 2336 (cit. on pp. 2, 3, 43, 46, 61, 64, 66, 67, 69, 70, 72, 75, 80, 100).

[25] S. Cucerzan, "Large-scale named entity disambiguation based on Wikipedia data",
In Proc. 2007 Joint Conference on EMNLP and CNLL, 2007 708 (cit. on pp. 2, 39, 41, 97).

[26] K. Nebhi, "Named entity disambiguation using freebase and syntactic parsing", *Proceedings of the First International Workshop on Linked Data for Information Extraction (LD4IE 2013) co-located with the 12th International Semantic Web Conference (ISWC 2013)*, Gentile, AL; Zhang, Z.; d'Amato, C. & Paulheim, H., 2013 (cit. on p. 2).

[27] H. T. Nguyen and T. H. Cao, "Named entity disambiguation: A hybrid statistical and rule-based incremental approach", *Asian Semantic Web Conference*, Springer, 2008 420 (cit. on p. 2).

[28] X. Yang et al., "Learning Dynamic Context Augmentation for Global Entity Linking", *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 271 (cit. on pp. 2, 6, 9, 45–47, 61, 75, 83, 84, 87, 88, 91, 94–97, 100, 127, 130).

[29] I. Y. et al., "Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation", *CoNLL*, 2016 (cit. on pp. 2, 3, 9, 46, 88, 97).

[30] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation **9** (1997) 1735 (cit. on pp. 2, 30, 31).

[31] A. Vaswani et al., "Attention is all you need", *Advances in neural information processing systems*, 2017 5998 (cit. on pp. 2, 12, 37, 72, 94, 127).

[32] J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *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)*, 2019 4171 (cit. on pp. 2, 6, 9, 10, 12, 13, 37, 43, 72–75, 78, 94, 95, 111, 116, 117, 119, 124, 127, 130).

[33] X. Cheng and D. Roth, "Relational inference for wikification", *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013 1787 (cit. on pp. 2, 98).

[34] J. R. Raiman and O. M. Raiman, "Deeptype: multilingual entity linking by neural type system evolution", *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018 (cit. on pp. 2, 3, 9, 46, 64, 83, 88, 97, 98, 100).

[35] S. Vashishth et al., "RESIDE: Improving Distantly-Supervised Neural Relation Extraction using Side Information", *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, ed. by E. Riloff et al., Association for Computational Linguistics, 2018 1257 (cit. on pp. 2, 7, 40, 47, 83, 89, 100).

[36] P. N. Mendes et al., "DBpedia spotlight: shedding light on the web of documents", *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS 2011, Graz, Austria, September 7-9, 2011*, ACM, 2011 1 (cit. on pp. 3, 41, 42, 46, 76, 77, 79).

[37] N. Kolitsas, O.-E. Ganea and T. Hofmann, "End-to-End Neural Entity Linking", *Proceedings of the 22nd Conference on Computational Natural Language Learning*, 2018 519 (cit. on pp. 3, 6, 9, 43, 44, 46, 64, 70, 72, 73, 76–79, 81, 90).

[38] M. Dubey et al.,
"EARL: Joint Entity and Relation Linking for Question Answering over Knowledge Graphs",
ISWC 2018, 2018 (cit. on pp. 3, 128, 129).

[39] S. Broscheit,
"Investigating Entity Knowledge in BERT with Simple Neural End-To-End Entity Linking",
Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL),
2019 677 (cit. on pp. 3, 44, 73, 76, 79).

[40] M. Dubey et al., "Asknow: A framework for natural language query formalization in sparql",
International Semantic Web Conference, Springer, 2016 300 (cit. on pp. 3, 5, 40).

[41] R. Beaumont, B. Grau and A. L. Ligozat,
SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF,
CEUR Workshop Proceedings (2015) (cit. on pp. 3, 6, 52, 54).

[42] K. Fundel, R. Küffner and R. Zimmer, *RelEx—Relation extraction using dependency parse trees*,
Bioinformatics (2007) (cit. on pp. 3, 39, 127).

[43] C. Casalnuovo, K. Sagae and P. T. Devanbu,
Studying the difference between natural and programming language corpora,
Empirical Software Engineering (2019) 1 (cit. on p. 5).

[44] K. Singh et al., *No one is perfect: Analysing the performance of question answering components over the DBpedia knowledge graph*, J. Web Semant. **65** (2020) 100594 (cit. on p. 5).

[45] K. Singh et al., *No one is perfect: Analysing the performance of question answering components over the dbpedia knowledge graph*, arXiv preprint arXiv:1809.10044 (2018) (cit. on pp. 5, 41).

[46] G. A. Miller, *WordNet: An electronic lexical database*, MIT press, 1998
(cit. on pp. 5, 6, 11, 45, 128).

[47] H. Liu and P. Singh, *ConceptNet—a practical commonsense reasoning tool-kit*,
BT technology journal **22** (2004) 211 (cit. on pp. 5, 11, 13, 45, 119).

[48] R. Navigli and S. P. Ponzetto,
"BabelNet: Building a Very Large Multilingual Semantic Network",
ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, The Association for Computer Linguistics, 2010
216 (cit. on pp. 5, 11, 45).

[49] N. Nakashole, G. Weikum and F. Suchanek,
"PATTY: A taxonomy of relational patterns with semantic types",
Proceedings of the EMNLP 2012, Association for Computational Linguistics, 2012 1135
(cit. on pp. 5, 40).

[50] T. Mikolov et al., *Distributed representations of words and phrases and their compositionality*,
Advances in neural information processing systems **26** (2013) 3111 (cit. on pp. 6, 35, 37, 127).

[51] J. Pennington, R. Socher and C. D. Manning, "Glove: Global vectors for word representation.",
EMNLP, 2014 (cit. on pp. 6, 35–37, 70).

[52] Z. Yang et al., "Xlnet: Generalized autoregressive pretraining for language understanding",
Advances in neural information processing systems, 2019 5754
(cit. on pp. 6, 9, 10, 12, 37, 45, 47, 84, 85, 87, 91, 94, 95, 116, 127, 130).

[53] Y. L. et al., *RoBERTa: A Robustly Optimized BERT Pretraining Approach*, CoRR (2019) (cit. on pp. 6, 10, 12, 45, 84, 85, 87, 95, 130).

[54] L. Wu et al., *Scalable Zero-shot Entity Linking with Dense Entity Retrieval*, arXiv preprint arXiv:1911.03814 (2019) (cit. on p. 6).

[55] G. Luo et al., "Joint Entity Recognition and Disambiguation", *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2015 879 (cit. on pp. 6, 130).

[56] I. Yamada et al., *Global Entity Disambiguation with Pretrained Contextualized Embeddings of Words and Entities*, arXiv: Computation and Language (2019) (cit. on pp. 6, 9).

[57] S. Chen et al., "Improving entity linking by modeling latent entity type information", *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020 (cit. on pp. 6, 46, 88, 97, 98, 127).

[58] K. Bollacker et al., "Freebase: a collaboratively created graph database for structuring human knowledge", *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, ACM, 2008 1247 (cit. on pp. 9, 40, 44).

[59] S. Riedel, L. Yao and A. McCallum, "Modeling Relations and Their Mentions without Labeled Text", *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Proceedings, Part III*, ed. by J. L. Balcázar et al., vol. 6323, Lecture Notes in Computer Science, Springer, 2010 148 (cit. on pp. 9, 40).

[60] Y. Zhang et al., "Position-aware Attention and Supervised Data Improve Slot Filling", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, 2017 35 (cit. on p. 9).

[61] D. Sorokin and I. Gurevych, "Context-aware representations for knowledge base relation extraction", *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017 1784 (cit. on pp. 9, 131).

[62] S. Park et al., "ISOFT at QALD-5: Hybrid Question Answering System over Linked Data and Text Data.", *CLEF (Working Notes)*, 2015 (cit. on p. 9).

[63] C. Unger, A.-C. N. Ngomo and E. Cabrio, "6th Open Challenge on Question Answering over Linked Data (QALD-6)", *Semantic Web Evaluation Challenge*, Springer, 2016 171 (cit. on p. 9).

[64] R. Usbeck et al., "7th open challenge on question answering over linked data (QALD-7)", *Semantic Web Evaluation Challenge*, Springer, 2017 59 (cit. on pp. 9, 39).

[65] P. Trivedi et al., "LC-QuAD: A Corpus for Complex Question Answering over Knowledge Graphs", *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, Springer, 2017 210 (cit. on p. 9).

[66] A. Bordes et al., *Large-scale simple question answering with memory networks*, arXiv preprint arXiv:1506.02075 (2015) (cit. on pp. 9, 54).

[67] H. Zhu et al., "Graph Neural Networks with Generated Parameters for Relation Extraction", *Proceedings of ACL*, 2019 (cit. on p. 9).

[68] Z. Wang et al., "Knowledge graph embedding by translating on hyperplanes", *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI Press, 2014 1112 (cit. on pp. 9, 52, 54).

[69] Z. Zhang, J. Yang and H. Zhao, *Retrospective reader for machine reading comprehension*, arXiv preprint arXiv:2001.09694 (2020) (cit. on p. 10).

[70] D. Graff, *The AQUAINT corpus of English news text:[content copyright] Portions© 1998-2000 New York Times, Inc.,© 1998-2000 Associated Press, Inc.,© 1996-2000 Xinhua News Service*, Linguistic Data Consortium, 2002 (cit. on p. 10).

[71] A. Louis and A. Nenkova, "A corpus of general and specific sentences from news.", *LREC*, 2012 1818 (cit. on p. 10).

[72] H. ElSahar et al., "T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples", *LREC*, 2018 (cit. on pp. 10, 65, 69, 73, 76).

[73] A. Cetoli et al., "A Neural Approach to Entity Linking on Wikidata", *European Conference on Information Retrieval*, Springer, 2019 78 (cit. on pp. 10, 43, 46, 61, 69, 84–88, 90, 91, 96, 127).

[74] P. Jansen and D. Ustalov, "TextGraphs 2019 Shared Task on Multi-Hop Inference for Explanation Regeneration", *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, Association for Computational Linguistics, 2019 (cit. on pp. 13, 104, 107, 119, 121, 122).

[75] R. Speer, J. Chin and C. Havasi, "ConceptNet 5.5: An Open Multilingual Graph of General Knowledge", 2017 4444 (cit. on pp. 13, 45, 109).

[76] B. D. Mishra, N. Tandon and P. Clark, *Domain-Targeted, High Precision Knowledge Extraction*, Transactions of the Association for Computational Linguistics **5** (2017) 233 (cit. on p. 13).

[77] M Brysbaert, A. Warriner and V Kuperman, *Concreteness ratings for 40 thousand generally known English word lemmas.*, Behavior research methods **46** (2014) 904 (cit. on pp. 13, 107, 117).

[78] P. Jansen et al., *Framing QA as Building and Ranking Intersentence Answer Justifications*, Computational Linguistics **43** (2017) 407 (cit. on p. 13).

[79] M. P. K. Ravi et al., *CHOLAN: A Modular Approach for Neural Entity Linking on Wikipedia and Wikidata*, 2021 (cit. on pp. 15, 61, 62, 104, 129, 130).

[80] T. Berners-Lee, R. Fielding and H. Frystyk, *Hypertext transfer protocol–HTTP/1.0*, 1996 (cit. on p. 17).

[81] T. Berners-Lee, J. Hendler and O. Lassila, *The semantic web*, Scientific american **284** (2001) 34 (cit. on p. 18).

[82] T. Bray et al., *Extensible markup language (XML).*, World Wide Web Journal **2** (1997) 27 (cit. on p. 18).

[83] T. Berners-Lee, *Linked data, 2006*, 2006 (cit. on pp. 18, 19).

[84] C. Bizer et al., "Linked data on the web (LDOW2008)", *Proceedings of the 17th international conference on World Wide Web*, ACM, 2008 1265 (cit. on p. 19).

[85] J. Lehmann et al., *DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia*, Semantic Web **6** (2015) 167 (cit. on pp. 19, 44).

[86] D. Vrandečić and M. Krötzsch, *Wikidata: a free collaborative knowledgebase*, Communications of the ACM **57** (2014) 78 (cit. on p. 19).

[87] S. Auer, J. Lehmann and S. Hellmann, "LinkedGeoData: Adding a Spatial Dimension to the Web of Data", *The Semantic Web - ISWC 2009, 8th International Semantic Web Conference, ISWC 2009. Proceedings*, 2009 (cit. on p. 19).

[88] *RDF Schema 1.1 Recommendation*, URL: https://www.w3.org/TR/2014/REC-rdf-schema-20140225/ (cit. on pp. 19, 20).

[89] S. Álvarez-García et al., *Compressed vertical partitioning for efficient RDF management*, Knowledge and Information Systems **44** (2015) 439 (cit. on p. 20).

[90] J. Pérez, M. Arenas and C. Gutierrez, "Semantics and Complexity of SPARQL", *International semantic web conference*, vol. 4273, Springer, 2006 30 (cit. on p. 21).

[91] E. Osuna, R. Freund and F. Girosit, "Training support vector machines: an application to face detection", *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, IEEE, 1997 130 (cit. on p. 25).

[92] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features", *European conference on machine learning*, Springer, 1998 137 (cit. on p. 25).

[93] I. Guyon et al., *Gene selection for cancer classification using support vector machines*, Machine learning **46** (2002) 389 (cit. on p. 25).

[94] S. Mukherjee, E. Osuna and F. Girosi, "Nonlinear prediction of chaotic time series using support vector machines", *Neural Networks for Signal Processing VII. Proceedings of the 1997 IEEE Signal Processing Society Workshop*, IEEE, 1997 511 (cit. on p. 25).

[95] U Thissen et al., *Using support vector machines for time series prediction*, Chemometrics and intelligent laboratory systems **69** (2003) 35 (cit. on p. 25).

[96] *Support Vector Machine Simplified using R*, https://www.listendata.com/2017/01/support-vector-machine-in-r-tutorial.html, Accessed: 2021-02-01 (cit. on p. 26).

[97] *A Brief Overview of Support Vector Machines (SVM)*, https://www.iunera.com/kraken/fabric/support-vector-machines-svm/, Accessed: 2021-02-01 (cit. on p. 26).

[98] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines", *ICML*, 2010 (cit. on p. 28).

[99] D. Hendrycks and K. Gimpel, *Gaussian error linear units (gelus)*, arXiv preprint arXiv:1606.08415 (2016) (cit. on p. 28).

[100] I. Goodfellow et al., "Maxout networks", *International conference on machine learning*, PMLR, 2013 1319 (cit. on p. 28).

[101] Shruti Jadon, *Activation functions*, https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092, [Date Published: March 2018, Date Accessed: January 2019] (cit. on p. 28).

[102] D. E. Rumelhart, G. E. Hinton and R. J. Williams, *Learning representations by back-propagating errors*, Nature **323** (1986) 533 (cit. on pp. 28, 29, 34).

[103] R. Rojas, *Neural Networks: A Systematic Introduction, 1996*, Chap **1** () 3 (cit. on p. 28).

[104] A. A. Razborov, "On Small Depth Threshold Circuits", *Algorithm Theory - SWAT '92, Third Scandinavian Workshop on Algorithm Theory, Helsinki, Finland, July 8-10, 1992, Proceedings*, 1992 42 (cit. on p. 28).

[105] X. Li et al., "Deep neural networks for syllable based acoustic modeling in Chinese speech recognition", *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2013, Kaohsiung, Taiwan, October 29 - November 1, 2013*, 2013 1 (cit. on p. 28).

[106] G. E. Dahl et al., *Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition*, IEEE Trans. Audio, Speech & Language Processing **20** (2012) 30 (cit. on p. 28).

[107] Q. V. Le et al., "Building high-level features using large scale unsupervised learning", *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012 (cit. on p. 28).

[108] Y. LeCun et al., *Gradient-based learning applied to document recognition*, Proceedings of the IEEE **86** (1998) 2278 (cit. on p. 28).

[109] R. Collobert et al., *Natural Language Processing (Almost) from Scratch*, J. Mach. Learn. Res. **12** (2011) 2493 (cit. on p. 28).

[110] P. Covington, J. Adams and E. Sargin, "Deep neural networks for youtube recommendations", *Proceedings of the 10th ACM Conference on Recommender Systems*, ACM, 2016 191 (cit. on p. 28).

[111] C. Edwards, *Deep Learning Hunts for Signals Among the Noise*, Commun. ACM **61** (2018) 13, ISSN: 0001-0782 (cit. on p. 28).

[112] D. Quang and X. Xie, *DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences*, Nucleic acids research **44** (2016) e107 (cit. on p. 29).

[113] A. Graves et al., "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks", *Proceedings of the 23rd international conference on Machine learning*, ACM, 2006 369 (cit. on p. 29).

[114] A. Graves, *Generating sequences with recurrent neural networks*, arXiv preprint arXiv:1308.0850 (2013) (cit. on pp. 29, 32).

[115] A. Graves, A.-r. Mohamed and G. Hinton,
"Speech recognition with deep recurrent neural networks",
Acoustics, speech and signal processing (icassp), 2013 ieee international conference on,
IEEE, 2013 6645 (cit. on p. 29).

[116] S. Liu et al., "A Recursive Recurrent Neural Network for Statistical Machine Translation",
Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers, 2014 1491
(cit. on pp. 29, 30).

[117] I. Sutskever, O. Vinyals and Q. V. Le, "Sequence to Sequence Learning with Neural Networks",
Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, 2014
3104 (cit. on pp. 29–32).

[118] P. J. Werbos, *Backpropagation through time: what it does and how to do it*,
Proceedings of the IEEE **78** (1990) 1550 (cit. on pp. 29, 31, 32).

[119] O. IQ, *When to use Recurrent Neural Networks (RNN)*,
URL: https://iq.opengenus.org/when-to-use-recurrent-neural-networks-rnn/ (visited on 02/01/2021) (cit. on p. 29).

[120] T. Luong, H. Pham and C. D. Manning,
"Effective Approaches to Attention-based Neural Machine Translation",
Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015, 2015 1412 (cit. on pp. 30, 31, 67, 68).

[121] S. Hochreiter, *Investigations on dynamic neural networks*,
Diploma, Technical University m **91** (1991) (cit. on pp. 30, 33).

[122] Y. Bengio, P. Simard and P. Frasconi,
Learning long-term dependencies with gradient descent is difficult,
IEEE transactions on neural networks **5** (1994) 157 (cit. on pp. 30, 33).

[123] S. Hochreiter,
The vanishing gradient problem during learning recurrent neural nets and problem solutions,
International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems **6** (1998) 107
(cit. on pp. 30, 31).

[124] S. Hochreiter et al.,
Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001
(cit. on pp. 30, 31).

[125] B. Jafari, *Difference between feedback RNN and LSTM/GRU*,
URL: https://stats.stackexchange.com/questions/222584/difference-between-feedback-rnn-and-lstm-gru (visited on 02/01/2021) (cit. on p. 30).

[126] F. A. Gers, J. Schmidhuber and F. A. Cummins,
Learning to Forget: Continual Prediction with LSTM, Neural Computation **12** (2000) 2451
(cit. on p. 30).

[127] F. A. Gers, N. N. Schraudolph and J. Schmidhuber,
Learning Precise Timing with LSTM Recurrent Networks,
Journal of Machine Learning Research **3** (2002) 115,
URL: http://www.jmlr.org/papers/v3/gers02a.html (cit. on p. 31).

[128] K. Cho et al., "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation", *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, 2014 1724 (cit. on pp. 31, 32).

[129] I. J. Goodfellow, Y. Bengio and A. C. Courville, *Deep Learning*, Adaptive computation and machine learning, MIT Press, 2016, ISBN: 978-0-262-03561-3, URL: http://www.deeplearningbook.org/ (cit. on pp. 32–34).

[130] D. Zhu et al., *Negative Log Likelihood Ratio Loss for Deep Neural Network Classification*, CoRR **abs/1804.10690** (2018), arXiv: 1804.10690 (cit. on p. 33).

[131] I. Sutskever et al., "On the importance of initialization and momentum in deep learning", *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, 2013 1139, URL: http://jmlr.org/proceedings/papers/v28/sutskever13.html (cit. on p. 33).

[132] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*, 2010 249, URL: http://www.jmlr.org/proceedings/papers/v9/glorot10a.html (cit. on p. 33).

[133] N. Srivastava et al., *Dropout: a simple way to prevent neural networks from overfitting*, Journal of Machine Learning Research **15** (2014) 1929 (cit. on p. 34).

[134] T. S. Ferguson, *An Inconsistent Maximum Likelihood Estimate*, Journal of the American Statistical Association **77** (1982) 831, eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1982.10477894, URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1982.10477894 (cit. on p. 35).

[135] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, ed. by Y. Bengio and Y. LeCun, 2015 (cit. on pp. 35, 77, 118, 119).

[136] R. Lebret and R. Lebret, *Word Emdeddings through Hellinger PCA*, CoRR **abs/1312.5542** (2013), arXiv: 1312.5542 (cit. on p. 35).

[137] M. A. Qureshi and D. Greene, *EVE: Explainable Vector Based Embedding Technique Using Wikipedia*, CoRR **abs/1702.06891** (2017), arXiv: 1702.06891 (cit. on p. 35).

[138] O. Levy and Y. Goldberg, "Neural Word Embedding as Implicit Matrix Factorization", *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 2014 2177 (cit. on p. 35).

[139] R. Socher et al., "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank", *EMNLP*, 2013 1631 (cit. on p. 35).

[140] R. Socher et al., "Parsing with Compositional Vector Grammars", *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, 2013 455 (cit. on p. 35).

[141] T. Mikolov et al., *Efficient Estimation of Word Representations in Vector Space*, CoRR **abs/1301.3781** (2013), arXiv: 1301.3781 (cit. on p. 35).

[142] Google Researchers, *Machine Learning Crash Course*, https://developers.google.com/machine-learning/crash-course/embeddings/translating-to-a-lower-dimensional-space , [Online course, Date Accessed: January, 2019] (cit. on p. 36).

[143] M. E. Peters et al., "Deep contextualized word representations", *Proc. of NAACL*, 2018 (cit. on p. 37).

[144] A. Radford et al., *Improving language understanding by generative pre-training*, URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf (2018) (cit. on p. 37).

[145] A. Radford et al., *Language models are unsupervised multitask learners*, OpenAI blog **1** (2019) 9 (cit. on p. 37).

[146] Z. Dai et al., *Transformer-xl: Attentive language models beyond a fixed-length context*, arXiv preprint arXiv:1901.02860 (2019) (cit. on p. 37).

[147] J. Hoffart et al., "Robust Disambiguation of Named Entities in Text", *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, 2011 782 (cit. on pp. 39, 43, 69, 73, 76, 77, 79, 86, 97, 98, 127).

[148] L. Ratinov et al., "Local and Global Algorithms for Disambiguation to Wikipedia", *ACL*, 2011 1375 (cit. on pp. 39, 97, 98).

[149] D. Zelenko, C. Aone and A. Richardella, *Kernel methods for relation extraction*, Journal of machine learning research (2003) (cit. on pp. 39, 127).

[150] R. C. Bunescu and R. J. Mooney, "A Shortest Path Dependency Kernel for Relation Extraction", *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, Association for Computational Linguistics, 2005 724 (cit. on p. 39).

[151] M. Banko et al., "Open Information Extraction from the Web", *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, Morgan Kaufmann Publishers Inc., 2007 2670 (cit. on p. 39).

[152] J. Zhu et al., "StatSnowball: A Statistical Approach to Extracting Entity Relationships", *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, ACM, 2009 101, ISBN: 978-1-60558-487-4 (cit. on p. 39).

[153] M. Banko and O. Etzioni, "The Tradeoffs Between Open and Traditional Relation Extraction", *Proceedings of ACL-08: HLT*, Association for Computational Linguistics, 2008 28 (cit. on p. 39).

[154] M. Mintz et al., "Distant supervision for relation extraction without labeled data", *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, ed. by K. Su, J. Su and J. Wiebe, The Association for Computer Linguistics, 2009 1003 (cit. on p. 39).

[155] R. Hoffmann et al., "Knowledge-Based Weak Supervision for Information Extraction of Overlapping Relations", *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, ed. by D. Lin, Y. Matsumoto and R. Mihalcea, The Association for Computer Linguistics, 2011 541 (cit. on p. 40).

[156] A. Smirnova and P. Cudré-Mauroux, *Relation extraction using distant supervision: A survey*, ACM Computing Surveys (CSUR) **51** (2018) 1 (cit. on p. 40).

[157] M. Remy, *Wikipedia: The free encyclopedia*, Online Information Review **26** (2002) 434 (cit. on p. 40).

[158] G. Ji et al., "Distant supervision for relation extraction with sentence-level attention and entity descriptions", *Thirty-First AAAI Conference on Artificial Intelligence*, 2017 (cit. on pp. 40, 89).

[159] L. Hu et al., "Improving Distantly-Supervised Relation Extraction with Joint Label Embedding", *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*, ed. by K. Inui et al., Association for Computational Linguistics, 2019 3819 (cit. on p. 40).

[160] C. Unger et al., "Template-based Question Answering over RDF Data", *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, ACM, 2012 639, ISBN: 978-1-4503-1229-5 (cit. on p. 40).

[161] J.-D. Kim and K. B. Cohen, "Natural language query processing for SPARQL generation: A prototype system for SNOMED CT", *Proceedings of biolink*, 2013 32 (cit. on p. 40).

[162] D. Gerber and A.-C. N. Ngomo, "Bootstrapping the linked data web", *1st Workshop on Web Scale Knowledge Extraction @ISWC*, 2011 (cit. on p. 40).

[163] K. Xu et al., *Question answering on freebase via relation extraction and textual evidence*, arXiv preprint arXiv:1603.00957 (2016) (cit. on p. 40).

[164] K. Singh et al., "Towards a message-driven vocabulary for promoting the interoperability of question answering systems", *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, IEEE, 2016 386 (cit. on p. 40).

[165] K. Singh et al., "Qanary - The Fast Track to Creating a Question Answering System with Linked Data Technology", *The Semantic Web - ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, 2016 183 (cit. on p. 40).

[166] A. Both et al., "Qanary–a methodology for vocabulary-driven open question answering systems", *ESWC*, 2016 (cit. on pp. 40, 50).

[167] D. Diefenbach et al., "The Qanary Ecosystem: Getting New Insights by Composing Question Answering Pipelines", *Web Engineering - 17th International Conference, ICWE 2017, Rome, Italy, June 5-8, 2017, Proceedings*, Springer, 2017 171 (cit. on p. 40).

[168] E. Marx et al., “Towards an open question answering architecture”, *Proceedings of the 10th International Conference on Semantic Systems*, ACM, 2014 57 (cit. on pp. 40, 50).

[169] K. Balog, “Entity Linking”, *Entity-Oriented Search*, Springer International, 2018 (cit. on pp. 40, 42, 64).

[170] P. Ferragina and U. Scaiella, “TAGME: on-the-fly annotation of short text fragments (by wikipedia entities)”, *Proceedings of the 19th ACM Conference on Information and Knowledge Management, CIKM 2010, Toronto, Ontario, Canada, October 26-30, 2010*, 2010 1625 (cit. on pp. 41, 86).

[171] R. Mihalcea and A. Csomai, “Wikify!: Linking Documents to Encyclopedic Knowledge”, *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM ’07, ACM, 2007 233, ISBN: 978-1-59593-803-9 (cit. on p. 41).

[172] S. Kulkarni et al., “Collective Annotation of Wikipedia Entities in Web Text”, *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’09, 2009 457, ISBN: 978-1-60558-495-9 (cit. on p. 41).

[173] J. Hoffart, Y. Altun and G. Weikum, “Discovering Emerging Entities with Ambiguous Names”, *Proceedings of the 23rd International Conference on World Wide Web*, WWW ’14, 2014 385, ISBN: 978-1-4503-2744-2 (cit. on p. 41).

[174] D. B. Nguyen, M. Theobald and G. Weikum, *J-NERD: Joint Named Entity Recognition and Disambiguation with Rich Linguistic Features*, TACL **4** (2016) 215 (cit. on pp. 41, 43).

[175] A. Moro, A. Raganato and R. Navigli, *Entity Linking meets Word Sense Disambiguation: a Unified Approach*, Transactions of the Association for Computational Linguistics **2** (2014) 231 (cit. on pp. 41, 42, 77, 79).

[176] R. Usbeck et al., “AGDISTIS-graph-based disambiguation of named entities using linked data”, *International Semantic Web Conference*, Springer, 2014 457 (cit. on pp. 41, 42).

[177] G. Rizzo and R. Troncy, “NERD: A Framework for Unifying Named Entity Recognition and Disambiguation Extraction Tools”, *13th Conference of the European Chapter of the Association for Computational Linguistics*, 2012 (cit. on p. 41).

[178] J.-D. Kim et al., *OKBQA Framework for collaboration on developing natural language question answering systems*, (2017) (cit. on p. 42).

[179] A. Both et al., “Qanary - A Methodology for Vocabulary-Driven Open Question Answering Systems”, *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*, Springer, 2016 625 (cit. on p. 42).

[180] M. Dojchinovski and T. Kliegr, “Entityclassifier.eu: Real-time Classification of Entities in Text with Wikipedia”, *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, ECMLPKDD’13, Springer-Verlag, 2013 654 (cit. on p. 42).

[181] J. R. Finkel, T. Grenager and C. D. Manning,
"Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling",
ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA,
The Association for Computer Linguistics, 2005 363 (cit. on p. 42).

[182] P. Ferragina and U. Scaiella, *Fast and Accurate Annotation of Short Texts with Wikipedia Pages*,
IEEE Software **29** (2012) 70 (cit. on p. 42).

[183] V. Yadav and S. Bethard,
"A Survey on Recent Advances in Named Entity Recognition from Deep Learning models",
Proceedings of the 27th International Conference on Computational Linguistics, 2018 2145
(cit. on pp. 42, 43).

[184] D. Seyler et al.,
"A Study of the Importance of External Knowledge in the Named Entity Recognition Task",
Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers, 2018 241 (cit. on p. 43).

[185] A. Delpeuch, *Opentapioca: Lightweight entity linking for wikidata*,
arXiv preprint arXiv:1904.09131 (2019) (cit. on pp. 43, 44, 46, 69–71, 76, 86).

[186] R. Grishman and B. M. Sundheim, "Message understanding conference-6: A brief history",
COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics,
1996 (cit. on p. 43).

[187] T. Rocktäschel et al., "WBI-NER: The impact of domain-specific features on the performance of identifying and classifying mentions of drugs",
Second Joint Conference on Lexical and Computational Semantics (SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*,
2013 356 (cit. on p. 43).

[188] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning",
Proceedings of the 25th international conference on Machine learning, 2008 160 (cit. on p. 43).

[189] A. Akbik, D. Blythe and R. Vollgraf, "Contextual String Embeddings for Sequence Labeling",
Proceedings of the 27th International Conference on Computational Linguistics,
Association for Computational Linguistics, 2018 1638 (cit. on p. 43).

[190] D. Ceccarelli et al., "Dexter: an open source framework for entity linking", *Proceedings of the sixth international workshop on Exploiting semantic annotations in information retrieval*, 2013
17 (cit. on p. 43).

[191] M. Cornolti et al.,
"A piggyback system for joint entity mention detection and linking in web queries",
Proceedings of the 25th International Conference on World Wide Web, 2016 567 (cit. on p. 43).

[192] S. Zwicklbauer, C. Seifert and M. Granitzer,
"Robust and collective entity disambiguation through semantic embeddings",
Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, 2016 425 (cit. on p. 43).

[193] O. Sevgili et al., *Neural Entity Linking: A Survey of Models based on Deep Learning*, 2020,
arXiv: 2006.00575 [cs.CL] (cit. on pp. 43, 45, 73).

[194] Z. Fang et al., “Joint entity linking with deep reinforcement learning”, *The World Wide Web Conference*, 2019 438 (cit. on pp. 43, 46, 88, 97, 98).

[195] Y. Cao et al., “Bridge text and knowledge by learning multi-prototype entity mention embedding”, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017 1623 (cit. on p. 43).

[196] O.-E. Ganea and T. Hofmann, “Deep Joint Entity Disambiguation with Local Neural Attention”, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017 2619 (cit. on pp. 43, 46, 72, 73, 75, 76, 78, 88, 95, 97, 98).

[197] V. I. Spitkovsky and A. X. Chang, *A cross-lingual dictionary for english wikipedia concepts*, (2012) (cit. on p. 43).

[198] Y. Cao et al., “Neural Collective Entity Linking”, *Proceedings of the 27th International Conference on Computational Linguistics*, 2018 675 (cit. on p. 43).

[199] L. Logeswaran et al., “Zero-Shot Entity Linking by Reading Entity Descriptions”, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019 3449 (cit. on p. 43).

[200] T. Févry et al., “Empirical Evaluation of Pretraining Strategies for Supervised Entity Linking”, *Automated Knowledge Base Construction*, 2020 (cit. on pp. 44, 73, 77, 79, 127).

[201] I. O. Mulang et al., “Encoding Knowledge Graph Entity Aliases in Attentive Neural Network for Wikidata Entity Linking”, *International Conference on Web Information Systems Engineering*, Springer, 2020 328 (cit. on pp. 44, 46, 61, 62, 76–78, 84, 90, 100, 129, 130).

[202] F. M. Suchanek, G. Kasneci and G. Weikum, “Yago: a core of semantic knowledge”, *Proc. of the 16th Int. Conf. on World Wide Web*, 2007 697 (cit. on pp. 44, 64).

[203] P. Rosso, D. Yang and P. Cudré-Mauroux, “Beyond triplets: hyper-relational knowledge graph embedding for link prediction”, *Proceedings of The Web Conference 2020*, 2020 1885 (cit. on pp. 44, 89).

[204] N. Chah, *Freebase-triples: A methodology for processing the freebase data dumps*, arXiv preprint arXiv:1712.08707 (2017) (cit. on p. 44).

[205] G. A. Miller, *WordNet: a lexical database for English*, Communications of the ACM **38** (1995) 39 (cit. on p. 45).

[206] A. Sakor et al., “Falcon 2.0: An Entity and Relation Linking Tool over Wikidata”, *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020 3141 (cit. on p. 46).

[207] M. Klang and P. Nugues, “Hedwig: A Named Entity Linker”, *Proceedings of The 12th Language Resources and Evaluation Conference*, 2020 4501 (cit. on p. 46).

[208] D. Sorokin and I. Gurevych, *Mixing context granularities for improved entity linking on question answering data across entity categories*, arXiv preprint arXiv:1804.08460 (2018) (cit. on p. 46).

[209] X. Lin et al., *KBPearl: a knowledge base population system supported by joint entity and relation linking*, Proceedings of the VLDB Endowment **13** (2020) 1035 (cit. on p. 46).

[210] D. Banerjee et al.,
"PNEL: Pointer Network based End-To-End Entity Linking over Knowledge Graphs",
International Semantic Web Conference, Springer, 2020 21 (cit. on p. 46).

[211] B. Huang et al., *Entity Linking for Short Text Using Structured Knowledge Graph via Multi-grained Text Matching*, Proc. Interspeech 2020 (2020) 4178 (cit. on p. 46).

[212] E. Boros et al.,
"Robust named entity recognition and linking on historical multilingual documents",
Conference and Labs of the Evaluation Forum (CLEF 2020), vol. 2696, Paper 171,
CEUR-WS Working Notes, 2020 1 (cit. on p. 46).

[213] V. Provatorova et al., "Named Entity Recognition and Linking on Historical Newspapers: UvA. ILPS & REL at CLEF HIPE 2020", *CLEF*, 2020 (cit. on p. 46).

[214] K. Labusch and C. Neudecker,
"Named Entity Disambiguation and Linking Historic Newspaper OCR with BERT", *CLEF*, 2020 (cit. on p. 46).

[215] J. A. Botha, Z. Shan and D. Gillick, *Entity Linking in 100 Languages*,
arXiv preprint arXiv:2011.02690 (2020) (cit. on p. 46).

[216] B. Harandizadeh and S. Singh,
"Tweeki: Linking Named Entities on Twitter to a Knowledge Graph",
Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020), 2020 222 (cit. on p. 46).

[217] Y. Y. et al., "Collective Entity Disambiguation with Structured Gradient Tree Boosting", *NAACL*, 2018 (cit. on pp. 46, 88, 97).

[218] P. Le and I. Titov, "Improving Entity Linking by Modeling Latent Relations between Mentions", *ACL*, 2018 (cit. on pp. 46, 88, 97, 98).

[219] H. Hajishirzi et al.,
"Joint coreference resolution and named-entity linking with multi-pass sieves",
Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 2013 289 (cit. on p. 45).

[220] N. Gupta, S. Singh and D. Roth,
"Entity linking via joint encoding of types, descriptions, and context",
Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, 2017 2681 (cit. on p. 45).

[221] G. Durrett and D. Klein, *A joint model for entity analysis: Coreference, typing, and linking*, Transactions of the association for computational linguistics **2** (2014) 477 (cit. on pp. 45, 73).

[222] H. Wang, H. Ren and J. Leskovec,
Entity Context and Relational Paths for Knowledge Graph Completion,
arXiv preprint arXiv:2002.06757 (2020) (cit. on p. 47).

[223] X. Wang et al., "Kgat: Knowledge graph attention network for recommendation", *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019 950 (cit. on p. 47).

[224] K. Wiharja et al., *Schema aware iterative Knowledge Graph completion*,
Journal of Web Semantics **65** (2020) 100616, ISSN: 1570-8268 (cit. on p. 47).

[225] Ó. Ferrández et al.,
The QALL-ME framework: A specifiable-domain multilingual question answering architecture,
Web semantics: Science, services and agents on the world wide web **9** (2011) 137 (cit. on p. 50).

[226] S. Hakimov et al.,
Semantic question answering system over linked data using relational patterns,
Proceedings of the Joint EDBT/ICDT 2013 Workshops on - EDBT '13 (2013) 83
(cit. on pp. 50, 54).

[227] C. Unger et al., *Question answering over linked data (QALD-5)*,
CEUR Workshop Proceedings **1391** (2015) (cit. on pp. 50, 59).

[228] D. Lukovnikov et al., "Neural Network-based Question Answering over Knowledge Graphs on Word and Character Level",
Proceedings of the 26th International Conference on World Wide Web - WWW '17,
ACM Press, 2017 1211 (cit. on p. 54).

[229] P. Liang, *Learning Compositional Semantics*, (2013) 1 (cit. on p. 54).

[230] K. Nebhi, *A rule-based relation extraction system using DBpedia and syntactic parsing*,
CEUR Workshop Proceedings **1064** (2013) 1, ISSN: 16130073 (cit. on p. 55).

[231] S. Reddy et al., *Transforming Dependency Structures to Logical Forms for Semantic Parsing*,
(2016) (cit. on p. 56).

[232] C. Felser, T. Marinis and H. Clahsen,
Children's Processing of Ambiguous Sentences: A Study of Relative Clause Attachment,
Language Acquisition **11** (2003) 127, ISSN: 1048-9223 (cit. on p. 56).

[233] A.-C. Ngonga Ngomo et al.,
Sorry, i don't speak SPARQL: translating SPARQL queries into natural language, (2013) 977
(cit. on p. 56).

[234] A. Budanitsky and G. Hirst,
Evaluating WordNet-based Measures of Lexical Semantic Relatedness,
Computational Linguistics **32** (2006) 13 (cit. on p. 57).

[235] Z. Wu and M. Palmer, "Verbs semantics and lexical selection",
Proceedings of the 32nd annual meeting on Association for Computational Linguistics,
Association for Computational Linguistics, 1994 (cit. on p. 57).

[236] C. Leacock and M. Chodorow,
Combining local context and WordNet similarity for word sense identification,
WordNet: An electronic lexical database **49** (1998) 265 (cit. on p. 57).

[237] C. Bizer et al., *DBpedia - A crystallization point for the Web of Data*,
Web Semantics: Science, Services and Agents on the World Wide Web **7** (2009) 154,
ISSN: 15708268 (cit. on p. 59).

[238] C. Gormley and Z. Tong,
Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine,
" O'Reilly Media, Inc.", 2015 (cit. on p. 70).

[239] R. Shu and A. Miura, "Residual Stacking of RNNs for Neural Machine Translation",
Proceedings of the 3rd Workshop on Asian Translation, WAT@COLING 2016, Osaka, Japan, December 2016, 2016 223 (cit. on p. 72).

[240] S. Edunov et al., "Understanding Back-Translation at Scale",
Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, 2018 489 (cit. on p. 72).

[241] D. Blythe, A. Akbik and R. Vollgraf,
Syntax-Aware Language Modeling with Recurrent Neural Networks, 2018,
arXiv: 1803.03665 (cit. on p. 72).

[242] F. Piccinno and P. Ferragina, "From TagME to WAT: a new entity annotator",
Proceedings of the first international workshop on Entity recognition & disambiguation, 2014 55 (cit. on pp. 73, 76, 79).

[243] M. E. Peters et al., "Knowledge Enhanced Contextual Word Representations",
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 43 (cit. on pp. 73, 76, 79).

[244] Z. Guo and D. Barbosa, *Robust named entity disambiguation with random walks*,
Semantic Web **9** (2018) 459 (cit. on pp. 73, 76, 97, 98).

[245] E. F. T. K. Sang and F. D. Meulder,
Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition,
CoRR **cs.CL/0306050** (2003) (cit. on p. 74).

[246] I. Yamada and H. Shindo, *Pre-training of deep contextualized embeddings of words and entities for named entity disambiguation*, arXiv preprint arXiv:1909.00426 (2019) (cit. on p. 76).

[247] A. Moro, A. Raganato and R. Navigli,
Entity linking meets word sense disambiguation: a unified approach,
Transactions of the Association for Computational Linguistics **2** (2014) 231 (cit. on pp. 76, 79).

[248] L. Yao, C. Mao and Y. Luo, *KG-BERT: BERT for Knowledge Graph Completion*,
arXiv preprint arXiv:1909.03193 (2019) (cit. on p. 77).

[249] T. Wolf et al., *HuggingFace's Transformers: State-of-the-art Natural Language Processing*,
ArXiv **abs/1910.03771v5** (2019) (cit. on p. 77).

[250] W. L. et al., *K-BERT: Enabling Language Representation with Knowledge Graph*, ArXiv (2019) (cit. on pp. 83, 84).

[251] A. B. K. S. J. H. S. A. Isaiah Onando Mulang' Abhishek Nadgeri,
Its just the silly context! Analyzing the Role of Wikidata context onEntity Disambiguation Models,
The World Wide Web Journal (2021) (cit. on p. 83).

[252] A. L. et al., *PyTorch-BigGraph: A Large-scale Graph Embedding System*, CoRR (2019) (cit. on p. 84).

[253] S. Heindorf et al., "Vandalism detection in wikidata", *CIKM*, 2016 (cit. on p. 84).

[254] H. Shahbazi et al.,
Entity-aware ELMo: Learning Contextual Entity Representation for Entity Disambiguation,
CoRR (2019) (cit. on p. 88).

[255] J. Frey et al., *Evaluation of metadata representations in RDF stores*,
Semantic Web **10** (2019) 205 (cit. on p. 89).

[256] H. Wang et al., “Knowledge-aware graph neural networks with label smoothness regularization for recommender systems”, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019 968 (cit. on p. 89).

[257] E. Charton et al., “Improving Entity Linking using Surface Form Refinement”, *LREC*, 2014 (cit. on p. 90).

[258] K. Lee et al., *End-to-end Neural Coreference Resolution*, CoRR **abs/1707.07045** (2017), arXiv: 1707.07045 (cit. on p. 90).

[259] B. Hachey, W. Radford and J. R. Curran, “Graph-based named entity linking with wikipedia”, *International conference on web information systems engineering*, Springer, 2011 213 (cit. on p. 90).

[260] E. Zhou and J. D. Choi, “They Exist! Introducing Plural Mentions to Coreference Resolution and Entity Linking”, *Proceedings of the 27th International Conference on Computational Linguistics*, Association for Computational Linguistics, 2018 24 (cit. on p. 90).

[261] C. Tan et al., “Multiway Attention Networks for Modeling Sentence Pairs”, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, International Joint Conferences on Artificial Intelligence Organization, 2018 4411 (cit. on p. 94).

[262] T. Rocktäschel et al., *Reasoning about Entailment with Neural Attention*, CoRR **abs/1509.06664** (2016) (cit. on p. 94).

[263] D. Milne and I. H. Witten, “Learning to link with wikipedia”, *Proceedings of the 17th ACM conference on Information and knowledge management*, ACM, 2008 509 (cit. on pp. 97, 98).

[264] N. Lazic et al., *Plato: A Selective Context Model for Entity Resolution.*, Trans. Assoc. Comput. Linguistics **3** (2015) 503 (cit. on p. 97).

[265] A. Globerson et al., “Collective Entity Resolution with Multi-Focal Attention”, *ACL*, 2016 (cit. on p. 97).

[266] H. Huang, L. P. Heck and H. Ji, *Leveraging Deep Neural Networks and Knowledge Graphs for Entity Disambiguation*, CoRR **abs/1504.07678** (2015) (cit. on p. 97).

[267] O. Ganea et al., *Probabilistic Bag-Of-Hyperlinks Model for Entity Linking*, CoRR **abs/1509.02301** (2015), arXiv: 1509.02301 (cit. on p. 97).

[268] A. Chisholm and B. Hachey, *Entity Disambiguation with Web Links*, TACL **3** (2015) 145 (cit. on p. 97).

[269] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions”, *Advances in Neural Information Processing Systems*, 2017 4765 (cit. on p. 103).

[270] D. Paul and A. Frank, “Ranking and Selecting Multi-Hop Knowledge Paths to Better Predict Human Needs”, *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)*, 2019 3671 (cit. on p. 103).

[271] M. T. Ribeiro, S. Singh and C. Guestrin,
"Why should i trust you?: Explaining the predictions of any classifier", *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, ACM, 2016 1135 (cit. on p. 103).

[272] P. Jansen et al., "WorldTree: A Corpus of Explanation Graphs for Elementary Science Questions supporting Multi-hop Inference", *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC)*, 2018 (cit. on pp. 103, 105–108, 116, 118).

[273] I. O. Mulang, J. D'Souza and S. Auer,
"Fine-tuning BERT with Focus Words for Explanation Regeneration",
Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics, 2020 125 (cit. on pp. 104, 121).

[274] S. A. Jennifer D'Souza Isaiah Onando Mulang',
Ranking Facts for Explaining Answers to ElementaryScience Questions, () (cit. on p. 104).

[275] D. Khashabi et al.,
On the capabilities and limitations of reasoning for natural language understanding,
arXiv preprint arXiv:1901.02522 (2019) (cit. on p. 106).

[276] D. Fried et al., *Higher-order lexical semantic models for non-factoid answer reranking*,
Transactions of the Association for Computational Linguistics **3** (2015) 197 (cit. on p. 106).

[277] J. D'Souza, I. O. Mulang and S. Auer, "Team SVMrank: Leveraging Feature-rich Support Vector Machines for Ranking Explanations to Elementary Science Questions",
Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13), 2019 90 (cit. on pp. 106, 116, 119, 121).

[278] J. Fürnkranz and E. Hüllermeier, "Preference learning and ranking by pairwise comparison", *Preference learning*, Springer, 2010 65 (cit. on pp. 106, 124).

[279] T. Kamishima, H. Kazawa and S. Akaho,
"A survey and empirical comparison of object ranking methods", *Preference learning*,
Springer, 2010 181 (cit. on pp. 106, 115, 124).

[280] A. M. Turing, "Computing machinery and intelligence", *Parsing the turing test*, Springer, 2009 23 (cit. on p. 107).

[281] P. Clark et al.,
Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge,
ArXiv **abs/1803.05457** (2018) (cit. on pp. 107, 118).

[282] G. Angeli, M. J. J. Premkumar and C. D. Manning,
"Leveraging Linguistic Structure For Open Domain Information Extraction", *ACL*, 2015
(cit. on pp. 110, 119).

[283] Y. K. Chia, S. Witteveen and M. Andrews, "Red Dragon AI at TextGraphs 2019 Shared Task: Language Model Assisted Explanation Generation", *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, 2019 85
(cit. on pp. 111, 116, 119–121).

[284] P. Jansen et al., "What's in an Explanation? Characterizing Knowledge and Inference Requirements for Elementary Science Exams", *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*,
The COLING 2016 Organizing Committee, 2016 2956 (cit. on pp. 112, 116).

[285] V. Melnikov et al., *Pairwise versus Pointwise Ranking: A Case Study*, Schedae Informaticae **25** (2016) 73 (cit. on pp. 112, 124).

[286] T. Joachims, "Training linear SVMs in linear time", *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2006 217 (cit. on pp. 114, 119).

[287] T. Kamishima, H. Kazawa and S. Akaho, "Supervised ordering-an empirical survey", *Fifth IEEE International Conference on Data Mining (ICDM'05)*, IEEE, 2005 4 (cit. on pp. 115, 124).

[288] V. N. Vapnik, *The nature of statistical learning theory*, 1995 (cit. on p. 115).

[289] T. Joachims, *Learning to classify text using support vector machines*, vol. 668, Springer Science & Business Media, 2002 (cit. on p. 115).

[290] P. Clark, P. Harrison and N. Balasubramanian, "A study of the knowledge base requirements for passing an elementary science test", *Proceedings of the 2013 workshop on Automated knowledge base construction*, ACM, 2013 37 (cit. on p. 116).

[291] R. Das et al., "Chains-of-Reasoning at TextGraphs 2019 Shared Task: Reasoning over Chains of Facts for Explainable Multi-hop Inference", *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, 2019 101 (cit. on pp. 116, 120–122).

[292] S. Swayamdipta et al., *Frame-Semantic Parsing with Softmax-Margin Segmental RNNs and a Syntactic Scaffold*, arXiv preprint arXiv:1706.09528 (2017) (cit. on p. 119).

[293] P. Banerjee, "ASU at TextGraphs 2019 Shared Task: Explanation ReGeneration using Language Models and Iterative Re-Ranking", *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, 2019 78 (cit. on pp. 119–121, 124).

[294] R. Nogueira and K. Cho, *Passage Re-ranking with BERT*, arXiv preprint arXiv:1901.04085 (2019) (cit. on p. 120).

[295] P. Xu and D. Barbosa, "Connecting Language and Knowledge with Heterogeneous Representations for Neural Relation Extraction" (cit. on p. 127).

[296] A. Hogan et al., *Knowledge graphs*, (2020) (cit. on p. 127).

[297] J. Z. Pan et al., "Entity Enabled Relation Linking", *International Semantic Web Conference*, Springer, 2019 523 (cit. on pp. 128, 129).

[298] K. Singh et al., "Capturing knowledge in semantically-typed relational patterns to enhance relation linking", *Proceedings of the Knowledge Capture Conference*, 2017 1 (cit. on p. 129).

[299] Y. Lin et al., "Learning entity and relation embeddings for knowledge graph completion", *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 29, 1, 2015 (cit. on p. 131).

APPENDIX A

List of Publications

- *Journal Papers (peer reviewed)*
  1. **I.O. Mulang'**, K. Singh, A. Nadgeri, S. Shekarpour, J. Hoffart, S. Auer. *Its just the silly context! Analyzing the Role of Wikidata context on Entity Disambiguation Models* (Under Review - World Wide Web Journal)
  2. J. D'Souza, **I.O. Mulang'**, S. Auer. *Ranking Facts for Explaining Answers to Elementary Science Questions*. (Under Review - Journal of Natural Language Engineering)
- *Conference Papers (peer reviewed)*
  3. **I.O. Mulang'**, K Singh, F Orlandi. *Matching Natural Language Relations to Knowledge Graph Properties for Question Answering.* In Proceedings of the Semantics, ACM, 2017. DOI : https://doi.org/10.1145/3132218.3132229
  4. **I.O. Mulang'**, K. Singh, A. Vyas, S. Shekarpour, M.E. Vidal, Jens Lehmann, Sören Auer *Encoding Knowledge Graph Entity Aliases in Attentive Neural Network for Wikidata Entity Linking*. In proceedings of the Web Information Systems Engineering – WISE 2020. WISE 2020. Lecture Notes in Computer Science, vol 12342. Springer. DOI: https://doi.org/10.1007/978-3-030-62005-9_41
  5. M. Prabhakar, K. Singh, **I.O. Mulang'**, S. Shekarpour, J. Hoffart, J. Lehmann. *CHOLAN: A Modular Approach for Neural Entity linking over Wikidata and Wikipedia*. In the Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics. Online URL: https://aclanthology.org/2021.eacl-main.40
  6. **I.O. Mulang'**, K. Singh, C. Prabhu, A. Nadgeri, J. Hoffart, J. Lehmann. *Evaluating the Impact of Knowledge Graph Context on Entity Disambiguation Models* In CIKM'20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020, pages 2157–2160, 2020. ACM. DOI: https://doi.org/10.1145/3340531.3412159
  7. **I.O. Mulang'**, J. D'Souza, S. Auer. *Fine-tuning BERT with Focus Words for Explanation Regeneration*. In Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics. StarSem, 2020. Online URL: https://aclanthology.org/2020.starsem-1.13
- *Workshop Articles (peer reviewed)*

8. Jennifer D'Souza, **Isaiah Onando Mulang', Sören Auer**.*Team SVMrank: Leveraging Feature-rich Support Vector Machines for Ranking Explanations to Elementary Science Questions. TextGraph workshop, EMNLP 2019*. DOI: http://dx.doi.org/10.18653/v1/D19-5312

- *Miscellaneous Papers (peer reviewed)*

  Following publications originated during and are related to this thesis but are not part of the thesis itself.

9. K Singh, **I.O Mulang'**, Jaradeh, A Sakor, I Lytra, ME Vidal, C Lange, S Auer. *Capturing Knowledge in Semantically-typed Relational Patterns to Enhance Relation Linking.* In Proceedings of the Knowledge Capture Conference (K-Cap), 2017, ACM. DOI: https://doi.org/10.1145/3148011.3148031
10. A. Sakor, **I.O. Mulang'**, K. Singh, S. Shekarpour, M.E. Vidal, J. Lehmann, S. Auer. *Old is Gold: Linguistic Driven Approach for Entity and Relation Linking of Short Text* - NAACL, 2019. DOI: http://dx.doi.org/10.18653/v1/N19-1243

Abbreviations and Acronyms

AI	Artificial Intelligence
NLU	Natural Language Understanding
NLP	Natural Language Processing
QA	Question Answering
NER	Named Entity Recognition
NED	Named Entity Disambiguation
NERD	Named Entity Recognition and Disambiguation
RE	Relation Extraction
RL	Relation Linking
CL	Class Linking
QB	Query Builder
LOD	Linked Open Data
PDF	Portable Document Format
CSV	Coma Separated Values
XML	Extensible Markup Language
RDF	Resource Description Framework
HTML	HyperText Markup Language
HTTP	The Hypertext Transfer Protocol
KG	Knowledge Graph
KB	Knowledge Base
URI	Unified Resource Identifier
NN	Neural Networks
LSTM	Long Short Term Memory (Networks)
LM	Language Model
GNN	Graph Neural Networks

List of Figures

List of Tables