© Jurnal Nasional Teknik Elektro dan Teknologi Informasi
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License
DOI: 10.22146/jnteti.v13i2.9184

Entity and Relation Linking for Knowledge Graph Question Answering Using Gradual Searching

Adila Alfa Krisnadhi[1], Mohammad Yani[2], Indra Budi[3]

[1] Computer Science Study Program, Faculty of Computer Science, Universitas Indonesia, Depok, Jawa Barat 16424, Indonesia
[2] Program Studi Rekayasa Perangkat Lunak, Jurusan Teknik Informatika, Politeknik Negeri Indramayu, Indramayu, Jawa Barat 45252, Indonesia
[3] Information Systems Study Program, Faculty of Computer Science, Universitas Indonesia, Depok, Jawa Barat 16424, Indonesia

[Received: 10 August 2023, Revised: 12 October 2023, Accepted: 18 March 2024]
Corresponding Author: Mohammad Yani (email: mohammad.yani@polindra.ac.id)

ABSTRACT — Knowledge graph question answering (KGQA) systems have an important role in retrieving data from a knowledge graph (KG). With the system, regular users can access data from a KG without the need to construct a formal SPARQL query. KGQA systems receive a natural language question (NLQ) and translate it into a SPARQL query through three main tasks, namely, entity and relation detection, entity and relation linking, and query construction. However, the translation is not trivial due to lexical gaps and entity ambiguity that may occur during entity or relation linking. This research proposed an approach based on multiclass classification of NLQ whose entity occurrences are detected into categories based on KG relations to address the lexical gap challenge. Next, to solve the entity ambiguity challenge, this research proposed a three-stage searching procedure to determine appropriate KG entities associated with the NLQ entities, given the correspondence between the NLQ and a particular KG relation. This three-stage searching consisted of text-based searching, vector-based searching, and entity and relation pairing. The proposed approach was evaluated on the SimpleQuestions and LC-QuAD 2.0 datasets. The experiments demonstrated that the proposed approach outperformed the state-of-the-art baseline. For the relation linking task, the proposed approach reached 89.87% and 74.83% recall for the SimpleQuestions and LC-QuAD 2.0, respectively. This approach also achieved 91.74% and 61.96% recall on the entity linking tasks for the SimpleQuestions and LC-QuAD 2.0, respectively.

KEYWORDS — Entity Linking, Relation Linking, KGQA System, Knowledge Graph.

I. INTRODUCTION

Research on question answering (QA) systems is still interesting and massive in a natural language community. There are three kinds of QA systems: information retrieval question answering (IRQA), knowledge graph question answering (KGQA), and hybrid question answering (HQA) [1]. The difference between IRQA and KGQA is in the data used; IRQA uses text as the data, while KGQA uses a knowledge graph (KG) rather than text. HQA uses a combination of KG and text at the same time.

Research on KGQA has remained challenging in the last decade. A KG is a collection of statements of facts, given as triples, each of which describes entities interlinked by a relation. Here, an entity in a KG can represent an object, event, or concept in the real world. To retrieve data expressed by those statements, one traditionally needs to write a formal query in a formal language like SPARQL, which may be challenging for lay users. Here, instead of SPARQL querying, a KGQA system aims to help lay users by allowing data retrieval from a KG using a natural language question (NLQ) as an input. The task of formulating a SPARQL query that expresses the input NLQ is accomplished by the KGQA system automatically.

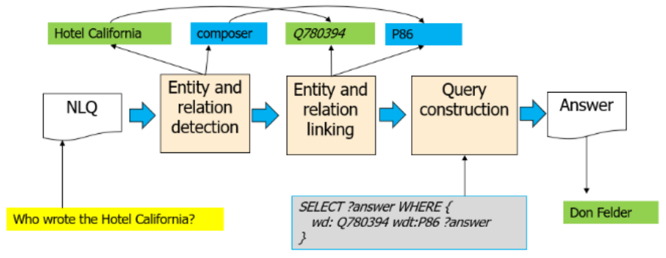
The existing KGQA system uses a query constructor to retrieve data from a KG. The query constructor requires proper entities and relations and the position of entities and relations obtained by the entity and relation linking to get the correct answer. However, the existing entity and relation models only provide a set of entities and relations but do not provide the position of entities and relations in the triple. Therefore, the query constructor should try all possibilities of the position of entities and relations. A model proposed by [2] was used to address the issue. This research focused on the entity and relation linking tasks by utilizing the result of the entity detection proposed by [2].

II. KNOWLEDGE GRAPH QUESTION ANSWERING

Technically, a KGQA system translates an NLQ into a SPARQL query that can then be used to retrieve data through three main tasks, namely, entity and relation detection, entity and relation linking, and query construction. Using SPARQL to retrieve data in a KG is not trivial. It requires the system to run some tasks. The first task is to detect entities and relations in the question. These entities and relations must then be linked to into entity and relation in the KG. The final task is to construct a correct query by using the correct entities and relations to yield the answer.

Figure 1 illustrates how such an end-to-end KGQA system works. Given a question (q) “Who wrote the Hotel California?” entity and relation detection task extracts entity(s) and relation(s) mentioned in the question. In this example, the system obtains “Hotel California” and “composer” as the entity and relation, respectively. Entity and relation linking task links entity(s) and relation(s) obtained by the former task. This task outputs “Q780394” and “P86” as the entity ID and relation ID, respectively. Query construction constructs a formal query to retrieve data from the KG using the list of entity(s) and relation(s) obtained by the entity and relation linking task [3]. The constructed query at the task is:

```
SELECT ?answer WHERE {
   wd:Q780394 wdt:P86 ?answer .
}
```


Hotel California
composer
Q780394
P86
NLQ
Entity and relation detection
Entity and relation linking
Query construction
Answer
Who wrote the Hotel California?
Don Felder

```
SELECT ?answer WHERE {
   wd: Q780394 wdt:P86 ?answer
}
```

Figure 1. Running example of KGQA system.

The query is written in SPARQL syntax. SPARQL is a language standard used to access data in resource description framework (RDF) form stored in a KG [4]. RDF is a framework used to express information about resources such as humans and documents [5].

Of the three tasks, the last is rather straightforward once the entities and relations for the given NLQ are obtained. Meanwhile, the first two are more difficult. A previous work was proposed to address the issues of the entity detection task [2]. The first task (entity and relation detection) addressed an issue in defining the position of the detected entities in a triple for a given [2]. The model extracts entities mentioned in the question using a position-based pattern approach. With this approach, the model outputs a set of entities and the position in the triple. The result of the model can assist the entity and relation linking task in finding the proper entities and relations in the KG.

This research focused on the remaining tasks, namely entity and relation linking. In addition, based on the proposed approach, the relation detection task can be skipped entirely because the relation linking task can already associate a KG relation to an NLQ once all entities are detected.

Two main challenges need to be addressed with respect to entity and relation linking: entity ambiguity and lexical gap [6]. Entity ambiguity refers to the case in which entities in the NLQ can be associated with a KG entity with the same name but different meanings (also known as polysemy) [7], [8]. Meanwhile, lexical gap refers to the case in which the surface form of entities or relations in the question completely differs from the surface form of entities and relations in the KG. Figure 2 and Figure 3 illustrate these challenges with more concrete examples.

In Figure 2, there are three entities named "Ronaldo." Each entity has the same label but a different meaning. Entities with ID Q529207, Q54588254, Q21027936 represent "Ronaldo" as a Brazilian footballer, a musician, and a film title, respectively. However, the entity mentioned in the questions is "Ronaldo," a Brazilian footballer (Q529207). Even though this issue is easy to address for humans, it is not for machines. This issue reaches 60% of other issues on KGQA system [6].

Figure 3 illustrates the issue of lexical gap in the relation linking task. In the example, the relation "born" is expressed in a different lexical in the KG, namely, "place of birth." This issue leads to a mistake linking in the relation linking task. The lexical gap issue is the most occurred in the KGQA systems, namely 65.7% [6].

At least there are two approaches to addressing the issues. In the first approach, entity and relation linking are performed independently [9]–[11]. Here, whatever results obtained from the two tasks are used to construct an appropriate KG query. The drawback of this approach is that

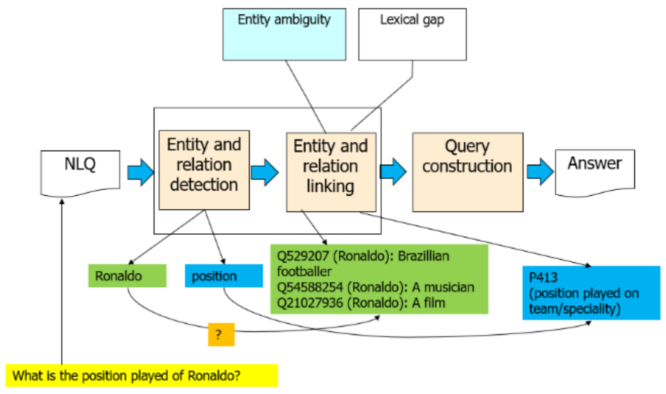

Entity ambiguity
Lexical gap
NLQ
Entity and relation detection
Entity and relation linking
Query construction
Answer
Ronaldo
position
Q529207 (Ronaldo): Brazillian footballer
Q54588254 (Ronaldo): A musician
Q21027936 (Ronaldo): A film
P413 (position played on team/speciality)
?
What is the position played of Ronaldo?

Figure 2. Illustration of entity ambiguity issue.

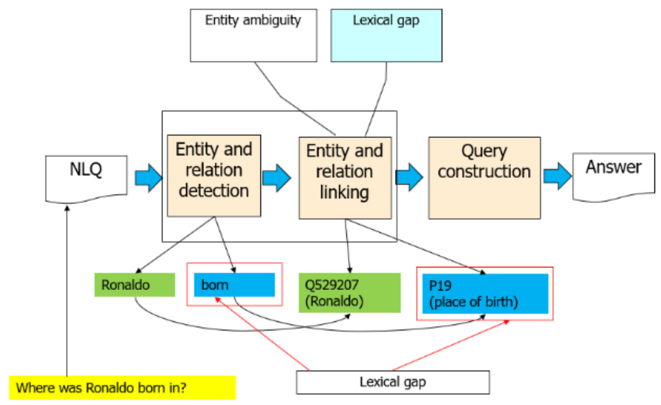


Figure 3. Illustration of lexical gap issue.

the independence between entity linking and relation linking causes the former to fail to exploit possibly useful relation information that can be obtained from the latter. Hence, mismatches can potentially occur due to ambiguities in entity names/mentions.

In the second approach, entities and relations are paired. The aim is to simultaneously find both a KG entity and relation that can be matched with the entity and relation mentioned in the input NLQ. Previous works following this approach did not use additional semantic information that can be obtained from entity descriptions and name aliases available in the KG [12]–[14]. It becomes a disadvantage since the process in principle must exhaustively consider many possible entity-relation pairs in the KG.

As an alternative to the above approaches, it is proposed that relation linking first be performed before entity linking. More precisely, the relation linking is modeled as a multiclass classification whose target comes from KG relations and whose input is a masked NLQ, i.e., the input NLQ but with all entity mentions masked using the special [ENT] token. This approach exploits the fact that entity mentions have been detected in a preceding entity detection step (the solution of which has been described in the previous work [2]). Moreover, once an input NLQ has been matched to the appropriate relation(s), entity linking is done via a three-step search procedure: text-based searching, vector-based searching, and entity and relation pairing. The main contribution of this approach is that the approach can disambiguate entities in the KG by pairing entity and relation well and can identify the proper relation in the KG even when it is expressed in a different surface form from the relation in NLQ.

The rest of this paper is structured as follows. The Methodology section presents the proposed approach to

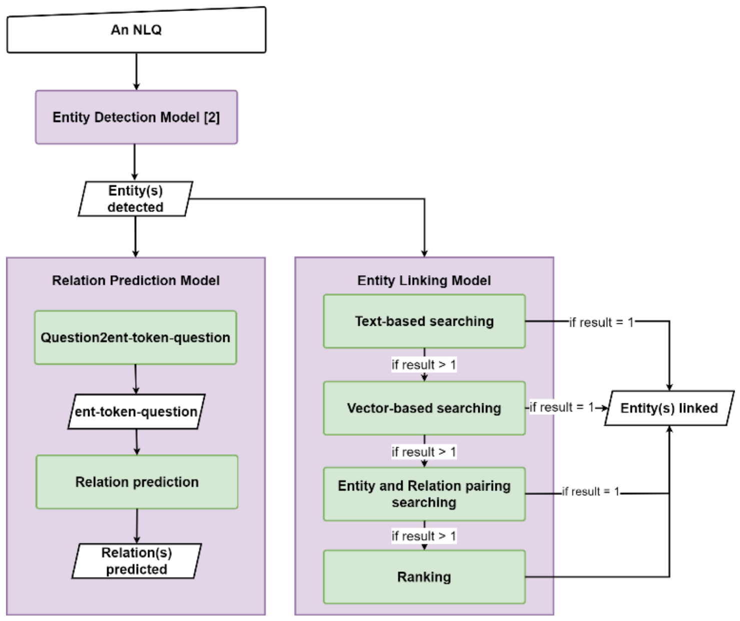


Figure 4. General architecture of the proposed approach.

address the issues of entity and relation linking as described above. The Result and Discussion section presents the experiment result compared with another approach. The final section concludes this work and introduces future work.

III. METHODOLOGY

Three existing methods were used to implement the entity and relation linking task, namely, sequential, parallel, and joint entity-relation approaches. The sequential approach worked by performing entity linking sequentially and then relation linking. The parallel approach performed both tasks at the same time. The joint entity-relation approach paired entity-relation to link them into a KG [10]–[16].

The proposed approach consisted of two subtasks: relation linking and entity linking tasks. The tasks aim to link an entity(s) and relation(s) extracted from the questions to an entity(s) and relation(s) in the KG. The tasks were sequentially performed. The proposed approach performed relation linking first and entity linking as the step after. It differs from the existing sequential approach that performs entity linking before relation linking. This approach was used due to the entity ambiguity issue occurring using the existing sequential approach, which failed to map the relation as the pair of an entity. Moreover, the lexical gap issue also occurred in the existing sequential approach, namely 65.7% [6].

The proposed approach used Elasticsearch as the local search engine. Two kinds of local search engines were proposed, namely, text-based search engines and vector-based search engines. These search engines encode entity ID and label of Wikidata from Wikidata dumps into an index form in Elasticsearch [17]. The difference is in the form of data stored in Elasticsearch; vector-based local search engine encodes data in a vector rather than text.

Figure 4 depicts the general architecture of the proposed approach for entity and relation linking for KGQA systems. For instance, an NLQ was given to this system. First, the entities mentioned in the NLQ were extracted using the entity detection model proposed by [2]. The model also provided information about the position of entities in the triple. The position information provided was about the entity position in the query and the entity position in the triple. The entity position in the query means that in what triple the entity exists, the first, second, or third triple. Meanwhile, the entity position in a triple show whether it is contained in the head or tail.

As depicted in Figure 4, the entity name extracted from the question was used to create ent-token-question where the token(s)/word(s) that corresponded to the entity name was converted into [ENT] token. A bidirectional encoder representation from transformer (BERT) model was used to implement a classification task in predicting a relation. The model of entity detection in [2] was used to link the entities extracted from a question. There were three approaches in linking the entity: text-based searching, vector-based searching, entity-relation pairing searching. The search processes were done sequentially. The subsequent process was run when the number of entity candidates was more than one. At the last subtask, the cosine similarity was employed to assess the similarity of all entity candidates.

A. RELATION LINKING

The purpose of this task is to link a relation used in the question to a relation in the KG. Conventionally, this task receives an input of relation detection task in the form of word/token identified as the relation in the question.

A multiclass classification was used to model the relation linking task. The input of the problem was a masked NLQ called ent-token-question. An ent-token-question is like the usual input NLQ, but with the occurrence of entity mentions masked by the special [ENT] token. The classification target consisted of KG relations, pairs of such relations, and sets containing three of such relations. In practice, not all possible combinations of relation pairs/triples were considered, but only those relation pairs or triples associated with the same question, according to the dataset used to build the classification model. Converting original questions into ent-token-question was done in the preprocessing stage.

The main idea of using ent-token-question is that the model can learn many expressions used to state relations on SimpleQuestions and LC-QuAD 2.0 dataset. Moreover, the main purpose of masking the entity mentions in the NLQ is to

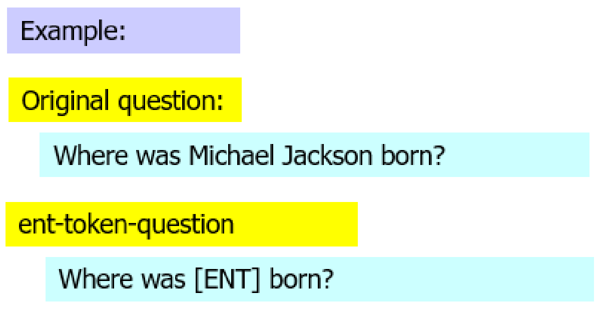


Figure 5. Illustration of conversion of question to ent-token-question.

emphasize the relation expression without depending on the entity mentions/names in the question. Figure 5 gives an example of an ent-token-question "Where was [ENT] born?" representing the relation "place of birth" (P19). Before training, a dataset containing the original questions was converted into a form that contains ent-token-question and ID relation in the first and the second column, respectively. The first column was used as the input feature of the model while the second one was used as the label.

Furthermore, the use of question words "what," "when," and "where" was also considered to indicate the relation mentioned in the question. Still in the same instance, question "Where was [ENT] born?" possibly resulted "place of birth" (P19) and "date of birth" (P569). The proposed approach could identify "place of birth" (P19) as the correct relation mentioned in the question by considering the question word used in the question.

A fine-tuned BERT [18] was used to predict a relation of a given question. Figure 6 depicts the input and output of the model in predicting the ID relation of a given question. Due to the limitation of the computing resources, the model simple transformers were used for training [19]. Simple transformers simplify the original face library [20] without omitting the substance.

B. ENTITY LINKING

This task consists of three subtasks: text-based searching, vector-based searching, and entity and relation pairing tasks. Text-based searching finds entity candidates by matching the entity label name in the question and the entity label name in the KG. For instance, all entities named "Joe Biden" will be returned when an entity named "Joe Biden" is quired.

Vector-based searching is used if the text-based searching does not output any return value or outputs more than one entity. This approach matches the vector value of the entity mentioned in the question with the entity information in the KG. The encoded information of entities in the KG is the label name, description, and alias name; meanwhile, the encoded information of questions is all phrases or words mentioned in the question. Still in the same example for searching "Joe Biden," the vector-based searching will return all entities that relate to the mention in the question. With the same example, text-based searching and vector-based searching obtains two different sets of entities, namely, [Q6279, Q65053339] and [Q6279, Q65053339, …, Q129756], respectively. In this example, using vector-based searching obtains a set size bigger than text-based searching. A big set size of vector-based searching has both advantages and disadvantages. The pros are that the set can contain entity(s) that does not exist in the set of text-based searching. While the cons are that the use of this set makes the search space wider. A universal sentence encoder

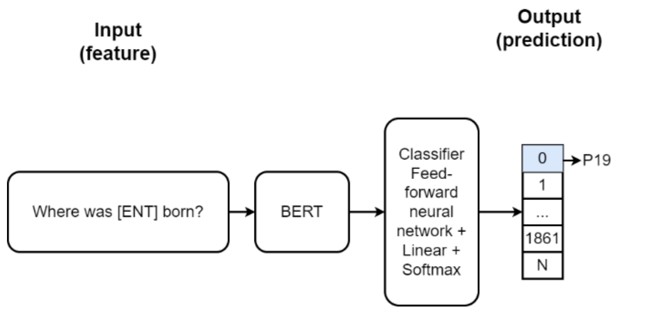


Figure 6. Relation prediction using classification multiclass task.

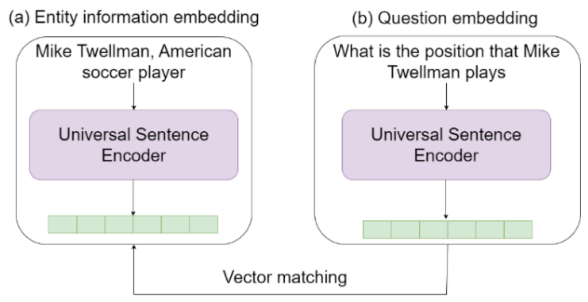

(a) Entity information embedding
(b) Question embedding
Mike Twellman, American soccer player
What is the position that Mike Twellman plays
Universal Sentence Encoder
Universal Sentence Encoder
Vector matching

Figure 7. Vector-based searching.

[21] was used for encoding information. Figure 7 depicts how vector matching runs.

Entity and relation pairing was used if the vector-based searching approach returns more than one entity (including ambiguous entities). This approach aims to check whether an entity has a particular relation that corresponds to the entity or not. If the return value was True, the entity was assumed as the proper entity and vice versa. However, if some entities had a particular relation found, the top-1 of entity candidates was chosen. For instance, the question "What is the position that Mike Twellman plays" yields a set of entity candidates [Q6849115, Q5921964, ..., Q5059480]. So, for each entity candidate was checked if the entity had a relation obtained from the relation linking task. If yes, it was assumed as the correct entity mentioned in the question. If not, the entity was eliminated.

In the subtask of relation linking, Question2ent-token-question converted a question into ent-token-question format. For example, the question "What is the position that Mike Twellman plays" is converted into "What is the position that [ENT] plays." The token "Mike Twellman" is converted into "[ENT]" as the token is an entity. The data construction task collected all ent-token-question derived by the Question2ent-token-question task. The constructed data contained two parts, namely ent-token-question and labels. The machine then trained the data using multiclass classification using the BERT model.

Three subtasks were proposed for entity linking task. Text-based searching looked for the proper entities by matching the string using exact matching and substring matching. Vector-based searching was used when the result of text-based searching did not find any matched strings. This task measured the similarity of two embedded string in the vector space. The final subtask was the entity-relation pairing task. This task searched for entity(s) that had a pair of a relation obtained from the relation linking task.

IV. RESULTS AND DISCUSSION

This section presents the experiment setting, evaluation method, and result and discussion of this research. The result of the experiment was compared to a baseline for entity and relation linking that used the same dataset and KG.

In the evaluation section, SimpleQuestions and LC-QuAD 2.0 are used as the existing KGQA systems remain drawbacks on the datasets described in the Introduction section. Moreover, this research only addressed the entity ambiguity and lexical gap issues in the KGQA system. This paper only addresses the issues for questions containing three triples for the maximum.

A. SIMPLEQUESTIONS AND LC-QUAD 2.0

SimpleQuestions contains a set of simple questions. A simple question is a question containing only one triple. A triple consists of a subject, predicate, and object [5]. SimpleQuestions has two versions, namely freebase version [22] and Wikidata version [23]. Wikidata SimpleQuestions contains 27,924 questions.

LC-QuAD 2.0 is a dataset that contains complex questions. A complex question is a question containing more than one triple. The dataset consists of ten different question types, including simple and complex questions such as multifact questions. This dataset contains 30 million questions and the query ground truth. The ground truth contains SPARQL query that represents the answer to the question [24].

B. EXPERIMENT SETTING

This section explains the experimental setting used in the evaluation of the proposed approach, namely, entities extractor tools, machine specification, dataset, KG, and data and parameter configuration used.

The entity extractor tool proposed called the position-based pattern (PBP) method [2] was used to extract the entity name from the question. Position-based pattern represents a pair (χ, ℓ) where χ denotes a head or tail position and ℓ is a token-based position. The method extracted an entity(s) mentioned in the question and gave the entity position in the triple. Triple is a form used by KGs to represent a fact.

The output of PBP is a PBP set *Cq*. Th question *q* “Where was John Morris Russel born?” was considered to illustrate the position-based pattern set. Token “John Morris Russell” was positioned in [2, 3, 4] indexes (started by 0) in *q*. Thus, the PBP set of *q* is as follows.

$$Cq = \{0: (head, [2,3,4])\}$$

the integer 0 means that the token “John Morris Russell” is in the first triple. The label of *head* means that “John Morris Russell” is a head pattern; head pattern is a pattern where an entity is positioned in the subject. For the classification task, *Cq* is encoded to be *0:head:ent:2_3_4*. The encoded *Cq* is used as the label of *q* in the multiclass classification task.

The prediction of position-based pattern used a multiclass classifier that ran under the transformer model. The input of the model was a question, and the output of the model was a predicted PBP of a given question. The output of PBP was used as the input of the proposed approach for linking the entity and relation mentioned in the question and the entity and relation in the KG. Before using the PBP for the entity linking tasks, a preprocessing was used. The preprocessing converted a set of integers of PBP into words/tokens that corresponded to the position of words/tokens in the question. The set of words/tokens obtained was then converted into ent-token-question form.

TABLE I
DISTRIBUTION DATA OF SIMPLEQUESTIONS AND LC-QUAD 2.0

Dataset	Training Data	Validation Data	Testing Data	Total
SimpleQuestions	19,481	2,821	5,622	27,924
LC-QuAD 2.0	22,132	3,306	6,383	31,821

This research used NVIDIA GPU GeForce GTX with 24 GB memory for training data. Meanwhile, a CPU 3.0 GHz Intel(R) i7-5960X with 128 GB memory was used for data preprocessing. The experiment used SimpleQuestions over Wikidata [23] rather than the original SimpleQuestions [22] as the original SimpleQuestions did not use Wikidata as the KG. In addition, this research used LC-QuAD 2.0 [24] over Wikidata for complex questions. Wikidata contains millions of entities that anyone can edit. Data in Wikidata is stored in RDF format [25].

This experiment used the default setting to split the dataset into training, validation, and testing data. This splitting composition was used as it can be fairly compared with Falcon 2.0. Table I presents the distribution of the SimpleQuestions and the LC-QuAD 2.0 dataset. The LC-QuAD 2.0 dataset does not assign a separate validation set; hence the validation data obtained was 13% from the data training.

This experiment used Wikidata as the KG. The facts used in this experiment were facts from the Wikidata dump. The dump was downloaded from https://dumps.wikimedia.org/ in 2019. The dump contained about 49 million and 6 thousand entities and relations, respectively.

C. EVALUATION METRIC

The proposed model outputted entities and relations and their positions in a triple. However, the existing entity and relation-linking model, such as Falcon 2.0, did not output the position of entities and relations in a triple. Therefore, to fairly compare the proposed approach and Falcon 2.0, the position information of entities and relations in a triple in this evaluation was not used.

The evaluation used a recall-based measure of the predicted entities and relations. The motivation of the recall-based measure use is summarized below.
Intuition:

a. the predicted entity linking and relation linking can output multiple entities or relations;
b. the ground truth also contains multiple entities or relations;
c. constructing a query of the predicted entity linking and relation linking that corresponds to the proper query is not trivial;
d. therefore, it is preferable if the result of entity linking and relation linking covers as a large proportion of ground truth entities or relations as possible;
e. therefore, a recall-based measure is more appropriate.

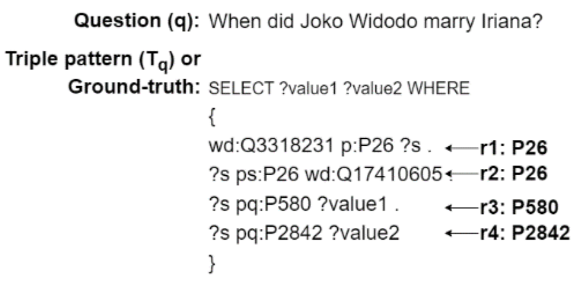
the following formula is used:
The q is a question and Y_q^* is the set of ground truth entities or relations. If the entity linking or relation linking model yields Y_q as the set of predicted entities or relations, the recall-based measure for q is:

$$score(q) = \frac{|Y_q^* \cap Y_q|}{|Y_q^*|}. \quad (1)$$

The model performance was then measured by the following average recall (AR) over testing set $\mathcal{D}$:

TABLE II
COMPARISON OF RECALL OF RELATION LINKING TASK BETWEEN THE PROPOSED APPROACH AND FALCON 2.0

Dataset	Falcon 2.0 (%)	The Proposed approach (%)
Tested on SimpleQuestions	41.33	91.74
Tested on LC-QuAD 2.0	27.75	61.96



Question (q): When did Joko Widodo marry Iriana?

Triple pattern (T_q) or Ground-truth:

```
SELECT ?value1 ?value2 WHERE
{
wd:Q3318231 p:P26 ?s .       ←r1: P26
?s ps:P26 wd:Q17410605 .     ←r2: P26
?s pq:P580 ?value1 .         ←r3: P580
?s pq:P2842 ?value2          ←r4: P2842
}
```

Figure 8. Question example with hidden relation and without keyword indicator on LC-QuAD 2.0.

$$AR(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{q\in\mathcal{D}} score(q). \qquad (2)$$

D. BASELINE MODEL

This experiment employed Falcon 2.0 as the baseline as it outperforms the other works as described in Introduction section. Moreover, other works did not provide any source code or API to replicate the result. In contrast to that, Falcon 2.0 provides an API to replicate the result of given datasets. Thus, in this experiment, Falcon 2.0 was chosen as the baseline.

E. RESULT OF RELATION LINKING

A recall-based measure in (1) was used to evaluate the proposed relation linking method. For instance, if the ground truth and prediction contain {'P19', 'P20'}, the recall score is 100%. However, if the prediction contains {'P19'}, the recall score is 50%.

This evaluation compared the proposed approach (with PBP-based entity detection) with the baseline, namely Falcon 2.0 [14]. The comparison between the proposed approach and Falcon 2.0 can be seen in Table II. From this table, it can be seen that the proposed approach outperformed Falcon 2.0 in recall. Converting original questions to ent-token-question can represent the expression of relation used in the question without depending on the entity name. This method can address the lexical gap issue that occurred in the relation linking task. For instance, the proposed approach can predict the relation "place of birth" (P19) from a given question "What female actor was born in [ENT]."

The proposed approach also outperformed Falcon 2.0 for LC-QuAD 2.0. However, the recall was lower than SimpleQuestions. It was due to LC-QuAD 2.0 having many questions and the relation was not explicitly defined in the questions (hidden relation).

The proposed approach could not address a question with a hidden relation and no keyword indicator in the question. Figure 8 gives a question example with a hidden relation without a keyword that indicates the relation. In that figure, "r1..r4" represents relations used in the question. The relation "spouse" (P26) and "start time" (P580) was not explicitly mentioned in the question, but there was a keyword indicator to refer to the relation, namely, "when" and "marry" for "start time" and "marry," respectively. In this case, this proposed approach could address the issue. However, the hidden relation without a keyword indicator, such as "place of marriage" (P2842), was hard to address using the proposed approach.

TABLE III
THE NUMBER OF QUESTIONS THAT THE PROPOSED APPROACH CAN ANSWER BUT FALCON 2.0 CANNOT

Dataset	Quality
SimpleQuestions	2.866
LC-QuAD 2.0	1.948

TABLE IV
THE EXAMPLE QUESTIONS OF SIMPLEQUESTIONS THAT THE PROPOSED APPROACH CAN ANSWER BUT FALCON 2.0 CANNOT

Question	Ground Truth	The Proposed Approach	Falcon 2.0
Who was born in the city of San Francisco?	p19	p19	p131
What was the cause of death of Yves Klein?	p509	p509	p20

TABLE V
THE EXAMPLE QUESTIONS OF LC-QUAD 2.0 THAT THE PROPOSED APPROACH CAN ANSWER BUT FALCON 2.0 CANNOT

Question	Ground Truth	The Proposed Approach	Falcon 2.0
What are the characters that appear in Nastes?	p1441 AND p674	p1441 AND p674	p953 AND p674
What genre film was the prequel to Zork II?	p156 AND p136	p156 AND p136	p155

Falcon 2.0 had a low recall due to its use of n-gram tiling to match the surface form of the relation in the question and the label name of the relation in the KG. Therefore, Falcon 2.0 could not address the lexical gap issue. For instance, Falcon 2.0 cannot identify the word/token "born" in "Who was born in the city of San Francisco" as a relation to "place of birth" in the KG. Table III, Table IV, and Table V show questions that can be answered by the proposed approach but not by Falcon 2.0 on the relation linking task.

F. RESULT OF ENTITY LINKING

This experiment evaluated the models over SimpleQuestions and LC-QuAD 2.0 datasets. The distribution of data can be seen in Table I. The exact matching was used to evaluate the recall of prediction by matching the entity(s) and order between the prediction and the ground truth. The correct result means when the prediction entity(s) and order are completely matched with the ground truth. For example, the question "Where did Roger Marquis die" has a ground truth {0:{'head':'Q7358590'}}. So, the model obtained the correct prediction if the prediction is {0:{'head':'Q7358590'}} too. This work did not address out-of-vocabulary issues. For example, if an entity "Anas Yani" exists in the question but not in the KG, the entity "Anas Yani" means an out-of-vocabulary issue. For that case, this proposed model found the closest similarity with that name, for instance, "Malik ibn Anas."

In this evaluation, some experiments were conducted by configuring the threshold value so that the entity linking achieved good performance. The experiment showed that the use of a 0.2 threshold could obtain the best recall. Table VI shows the results of the experiments in defining the best threshold value in the entity linking task.

TABLE VI
THRESHOLD VALUE EXPERIMENTS IN ENTITY LINKING

Threshold Value	Recall (%)
0.1	89.86
0.2	89.86
0.3	89.86
0.4	89.86
0.5	89.86
0.6	89.86
0.7	89.86
0.8	89.79
0.9	88.97

TABLE VII
COMPARISON OF RECALL OF ENTITY LINKING TASK BETWEEN THE PROPOSED APPROACH AND FALCON 2.0

Dataset	Falcon 2.0 (%)	The Proposed Approach (%)
Tested on SimpleQuestions	55.69	89.87
Tested on LC-QuAD 2.0	39.70	74.83

TABLE VIII
THE NUMBER OF QUESTIONS THAT THE PROPOSED APPROACH CAN ANSWER BUT FALCON 2.0 CANNOT

Dataset	Quality
SimpleQuestions	2.079
LC-QuAD 2.0	1.916

TABLE IX
THE EXAMPLE QUESTIONS OF SIMPLEQUESTIONS THAT THE PROPOSED APPROACH CAN ANSWER BUT FALCON 2.0 CANNOT

Question	Ground Truth	The Proposed Approach	Falcon 2.0
Where did Vic Frazier die?	q7924799	q7924799	q36687
What is Alain Sutter position?	q503421	q503421	q65602988

TABLE X
THE EXAMPLE QUESTIONS OF LC-QUAD 2.0 THAT THE PROPOSED APPROACH CAN ANSWER BUT FALCON 2.0 CANNOT

Question	Ground Truth	The Proposed Approach	Falcon 2.0
When did Ashton Kutcher divorce Demi Moore?	q164782 AND q43044	q164782 AND q43044	Empty
What is the fashion house of Alexander Mcqueen?	q207939 AND q3661311	q207939 AND q3661311	q17659819

This section compares the proposed approach, added with PBP, with Falcon 2.0. Table VII shows the comparison recall between the proposed approach and Falcon 2.0. The experiment results demonstrated that the proposed approach outperformed Falcon 2.0.

In general, the recall of the proposed approach on entity linking outperformed Falcon 2.0 by about 30% of increasing. The recall of Falcon 2.0 was low as Falcon 2.0 was not enough good in the relation linking task. Therefore, pairing entity and relation done by Falcon 2.0 lead to wrong facts. For this reason, Falcon 2.0 could not disambiguate well the issue of entity ambiguity. For instance, a question "What is Alain Sutter position played in" returns two entities named "Alain Sutter" in the KG, namely, "Alain Sutter" (Q503421) as a Swiss footballer and "Alain Sutter" (Q65602988) as a French politician. Of the two entities, only entity "Alain Sutter" (Q503421) that had a relation "position played on team" (P413). Thus, the entity "Alain Sutter" (Q503421) was the correct entity required. Since Falcon 2.0 failed to obtain the correct relation, Falcon 2.0 could not disambiguate the proper entity required. Table VIII, Table IX, and Table X show questions that can be answered by the proposed approach but not by Falcon 2.0 on the entity linking task.

V. CONCLUSION

The proposed approach using a gradual searching approach can be used for disambiguating entities and addressing the lexical gap issue on the entity and relation linking task for KGQA systems. This proposed approach was evaluated in SimpleQuestions and LC-QuAD 2.0 over Wikidata. The experiment demonstrated that this proposed approach outperformed Falcon 2.0 on the entity and relation linking task for both SimpleQuestions and LC-QuAD 2.0 dataset. This proposed approach reached a recall of about a 40% increase for entity linking and a 30% increase for relation linking.

The proposed approach did not explicitly address a hidden relation issue. However, by considering question words mentioned in a question, the use of ent-token-question approach that implemented universal sentence encoding could predict the closest pair of entity-relation between phrases in the question and triple in the KG.

Future work can consider using a text-based corpus to enrich an entity's information. The corpus is used when the information on an entity from the KG is not completely available. Thus, the entity disambiguation task can work better in choosing the proper entity. Moreover, the use of the owl:sameAs property can be considered to find an entity in the KG that may be written in different lexical.

A hybrid approach can also be interesting work for the entity and relation linking task for KGQA systems. The use of other KGs also can be interesting to consider obtaining any other resources about entities that relate to the question.

CONFLICTS OF INTEREST

The authors declare that there is no conflict of interest.

AUTHORS' CONTRIBUTIONS

Conceptualization, Adila Alfa Krisnadhi; methodology, Mohammad Yani; formal analysis, Adila Alfa Krisnadhi; investigation, Mohammad Yani; writing—original draft preparation, Mohammad Yani; writing—review and editing, Adila Alfa Krisnadhi and Indra Budi; validation, Indra Budi.

ACKNOWLEDGMENT

This research is funded by the Faculty of Computer Science of Universitas Indonesia.

REFERENCES

[1] D. Jurafsky and J.H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition,* 2nd ed. London, England: Prentice Hall, 2009.

[2] M. Yani, A.A. Krisnadhi, and I. Budi, "A better entity detection of question for knowledge graph question answering through extracting position-based patterns," *J. Big Data*, vol. 9, pp. 1–26, Jun. 2022, doi: 10.1186/s40537-022-00631-1.

[3] M. Yani and A.A. Krisnadhi, "Challenges, techniques, and trends of simple knowledge graph question answering: A survey," *Inf.*, vol. 12, no. 7, pp. 1–31, Jul. 2021, doi: 10.3390/info12070271.

[4] E. Prud'hommeaux and A. Seaborne (2008) "SPARQL query language for RDF," [Online], https://www.w3.org/TR/rdf-sparql-query/, access date: 15-Jan-2024.

[5] F. Manola, E. Miller, and B. McBride (2014) "RDF 1.1 primer," [Online], https://www.w3.org/TR/rdf11-primer/, access date: 15-Jan-2024.

[6] K. Höffner *et al.,* "Survey on challenges of question answering in the semantic web," *Semant. Web*, vol. 8, no. 6, pp. 895–920, Aug. 2017, doi: 10.3233/SW-160247.

[7] H. Bast and E. Haussmann, "More accurate question answering on freebase," *CIKM '15, Proc. 24th ACM Int. Conf. Inf. Knowl. Manag.,* 2015, pp. 1431–1440, doi: 10.1145/2806416.2806472.

[8] S. Shin, X. Jin, J. Jung, and K.-H. Lee, "Predicate constraints-based question answering over knowledge graph," *Inf. Process. Manag.,* vol. 56, no. 3, pp. 445–462, May 2019, doi: 10.1016/j.ipm.2018.12.003.

[9] K. Xu, S. Zhang, Y. Feng, and D. Zhao, "Answering natural language questions via phrasal semantic parsing," *Nat. Lang. Process. Chin. Comput.,* 2014, pp. 333–344, doi: 10.1007/978-3-662-45924-9_30.

[10] A. Delpeuch, "OpenTapioca: Lightweight entity linking for Wikidata," 2019, *arXiv: 1904.09131.*

[11] M. Dubey *et al.,* "AskNow: A framework for natural language query formalization in SPARQL," *Proc. 13th Int. Conf. Semant. Web. Latest Adv. New Domains*, 2016, pp. 300–316, doi: 10.1007/978-3-319-34129-3_19.

[12] M. Dubey, D. Banerjee, D. Chaudhuri, and J. Lehmann, "EARL: Joint entity and relation linking for question answering over knowledge graphs," 2018, *arXiv: 1801.03825.*

[13] A. Sakor, K. Singh, and M.-E. Vidal, "FALCON: An entity and relation linking framework over DBpedia," *Proc. ISWC 2019 Satell. Tracks (Posters Demonstr. Ind. Outrageous Ideas) co-located with 18th Int. Semant. Web Conf. (ISWC 2019),* 2019, pp. 265–268.

[14] A. Sakor, K. Singh, A. Patel, and M.-E. Vidal, "Falcon 2.0: An entity and relation linking tool over Wikidata," *CIKM '20, Proc. 29th ACM Int. Conf. Inf. Knowl. Manag.,* 2020, pp. 3141–3148, doi: 10.1145/3340531.3412777.

[15] C. Unger *et al.,* "Template-based question answering over RDF data," *WWW '12, Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 639–648, doi: 10.1145/2187836.2187923.

[16] K. Xu, S. Zhang, Y. Feng, and D. Zhao, "Answering natural language questions via phrasal semantic parsing," *Natural Lang. Process. Chin. Comput.,* 2014, pp. 333–344, doi: 10.1007/978-3-662-45924-9_30.

[17] (2024) The Wikimedia website. [Online], https://dumps.wikimedia.org/wikidatawiki/entities/latest-all.nt.gz, access date: 15-Jan-2024.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv: 1810.04805.*

[19] (2024) The Hugging Face website. [Online], https://huggingface.co/bert-base-cased, access date: 15-Jan-2024.

[20] T. Wolf *et al.,* "HuggingFace's transformers: State-of-the-art natural language processing," 2019, *arXiv: 1910.03771.*

[21] D. Cer *et al.,* "Universal sentence encoder for English," *Proc. 2018 Conf. Empir. Methods Natural Lang. Process., Syst. Demonstr.,* 2018, pp. 169–174, doi: 10.18653/v1/d18-2029.

[22] A. Bordes, N. Usunier, S. Chopra, and J. Weston, "Large-scale simple question answering with memory networks," 2015, *arXiv: 1506.02075.*

[23] D. Diefenbach, T.P. Tanon, K. Singh, and P. Maret, "Question answering benchmarks for Wikidata," *Proc. ISWC 2017 Posters Demonstr. Ind. Tracks co-located with 16th Int. Semant. Web Conf. (ISWC 2017),* 2017, pp. 1–4.

[24] M. Dubey, D. Banerjee, A. Abdelkawi, and J. Lehmann, "LC-QuAD 2.0: A large dataset for complex question answering over Wikidata and DBpedia," *Semant. Web – ISWC 2019,* 2019, pp. 69–78, doi: 10.1007/978-3-030-30796-7_5.

[25] M. Farda-Sarbas and C. Müller-Birn, "Wikidata from a research perspective - A systematic mapping study of Wikidata," 2019, *arXiv: 1908.11153.*