Matching Natural Language Relations to Knowledge Graph Properties for Question Answering

Isaiah Onando Mulang'
University of Bonn
Bonn, Germany
mulang@iai.uni-bonn.de

Kuldeep Singh
Fraunhofer IAIS
Sankt Augustin, Germany
kuldeep.singh@iais.fraunhofer.de

Fabrizio Orlandi
Fraunhofer IAIS
Sankt Augustin, Germany
orlandi@iai.uni-bonn.de

ABSTRACT

Research has seen considerable achievements concerning translation of natural language patterns into formal queries for Question Answering (QA) based on Knowledge Graphs (KG). One of the main challenges in this research area is about how to identify which property within a Knowledge Graph matches the predicate found in a Natural Language (NL) relation. Current approaches for formal query generation attempt to resolve this problem mainly by first retrieving the named entity from the KG together with a list of its predicates, then filtering out one from all the predicates of the entity. We attempt an approach to directly match an NL predicate to KG properties that can be employed within QA pipelines. In this paper, we specify a systematic approach as well as providing a tool that can be employed to solve this task. Our approach models KB relations with their underlying parts of speech, we then enhance this with extra attributes obtained from Wordnet and Dependency parsing characteristics. From a question, we model a similar representation of query relations. We then define distance measurements between the query relation and the properties representations from the KG to identify which property is referred to by the relation within the query. We report substantive recall values and considerable precision from our evaluation.

KEYWORDS

Knowledge Graph, Question Answering, Relation Extraction

ACM Reference format:
Isaiah Onando Mulang', Kuldeep Singh, and Fabrizio Orlandi. 2017. Matching Natural Language Relations to Knowledge Graph Properties for Question Answering. In *Proceedings of Semantics2017, Amsterdam, Netherlands, September 11–14, 2017,* 8 pages.
DOI: 10.1145/3132218.3132229

1 INTRODUCTION

The constantly growing amount of data and information available on the Web is driving research efforts on new efficient solutions for finding the right information in ever increasing data sources. Question Answering (QA) systems, which automatically translate natural language questions posed by humans into complex queries over knowledge bases, facilitate users' access to increasingly large and complex knowledge bases. Despite their apparent success with popular commercial products such as the Google Assistant[1] and Amazon's Alexa[2], QA systems still present many challenges[3].

Typical QA processes consist of different tasks such as named entity identification, named entity disambiguation, relation extraction and linking, query generation, query processing and answer generation [25]. In this work we focus on the particular step of *relation extraction for natural language questions.* We can define this as *the process required to identify semantic relations between named entities within a question expressed in natural language.* Relation extraction is not a new topic in the Natural Language Processing (NLP) research field [31]. However, novel solutions are currently being investigated when attempting to answer natural language questions using facts contained in a Knowledge Graph (KG) [18].

Knowledge graphs, such as DBpedia [1] or Google's Knowledge Graph[4], are gaining increasing importance especially for QA systems as they are (i) very extensive sources of facts, (ii) already structured, (iii) constantly growing/updated and (iv) publicly available on the Web. However, QA over KGs presents additional challenges: KGs are usually quite large and difficult to query and process; lexical forms for relations expressed in a question can differ from those used in the KG (usually referred to as the *Lexical Gap* [15]).

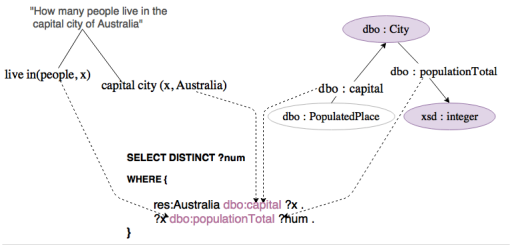

Figure 1: Question relations vs KG properties, an example

In addition to these challenges, in this paper we address an additional relevant problem. Question answering systems implement QA tasks either by dedicating individual components in its architecture to each task or by combing few tasks together in their implementation. In component-based QA systems and frameworks like OKBQA[5], QANARY [5], QALL-ME[9], openQA [19], researchers have implemented individual components dedicated to particular

Semantics2017, Amsterdam, Netherlands

978-1-4503-5296-3/17/09…$15.00
DOI: 10.1145/3132218.3132229

[1] https://assistant.google.com/
[2] https://developer.amazon.com/alexa
[3] See for example relevant research workshops at SIGIR (http://sigir2017.okbqa .org/) and ESWC (https://project-hobbit.eu/challenges/qald2017/).
[4] https://www.google.com/intl/es419/insidesearch/features/search/knowledge.html
[5] http://www.okbqa.org/

tasks. Stanford NER [6], NERD [7], Alchemy API, FOX[8], AGDISTIS[9] are some of the most popular dedicated tools/components for specific tasks like named entity recognition, named entity disambiguation in QA systems. However, to the best of our knowledge, there is no 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. The creation of a standalone and reusable component for relation extraction and linking in this context would not only facilitate reuse of the component in different QA systems but also create a benchmark for the research community for future comparison and evaluation.

In this paper we propose a novel approach, and an implementation, that addresses some of the aforementioned challenges: (1) It is capable of dealing with large KGs such as DBpedia; (2) It addresses the lexical gap problem through the combination of different similarity measures; (3) It is designed as an independent component that can be easily reused in different QA systems.

Current 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 predicates of the entity. 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[10] 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 usage of PATTY [12] which is a large corpus of relational patterns and associated DBpedia predicates due to its noisy behavior. 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 rather than improving overall performance. It is important to note that PATTY is not a relation linking tool, rather a knowledge base for semantically typed relational patterns which may be used in QA systems to implement relation linking task.

For our work, we performed evaluation using the QALD-5 dataset [27] 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 containing one relation).

[6]https://nlp.stanford.edu/software/CRF-NER.shtml
[7]http://nerd.eurecom.fr/
[8]http://aksw.org/Projects/FOX.html
[9]http://aksw.org/Projects/AGDISTIS.html
[10]"About WordNet". WordNet, Princeton University. 2010. http://wordnet.princeton.edu

The rest of the paper is structured as follows. Section 2 provides a concrete example for this work and Section 3 summarizes the related work for the areas of question answering and relation extraction. In Section 4 the overall proposed approach is described and in Section 5 the evaluation setup and the results of the experiments are explained before concluding the paper.

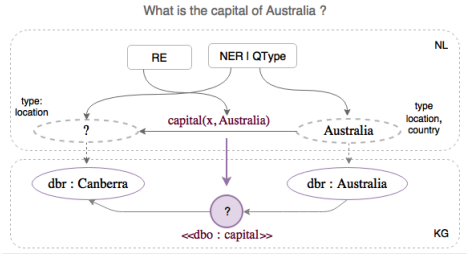

Figure 2: Problem example with the question: "What is the capital of Australia?"

2 MOTIVATING EXAMPLE

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 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 and 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 a 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).

3 RELATED WORK

Relation extraction 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 [31]. In the field of NLP and machine learning, researchers have addressed this problem using different approaches. The work in [31] introduces a kernel-based machine learning method for relation extraction in given natural language text. RelEx [10] uses dependency parse trees and applies a few simples rules to these trees to extract relation from free text.

Relation extraction in natural language text does find its applicability in the field of question answering. PATTY [20] is a popular work which is used in many question answering systems for linking relations to its knowledge base properties. PATTY mines semantically-typed relational patterns from a 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 [7] has a dedicated component

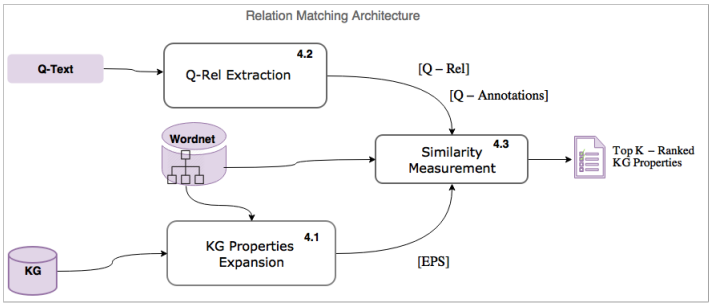

Figure 3: Overall relation matching system architecture: from a question (Q-Text) as input to a ranked list of top K properties in the KG matching the relations in the input question

for the relation extraction task that uses PATTY as underlying large corpus to find semantic relational patterns. TBSL [26] and LODQA [13] implement a two step process to directly translate a natural language question into a SPARQL query. During this translation process, TBSL uses BOA pattern [11] identifiers to detect properties (i.e. relations) that connect the entities present in the input question. Moreover, additional work such as [30] 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, due to the monolithic implementation of their QA pipeline, it is not trivial to reuse this specific module in other QA approaches. For example, reusing it in frameworks such as OKBQA[11], QANARY [5] and openQA [19] 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.

4 APPROACH

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. Figure 3[12] depicts the overall structure of the system.

4.1 KG Properties Expansion

A property in a KG is defined by an directed labeled edge between two nodes of the graph that is identified via a unique URI. Properties can be visualized 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 that are 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. A 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. With the understanding that the second view would demand that the named entities be first disambiguated before the properties can be matched.

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 syntactic analysis. At this stage (to retain the structure of the EPS and reduce the memory load time) we only consider extracting synonyms and hyponyms from a taxonomy like Wordnet and ignore elements related to the derivational forms. 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 [2] 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 [28]. The label attribute of the property provide a natural language mention of the property which is 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*[13]) namespace or the DBpedia properties one (*dbp*[14]). Out of a possible total of 63,764 items classified as properties in the DBpedia ontology, only about 3,500 have

[11] http://www.okbqa.org/
[12] Numbers 4.1 to 4.3 in Figure 3 indicate the respective section in the paper where each component is described

[13] *dbo* stands for: http://dbpedia.org/ontology/
[14] *dbp* stands for: http://dbpedia.org/property/

instances within the KG. We identify 2,795 properties[15] 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 since questions would demand properties that have participated in at least one factual instance within the KG.

Formally, a property $p \in P$, where P is defined in a graph G = {SxPxO} as the set of all properties in G, is expanded into a septuple $(\rho, \beta, \lambda, \omega, c, \mu, A)$ 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.

4.2 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.Given that questions are often succinct and may lack some distant syntactic and semantic associations that would normally be present in free text while also inherently contain implicit or explicit characteristics that may not be exhibited in free text, we make some assumptions and formulate constraints that would assist to represent 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 I.E. 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 Beaumont et al. [2] a set of phrases within the question can be determined that correspond to semantic

[15]This figure can be obtained from: http://wiki.dbpedia.org/services-resources/ontology

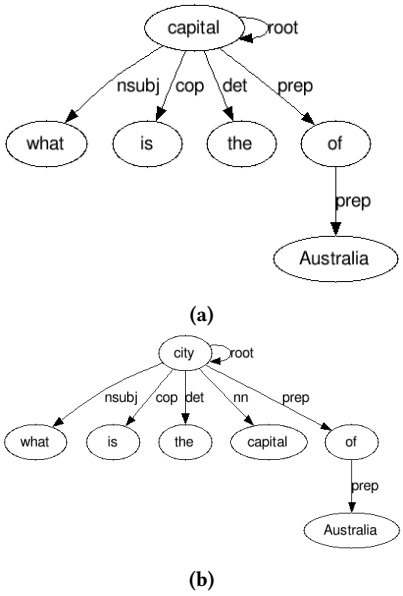

Figure 4: Simple question dependency parse tree

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 Bordes et al. [4], Lukovnikov et al. [18] and as such the relation can only match one unique property in the KB. Each simple question has a desire, the type of answer expected [12] a binary relation, which can be represented in logical form $rel(x, y)$ in which rel describes the relationship between known or unknown entities x and y [17] 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.

In this work, we represent a simple question as a single relation, hereafter referred as Q-Rel. Formally Q-Rel is an octuple (δ,η,α,ℓ,γ,$\mathcal{E}$,$\mathcal{N}$,Υ) 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 [28]
$\gamma \leftarrow$ The right element of the relation or the relation tail [28]
$\mathcal{E} \leftarrow$ 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 4a 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 4b. We overcome this difference at the dependency adjustment stage.

4.2.1 Question Desire. The question desire, sometimes called the question answer type[23] is a classification that denotes what kind of an answer is expected from the question.The task of question answer type identification is well studied with several approaches proposed. For this task, we modify an existing implementation available online[16] that is based on Lee and Roth [16]. The method trains a Support vector machine (SVM) classifier using the TREC[17] dataset with 94% accuracy on Coarse classes and 88% on fine classes. The SVM is a maximum margin classifier in which a function is defined that transforms the training vectors by mapping into a higher dimensional space then finds, in this higher dimensional space, a hyperplane that obtains the widest margin separation. For a clearer explanation of the usage of SVM for question classification see [32]. We only employ the coarse model for our classification since the six classes: location, human, abbreviation, entity, number, and description can be matched to the domain or range of a property. For the question *How many people live in the capital of Australia* we obtain the question type : number, on the other hand for the sub question *What is the capital city of Australia ?* we obtain the desire i.e. location.

4.2.2 Dependency Adjustment. Rules have been used in several relation extraction tasks for either directly identifying relations Nebhi [21] 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 simple question that would translate into $Q - Rels$. Based on the initial parse characteristics of a question, we identify the following four elements of complex questions as opportunities for decomposition into the constituent simple questions. Three of these are largely inspired by the work of Reddy et. al Reddy et al. [24] 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 [8, 22] 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 so as 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*

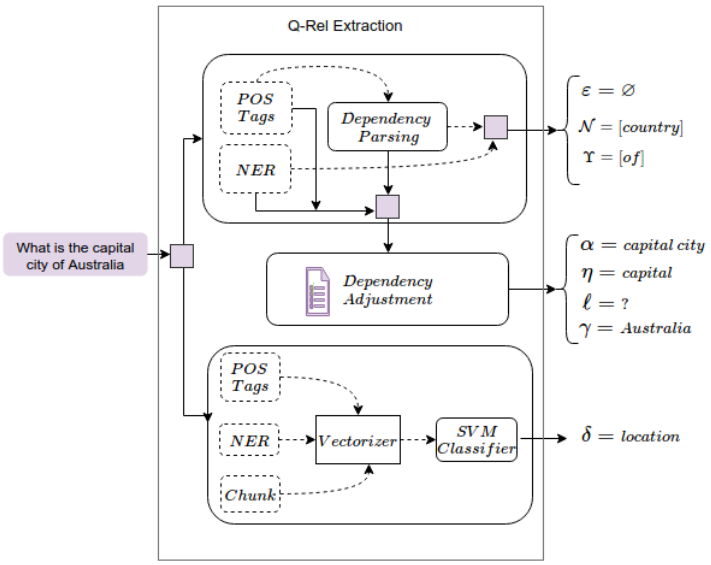

Figure 5: Generation of a Q-Rel

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 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 while 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 complementary 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 $\mathcal{N}$, 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*

[16] https://github.com/nausheenfatma/QuestionClassification
[17] http://trec.nist.gov/data.html

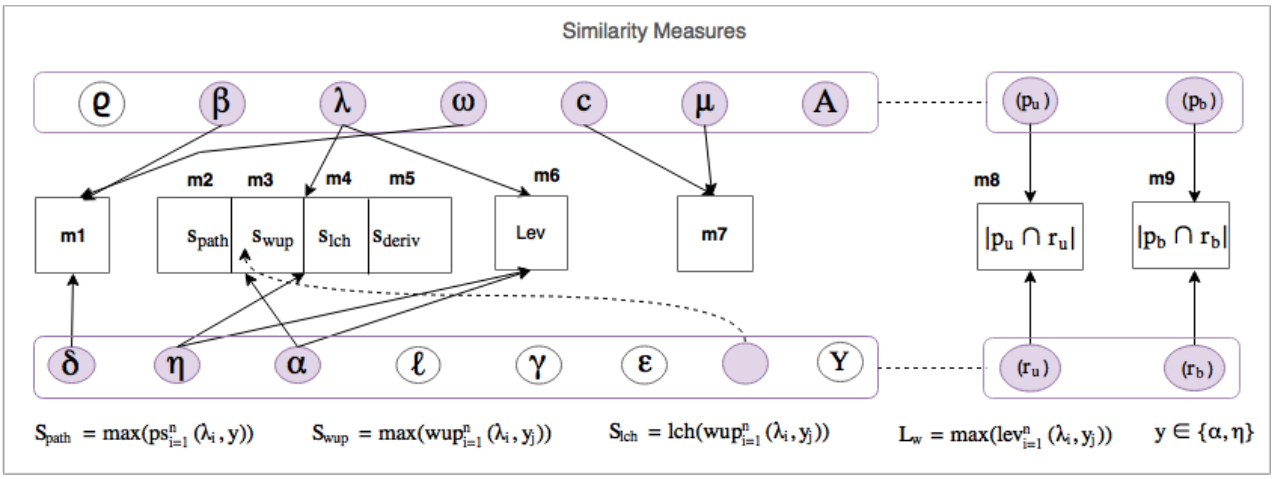

Figure 6: Similarity measures : S_{path} - Wordnet path similarity, S_{wup} - Wu-Palmer Similarity, S_{lch} - Leacock-Chodrow similarity, L_w - Levenstein weight obtained from the levestein similarity (Lev), p_u - Property unigrams, r_u - query relation unigrams, p_b - Property bigrams, r_b - query bigrams

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* and 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 5

After producing the Q-Rel we maintain the associated annotations related to the POS sequence and the Bag of words features.

4.3 Similarity Measurement

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 6 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.

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 [6]. 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:

$ps(r_1, r_2) = 2 * max\_depth - len(r_1, r_2)$

where max_depth is a constant representing the maximum depth of the Wordnet graph. In figure 6 the *ps* is used to obtain values of $m1$ and $m2$.

Wu-Palmer Similarity (m3) [29]:
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, not necessarily closest to the two senses. If multiple candidates for the LCS exist, those whose shortest path to the root node is the longest will be selected. Generally, the longer path is chosen for the calculations in situations where the LCS has multiple paths to the root.

Leacock-Chodorow Similarity (m4) [14]:
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 $Max\_LCS = 3.6375$.

Derivational_forms (m5):
Derivational forms of a word are terms belonging to different syntactic categories but have the same root form and a 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 would miss this relationship. This measure is used to produce the measure *m5* in Figure 6.

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

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

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:

Table 1: Performance Evaluation

		Cumulative Frequency at Rank Positions						Mean Precision @k		Recall @k	F-Measure
Num Properties	Total	Rank#1	Rank#2	Rank#3	Rank#4	Rank#5	Rank#10	#1	#10	#10	#10
1 Property	285	136	154	174	190	199	212	47.72%	55.69%	74.39%	63.70%
2 Properties	82	32	34	37	39	40	51	39.02%	43.63%	62.20%	51.29
		24	31	40	43	44	55	29.27%	39.69%	67.07%	49.87
3 Properties	9	0	1	1	1	1	1	0.00%	11.11%	11.11%	11.11%
		2	3	3	4	4	4	22.22%	30.55%	44.44%	36.21%
		3	3	3	5	5	6	33.33%	38.89%	66.67%	49.12%

$$lev(a_1, a_2) = \begin{cases} 1\,, \ if\ lev_{sim}(a_1, a_2) > 0.75\ \&\ a_1 \subseteq\supseteq a_2 \\ 0\,, \text{else} \end{cases}$$

The value 0.75 is obtained from an evaluation of words whose verb and noun forms give different lemma forms. A list of these words can be found in our github repository provided hereafter.

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 number unique objects as o. We first define a ratio $\mu = \frac{s}{o}$. We then use this ratio to penalize a value obtained from the total number of instances as follows: $\frac{c*n}{\sum_i c_i} * \mu$

Unigrams and Bigrams (m8,m9):
We obtain normalized values related to the size of the intersection between two pairs of unigrams $p_u \& r_u$ as well as bigrams $p_b \& r_b$ 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 bigram are derived from the sequence of the POSs in the sentences. The length of the intersection 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:

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

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

5 EVALUATION

5.1 Experiment Setup

For our evaluation we used the QALD-5 dataset [27] which consists of over 404 questions together with the corresponding formal queries (SPARQL) to be applied against the DBpedia ontology [1, 3]. We did not used higher versions of QALD because our aim is to see the performance of relation linking tool in contrast with the overall performance of QA systems. In later versions of QALD, not many QA systems have participated. The 28 questions from total were out of scope and had no corresponding SPARQL query. Since for our work, we focus on providing independent and reusable tool that identifies the URI of a property for pipelining in QA systems, we annotate the questions with the properties mentioned in the SPARQL queries to form the evaluation dataset.

The 376 viable questions are grouped into three categories based on the number of properties required within the SPARQL queries. A total of 285 questions require only one single property to be matched within the query. A further 82 questions require two properties to be matched and 9 of the question had 3 properties. We evaluate against the properties within the SPARQL queries w.r.t 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[18] and detailed description of practical implementation is online at the project wiki link: https://github.com/mulangonando/ReMatch/wiki.

5.2 Results

Table 1 illustrates our empirical results. For insightful understanding, The QALD questions are grouped into three categories. The first row of the table describe the categories of the questions which contains only one relation (1 Property) for example *Who is the wife of Donald Trump*. For such questions, our tool has precision of 47.72 percent when the correct result is at first position in final list of answers and 55.69 percent as average precision for top 10 properties. Recall and F-Measure are also considerably high for this type of questions, with values equal to 74.39% and 63.70% percent respectively.

For questions such as *How many people live in the capital city of Australia*, the expected properties from DBpedia are two: *populationTotal* and *capital*. For such questions (2 Properties), our tool provides overall precision of 39.02 percent for the relation occurred at first instance. In our example question, *populationTotal* represents the first relation of the input question. For questions such as *Which telecommunications organizations 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 considerably.

We analyzed overall precision and recall values of the systems which took part in QALD-5 challenge. We can observe that if our tool is used as a component to identify relations for input questions, it would not decrease overall precision and recall values of many of the systems like SemGraphQA, YodaQA, QAnswer as our tool has higher precision and recall value from many of these systems [27]. Besides Xser and AskNow, all other QA systems evaluated over QALD-5 have lower precision than 0.40 [7, 27]. Furthermore,

[18] https://github.com/mulangonando/ReMatch

we are not aware of any other independent relation linking tool with which we can compare our performance.

Overall, while aiming for component based QA systems using frameworks like Qanary[5] or OKBQA[19], our tool would improve on the overall performance of the QA system for the questions having single and double relation. However, for the questions with three relations, our tool would affect the overall performance of the QA system negatively.

6 CONCLUSIONS AND FUTURE WORK

This paper 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 and when the Question is too short to represent considerable amount of information in the *Q-Rel* such as seen with the question: *Give me all Cosmonauts*. The major challenges in such scenarios is the lack of tailored text corpora that can be used to train a learning algorithm.

As future work, We target to fine tune the similarity measures by learning the weights through known least squares optimization approaches and evaluate the results against our current results as a benchmark. We have identified the current use of embeddings 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 better precision. In cases where we have a recall value but the desired property has not been ranked top of the results, an approach would be determined to better rank the final result set.

7 ACKNOWLEDGMENT

This project has received funding from the DAAD (Deutscher Akademischer Austauschdienst).

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. DBpedia: A Nucleus for a Web of Open Data. Springer, Berlin, Heidelberg, 722–735.

[2] Romain Beaumont, Brigitte Grau, and Anne Laure Ligozat. 2015. SemGraphQA@QALD-5: LIMSI participation at QALD-5@CLEF. *CEUR Workshop Proceedings* 1391, 1 (2015).

[3] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web* 7, 3 (2009), 154–165.

[4] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. (6 2015). http://arxiv.org/abs/1506.02075

[5] Andreas Both, Dennis Diefenbach, Kuldeep Singh, Saedeeh Shekarpour, Didier Cherix, and Christoph Lange. 2016. Qanary–a methodology for vocabulary-driven open question answering systems. In *International Semantic Web Conference*. Springer, 625–641.

[6] Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Computational Linguistics* 32, 1 (mar 2006), 13–47.

[7] Mohnish Dubey, Sourish Dasgupta, Ankit Sharma, Konrad Höffner, and Jens Lehmann. 2016. AskNow: A Framework for Natural Language Query Formalization in SPARQL. (2016), 300–316.

[8] Claudia Felser, Theodore Marinis, and Harald Clahsen. 2003. Children's Processing of Ambiguous Sentences: A Study of Relative Clause Attachment. *Language Acquisition* 11, 3 (jul 2003), 127–163.

[9] Óscar Ferrández, Christian Spurk, Milen Kouylekov, Iustin Dornescu, Sergio Ferrández, Matteo Negri, Rubén Izquierdo, David Tomás, Constantin Orasan, Guenter Neumann, and others. 2011. The QALL-ME framework: A specifiable-domain multilingual question answering architecture. *Web semantics: Science, services and agents on the world wide web* 9, 2 (2011), 137–145.

[10] Katrin Fundel, Robert Küffner, and Ralf Zimmer. 2007. RelEx—Relation extraction using dependency parse trees. *Bioinformatics* (2007).

[11] Daniel Gerber and A-C Ngonga Ngomo. 2011. Bootstrapping the linked data web. In *1st Workshop on Web Scale Knowledge Extraction @ISWC*.

[12] Sherzod Hakimov, Hakan Tunc, Marlen Akimaliev, and Erdogan Dogdu. 2013. Semantic question answering system over linked data using relational patterns. *Proceedings of the Joint EDBT/ICDT 2013 Workshops on - EDBT '13* (2013), 83.

[13] Jin-Dong Kim and K Bretonnel Cohen. 2013. Natural language query processing for SPARQL generation: A prototype system for SNOMED CT. In *Proceedings of biolink*. 32–38.

[14] Claudia Leacock and Martin Chodorow. 1998. Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical database* 49, 2 (1998), 265–283.

[15] Jung-Tae Lee, Sang-Bum Kim, Young-In Song, and Hae-Chang Rim. 2008. Bridging lexical gaps between queries and questions on large online Q&A collections with compact translation models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ACL, 410–418.

[16] Xin Li and Dan Roth. 2002. Learning question classifiers. *Proceedings of the 19th International Conference on Computational linguistics* 1, 1 (2002), 1–7.

[17] Percy Liang. 2013. Learning Compositional Semantics. 1998 (2013), 1–7.

[18] Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. 2017. Neural Network-based Question Answering over Knowledge Graphs on Word and Character Level. In *Proceedings of the 26th International Conference on World Wide Web - WWW '17*. ACM Press, New York, New York, USA, 1211–1220.

[19] Edgard Marx, Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Konrad Höffner, Jens Lehmann, and Sören Auer. 2014. Towards an open question answering architecture. In *Proceedings of the 10th International Conference on Semantic Systems*. ACM, 57–60.

[20] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. PATTY: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. 1135–1145.

[21] Kamel Nebhi. 2013. A rule-based relation extraction system using DBpedia and syntactic parsing. *CEUR Workshop Proceedings* 1064 (2013), 1–6.

[22] Axel-Cyrille Ngonga Ngomo, Lorenz Bühmann, Christina Unger, Jens Lehmann, and Daniel Gerber. 2013. Sorry, i don't speak SPARQL: translating SPARQL queries into natural language. (2013), 977–988.

[23] John Prager, Jennifer Chu-Carroll, and Krzysztof Czuba. 2002. Statistical answer-type identification in open-domain question answering. *Proceedings of the second international conference on Human Language Technology Research* - (2002), 150.

[24] Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. 2016. Transforming Dependency Structures to Logical Forms for Semantic Parsing. (2016).

[25] Kuldeep Singh, Ioanna Lytra, Maria-Esther Vidal, Dharmen Punjani, Harsh Thakkar, Christoph Lange, and Sören Auer. QAestro – Semantic-based Composition of Question Answering Pipelines. *DEXA 2017*.

[26] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Template-based Question Answering over RDF Data. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12)*. New York, NY, USA.

[27] Christina Unger, Corina Forascu, Vanessa Lopez, Axel Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. 2015. Question answering over linked data (QALD-5). *CEUR Workshop Proceedings* 1391 (2015).

[28] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*. AAAI Press, 1112–1119.

[29] Zhibiao Wu and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Morristown, NJ, USA.

[30] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957* (2016).

[31] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of machine learning research* Feb (2003).

[32] Huang Zhiheng, Marcus Thint, and Zengchang Qin. 2008. Question Classification using HeadWords and their Hypernyms. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing* October (2008), 927–936.

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