Templet: A Collaborative System for Knowledge Graph Question Answering over Wikidata

Francisca Suárez
fsuarez@dcc.uchile.cl
DCC, Universidad de Chile
Santiago, Chile

Aidan Hogan
ahogan@dcc.uchile.cl
IMFD; DCC, Universidad de Chile
Santiago, Chile

ABSTRACT

We present Templet: an online question answering (QA) system for Wikidata. Templet is based on the collaboratively-edited repository QAWiki, which collects questions in multiple natural languages along with their corresponding structured queries. Templet generates templates from question–query pairs on QAWiki by replacing key entities with identifiers. Using autocompletion, the user can type a question in natural language, select a template, and again using autocompletion, select the entities they wish to insert into the template's placeholders, generating a concrete question, query and results. The main objectives of Templet are: (i) to enable users to answer potentially complex questions over Wikidata using natural language templates and autocompletion; (ii) to encourage users to collaboratively create new templates via QAWiki, which in turn can benefit not only Templet, but other QA systems.

CCS CONCEPTS

• **Information systems → Question answering**; **Graph-based database models**; **Wikis**.

KEYWORDS

question answering, Wikidata, knowledge graphs, user interfaces

ACM Reference Format:
Francisca Suárez and Aidan Hogan. 2023. Templet: A Collaborative System for Knowledge Graph Question Answering over Wikidata. In *Companion Proceedings of the ACM Web Conference 2023 (WWW '23 Companion), April 30-May 4, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3543873.3587335

1 INTRODUCTION

Open knowledge graphs such as Wikidata [15] make a wealth of structured data available to the public. Users can search Wikidata for an entity of interest to see (or even edit) the data about it. The knowledge graph structure of Wikidata further enables structured queries that can draw answers not just from a single entity, but rather from the knowledge graph as a whole. The Wikidata Query Service [10] then supports evaluating structured queries in the SPARQL language [8]. The service includes motivating examples of queries, ranging from a query that lists *cats*, to a query that lists *genes that predict a positive prognosis in colorectal cancer*.

However, most Web users are not familiar with structured query languages like SPARQL, and will thus struggle to define original queries. Research on *knowledge graph question answering* (*KGQA*) systems aims to assist such users. KGQA systems receive questions in natural language and try to directly answer them over a knowledge graph. Recent approaches exploit advances in Deep Learning [4], with one such line of research using *Neural Machine Translation* (*NMT*) to translate the user's question directly into a structured query over the knowledge graph (see, e.g., [6, 12, 16, 17]).

However, Deep Learning models typically require an extensive training corpus, which in the NMT case, implies the need for an extensive parallel corpus of question–query pairs. Unlike natural languages – where (human) translators have been manually translating between natural languages for millennia – a parallel corpus of natural language questions and their corresponding structured queries does not "naturally occur". Addressing this gap, authors have generated large-scale corpora via numerous approaches, including the employment of human annotators to generate simple question–query pairs [2], to paraphrase questions [7], etc.; and the use of query templates to generate a large number of query instances by replacing placeholders in the template with specific entities [7, 9, 12]. The former approach has been limited to simple questions: generating more complex questions requires more expert users and more manual effort. The latter approach leads to corpora with a large number of potentially complex instances, but relatively uniform ones; when a user subsequently poses a question that does not correspond to one of the templates used to generate training examples, the QA system will tend to perform poorly [6].

In this context, we have created *QAWiki*[1], a collaboratively-edited knowledge base of question–query pairs using Wikibase. QAWiki currently provides questions in both English and Spanish, SPARQL queries for Wikidata, annotations of mentions, paraphrased versions of questions, as well as specific relations between questions (generalises, specialises, disambiguates, etc.). QAWiki currently hosts 361 diverse question–query pairs, with annotated entity mentions. Though this may seem few, each such pair is handcrafted and can be generalised into a template in order to generate manifold instances; for comparison, DBNQA [9] contains 894 thousand question–query pairs generated from *215* templates. We aim to continue expanding QAWiki, with the help of the community.

To attract attention to QAWiki, and to encourage the community to contribute to it, we foresee the need for a complementary QA application layered on top of QAWiki. A key design philosophy for this application was to provide the community with full control and

WWW '23 Companion, April 30-May 4, 2023, Austin, TX, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9419-2/23/04.
https://doi.org/10.1145/3543873.3587335

[1]http://qawiki.org

transparency over the questions and their corresponding queries (and thus the answers they return), allowing users to collaborate on curating question–query pairs on QAWiki. We see this design goal as playing into the strengths of using knowledge graphs for QA versus using, e.g., large language models that are difficult to curate and explain (though both approaches are indeed complementary).

In this context, we have designed and implemented *Templet*: a template-based QA system built over QAWiki. Question–query pairs are taken from QAWiki, and using the mentions that the latter provides, entities are replaced with placeholders. The user finds a question template via autocomplete, and upon selecting one, is presented with an interface to fill the template with entities of interest via autocomplete. The system supports English and Spanish. A user who does not find, or is unhappy with, a particular question or its answers can edit QAWiki or Wikidata accordingly.

2 RELATED WORKS

For reasons of space, we herein focus on template-based KGQA for complex questions. For a broader discussion of QA, please see the tutorial by Unger et al. [14] and the survey by Diefenbach et al. [5].

In the context of QA over RDF knowledge graphs, Unger et al. [13] propose to parse a question into a SPARQL template, whose entities and predicates are identified and filled in a subsequent step using statistical techniques. Park et al. [11] use similarity measures to try to map questions to a predefined list of abstract SPARQL query templates (e.g., an `ASK` frame for boolean questions, a `COUNT` aggregation for counting questions, etc.). Athreya et al. [1] also classify questions into pre-defined templates, but rather using recursive neural networks. A number of other QA systems try to parse the template from the question (see [5] for discussion), which is a challenging task, particularly with sparse training data.

Templet is more straightforward and transparent than such approaches. It includes no statistical nor learning techniques. Users choose a template and then choose what entities they wish to fill into the template's placeholders via autocomplete. This approach depends on having a suitable template available in the QAWiki repository used; however, state-of-the-art QA techniques for knowledge graphs based, e.g., on NMT likewise often fail if the user poses a complex question not corresponding to a template seen in the training set [6]. If no corresponding template is found in the Templet system, users can choose to add it to QAWiki. Furthermore, to the best of our knowledge, Templet is the only QA-style system for Wikidata currently available online: https://templet.dcc.uchile.cl.

3 GENERATING TEMPLATES FROM QAWIKI

In Figure 1, we present an example of a simple question, and a more complex question from QAWiki.[2] We include a subset of the meta-data provided by QAWiki that are relevant for the purposes of Templet, where for each question we have the English expression and mentions (Q_{en}), the Spanish expression and mentions (Q_{es}), and the query in SPARQL (Q_{sp}). Mentions are annotated with one or more Wikidata entities (`Q*`) and properties (`P*`) independently of the query; for example, the first query does not use the entity `Q8142`

Q_{en} $\text{“}\textit{What } \overbrace{\textit{currency}}^{\texttt{Q8142,P38}} \textit{ does } \overbrace{\textit{Angola}}^{\texttt{Q916}} \textit{ use?}\text{”}$

Q_{es} $\text{“}\textit{¿Qué } \overbrace{\textit{divisa}}^{\texttt{Q8142,P38}} \textit{ se usa en } \overbrace{\textit{Angola}}^{\texttt{Q916}}\textit{?}\text{”}$

Q_{sp}
```
SELECT ?obj WHERE { wd:Q916 wdt:P38 ?obj }
```

Q_{en} $\text{“}\textit{Which } \overbrace{\textit{movie}}^{\texttt{Q11424}} \overbrace{\textit{directed by}}^{\texttt{P57}} \overbrace{\textit{Kubrick}}^{\texttt{Q2001}} \overbrace{\textit{won}}^{\texttt{P166}} \textit{ the most } \overbrace{\textit{Oscars}}^{\texttt{Q19020}}\textit{?}\text{”}$

Q_{es} $\text{“}\textit{¿Qué } \overbrace{\textit{película}}^{\texttt{Q11424}} \overbrace{\textit{dirigida por}}^{\texttt{P57}} \overbrace{\textit{Kubrick}}^{\texttt{Q2001}} \overbrace{\textit{ganó}}^{\texttt{P166}} \textit{ más } \overbrace{\textit{premios Óscar}}^{\texttt{Q19020}}\textit{?}\text{”}$

Q_{sp}
```
SELECT ?sbj WHERE {
  { SELECT (MAX(?count) AS ?max) WHERE {
      SELECT (COUNT(?obj) AS ?count) WHERE {
        ?sbj wdt:P31/wdt:P279* wd:Q11424 .
        ?sbj wdt:P57 wd:Q2001 . ?sbj wdt:P166 ?obj .
        ?obj wdt:P31*/wdt:P279* wd:Q19020 . }
      GROUP BY ?sbj } }
  { SELECT ?sbj (COUNT(?obj) AS ?max) WHERE {
      ?sbj wdt:P31/wdt:P279* wd:Q11424 .
      ?sbj wdt:P57 wd:Q2001 . ?sbj wdt:P166 ?obj .
      ?obj wdt:P31*/wdt:P279* wd:Q19020 . }
    GROUP BY ?sbj } }
```

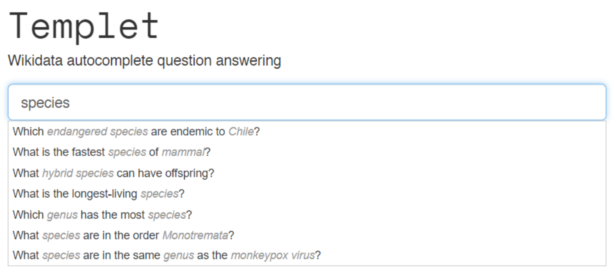
Figure 1: Meta-data for two QAWiki questions

for "*currency*", but rather uses the property `P38` instead. As QAWiki uses the Wikibase software, these meta-data can be accessed as RDF, and a SPARQL query service is provided over QAWiki.

Templet constructs question and query templates from QAWiki's specific instances. For example, from "*What currency does Angola use?*", we construct a template of the form "*What currency does* ___ *use?*", further replacing `wd:Q916` in the query with a placeholder. We do so by identifying entity ids of the form `wd:Q*` that appear in the subject or object position of a triple pattern in the query, replacing it with a placeholder that is associated with its corresponding mention in the text. For example, for the simple QAWiki question shown, we would identify the entity `wd:Q916`, create a placeholder `$1` for it, and associate the placeholder `$1` with the mention "*Angola*". The same process applies for both English and Spanish using the corresponding mentions. In the case of the second question, three placeholders would be generated, creating the template "*Which* ___ *directed by* ___ *won the most* ___*?*", which could be instantiated, for example, with "*Which miniseries directed by Cholodenko won the most Emmys?*" The specific entities to enter into each placeholder will be chosen by the user in the front-end using autocompletion.

We do not use placeholders for properties as this often (though not always) creates misleading text-based templates. Taking the question "*What currency does Angola use?*", its template with entity and property placeholders is "*What* ___ *does* ___ *use?*", which works for "*What publisher does The New England Journal of Medicine use?*", but not for "*What piece does smothered checkmate use?*" (Wikidata does not have a "*piece*" property but rather states that *smothered mate* (`Q903961`) ***uses*** (`P2283`) *knight* (`Q136`)). Furthermore, sometimes property mentions in QAWiki are not continuous. For example, in the question "*Where was tellurium discovered?*", the Wikidata property *location of discovery* (`P189`) is linked from the mention "*Where * discovered*" in QAWiki in order to distinguish it from the

[2] We abbreviate the expression of the second query for presentation purposes. The query could be greatly simplified by using a combination of `ORDER BY` and `LIMIT 1` if we did not care about ties; in this case, however, the movies Spartus and Barry Lyndon are tied as answers, where both are returned by the query shown.



Templet

Wikidata autocomplete question answering

species

Which *endangered species* are endemic to *Chile*?

What is the fastest *species* of *mammal*?

What *hybrid species* can have offspring?

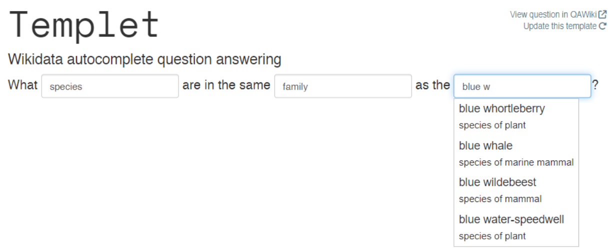
What is the longest-living *species*?

Which *genus* has the most *species*?

What *species* are in the order *Monotremata*?

What *species* are in the same *genus* as the *monkeypox virus*?

Figure 2: Screenshots of first autocomplete interaction



Templet

Wikidata autocomplete question answering

What species are in the same family as the blue w ?

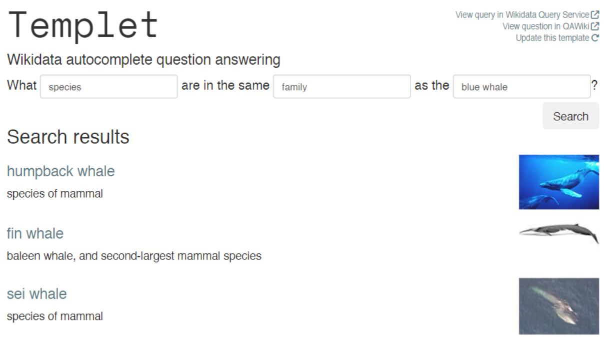
blue whortleberry
species of plant

blue whale
species of marine mammal

blue wildebeest
species of mammal

blue water-speedwell
species of plant

Figure 3: Screenshot of second autocomplete interaction



Templet

Wikidata autocomplete question answering

What species are in the same family as the blue whale ?

Search results

humpback whale
species of mammal

fin whale
baleen whale, and second-largest mammal species

sei whale
species of mammal

Figure 4: Screenshot showing a subset of results

related question "*When was tellurium discovered?*", which invokes the property *time of discovery or invention* (P575); the use of * indicates a wildcard of one or more characters in a discontinuous mention. Other complications relate to the use of inverse, hyponym or hypernym properties in the query compared with the question, as well as the use of implicit properties, where the mention "*horror film*" implies the property *genre* (P136). Generalising properties is thus prone to more problematic cases than generalising entities.

4 TEMPLET FRONT-END

The front-end of Templet is designed around two main interactions.

In the first interaction, the user selects a question template via autocomplete; for example, if they type "what cu", they will receive suggested templates including "*What currency does Angola use?*". We include the original entities in their placeholders as examples; otherwise some templates become very difficult to understand, such as "*Which* ___ ___ *have* ___ *parent* ___?" (for "*Which Chilean companies have U.S. parent companies?*"). Autocompletion normalises for capitalisation, accents, punctuation (important for omitting "¿" in Spanish), often-interchangeable interrogative pronouns (e.g., *what* ↔ *which*), and can also match words in the middle of a question template, as seen in Figure 2. To implement custom autocompletion in an efficient way, it is necessary to precompute and cache templates from QAWiki, which are refreshed each hour; the front-end also provides links to refresh an individual template from QAWiki in case changes are made. If a relevant template is not found, a prompt redirects the user to QAWiki so that they can optionally add their question for future users to benefit from.

Once the user selects a question template, the second interaction enables the user to choose entities via autocompletion for each placeholder. Assume, for example, that the user selects the bottom-most suggestion of Figure 2: "*What species are in the same genus as the monkeypox virus?*". As per Figure 3, they can use autocomplete (called via the Wikidata API) to replace the grey entities with ones of their choosing. In the top-right corner of Figure 3, one can also see options to view or refresh the QAWiki question.

Assuming the user fills the template from Figure 3 with "*What species are in the same family as the blue whale?*" and clicks Search, the entity replacements will be filled into the SPARQL query template and evaluated on the Wikidata Query Service. The user will then be presented results as shown in Figure 4. For each result, a label with a link to Wikidata is presented, along with a description and image, where available. Results can also be a simple value (e.g., a count or boolean); or a table of results. In the top-right corner, the user can also view the query in the Wikidata Query Service.

Templet also supports *contingent questions* in QAWiki that check the assumptions of a question. For example, "*When did Jimmy Carter die?*" is contingent on a positive answer to "*Is Jimmy Carter dead?*". Where available, these contingent questions are evaluated by Templet and flagged to the user if an unexpected answer is derived.

5 EVALUATION

We conducted a usability survey asking users to (*a*) solve in Templet "*When did Amy Winehouse die?*", "*Who played Hermione Granger in Harry Potter and the Philosopher's Stone*" and "*Which collections outside of Chile exhibit moai?*"; (*b*) continue to use Templet for questions of their own choice; and thereafter (*c*) complete a Systems Usability Scale (SUS) questionnaire [3] responding to usability-related claims on a Likert scale from 1 (disagree) to 5 (agree). A Spanish version of the survey was shared in a forum of students and staff at the Department of Computer Science, University of Chile, where it received 31 responses. An English version was shared on the public Wikidata mailing list, receiving 6 responses. The SUS results for both surveys are listed in Table 1, where a SUS score of above 68 is considered "above average". In this setting, the SUS score was 76.72 for the Spanish survey ($n = 31$), 59.58 for the English survey ($n = 6$), and 74.01 combining all respondents ($n = 37$).

The end of the survey included a comments section for qualitative remarks. A key point of feedback was that when searching for "*When did Amy Winehouse die?*" in the first interaction (see Figure 2), the system would rather suggest "*When did Stan Lee die?*", with the idea that in the second interaction, the entity "*Stan Lee*"

Table 1: SUS results (*m*: mean; *s*: standard deviation)

Claim to evaluate	*ES* (31)		*EN* (6)		*All* (37)	
	m	*s*	*m*	*s*	*m*	*s*
I think that I would like to use this system frequently.	3.53	0.98	2.33	0.52	3.34	1.02
I found the system unnecessarily complex	1.91	1.03	2.67	1.03	2.03	1.05
I thought the system was easy to use.	4.03	0.97	2.67	0.82	3.82	1.06
I think that I would need the support of a technical person to be able to use this system.	1.25	0.62	1.33	0.52	1.26	0.60
I found the various functions in this system were well integrated.	4.03	1.12	3.00	0.63	3.87	1.12
I thought there was too much inconsistency in this system.	1.94	1.37	2.50	1.05	2.03	1.33
I would imagine that most people would learn to use this system very quickly.	3.75	1.19	3.33	0.82	3.68	1.14
I found the system very cumbersome to use.	1.78	1.10	2.67	0.82	1.92	1.10
I felt very confident using the system.	3.94	1.11	3.00	1.26	3.79	1.17
I needed to learn a lot of things before I could get going with this system.	1.72	0.92	1.33	0.52	1.66	0.88
SUS Points	76.72	16.26	59.58	9.67	74.01	16.56

could be replaced by "*Amy Winehouse*". Some users struggled with this, particularly for the first question, and suggested that such entity labels could be auto-filled in some way. However, this would not be trivial in the case that the question continues beyond the first placeholder: it would be difficult to detect where the auto-filled entity ends, and the rest of the question continues. We rather opted to add a tooltip to hint that such placeholder labels can be replaced.

6 CONCLUSIONS

We present Templet: a question answering (QA) system for Wikidata which is, to the best of our knowledge, the only QA-style system currently available online for Wikidata, and the only such system that allows users to edit the collection of questions supported. Aside from answering questions, we believe that Templet can help to encourage contributions to QAWiki, which in turn can be used to train and evaluate Deep Learning models for (KG)QA. Templet could also act as a repository of example queries for the Wikidata Query Service that can be further personalised with custom entities.

In future work, we will explore ways to auto-fill entities directly in the first interaction. We also aim to explore selectively using placeholders for properties. Another idea would be to only suggest (or prioritise suggestions of) entities that can generate answers in the context of the template and the other entities selected. It may also be of interest to support additional result types, such as graph or map visualisations, etc. Regarding precision/recall, while this depends on the quality of the template defined, it would be interesting to see in how many cases the final query corresponds to the user's intent. Another idea would be to combine Templet with a large language model for questions not answerable as a query on Wikidata, such as procedural questions (e.g., "*How to tie a tie?*"), explanatory questions (e.g., "*Why is the sky blue?*"), etc.

Supplementary material. An online demo of Templet is available at https://templet.dcc.uchile.cl/. Source-code for Templet can be found at https://github.com/franpss/qa-autocomplete.

ACKNOWLEDGMENTS

This work was supported by Fondecyt Project No. 1221926 and ANID – Millennium Science Initiative Program – Code ICN17_002. We would like to thank Daniel Diomedi, Lucas Carrasco, Santiago Mass, Martín Paredes, and everyone else who contributed question–query pairs that form part of QAWiki and are used by Templet.

REFERENCES

[1] Ram G. Athreya, Srividya Kona Bansal, Axel-Cyrille Ngonga Ngomo, and Ricardo Usbeck. 2021. Template-based Question Answering using Recursive Neural Networks. In *15th IEEE International Conference on Semantic Computing, ICSC 2021, Laguna Hills, CA, USA, January 27-29, 2021.* IEEE, 195–198. https://doi.org/10.1109/ICSC50631.2021.00041
[2] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale Simple Question Answering with Memory Networks. *CoRR* abs/1506.02075 (2015). arXiv:1506.02075
[3] John Brooke. 1996. SUS – A quick and dirty usability scale. *Usability Evaluation in Industry* 189, 194 (1996), 4–7.
[4] Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, Priyansh Trivedi, Jens Lehmann, and Asja Fischer. 2019. Introduction to Neural Network based Approaches for Question Answering over Knowledge Graphs. *CoRR* abs/1907.09361 (2019). arXiv:1907.09361
[5] Dennis Diefenbach, Vanessa López, Kamal Deep Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowl. Inf. Syst.* 55, 3 (2018), 529–569. https://doi.org/10.1007/s10115-017-1100-y
[6] Daniel Diomedi and Aidan Hogan. 2022. Entity Linking and Filling for Question Answering over Knowledge Graphs. In *Proceedings of the 7th Natural Language Interfaces for the Web of Data (NLIWoD) co-located with the 19th European Semantic Web Conference (ESWC 2022), Hersonissos, Greece, May 29th, 2022 (CEUR Workshop Proceedings, Vol. 3196).* CEUR-WS.org, 9–24.
[7] Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. LC-QuAD 2.0: A Large Dataset for Complex Question Answering over Wikidata and DBpedia. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, Oct. 26-30, 2019, Proc., Part II (LNCS, Vol. 11779).* Springer, 69–78.
[8] Steve Harris, Andy Seaborne, and Eric Prud'hommeaux. 2013. SPARQL 1.1 Query Language. W3C Recommendation. http://www.w3.org/TR/sparql11-query/.
[9] Ann-Kathrin Hartmann, Tommaso Soru, and Edgard Marx. 2018. Generating a Large Dataset for Neural Question Answering over the DBpedia Knowledge Base. In *Workshop on Linked Data Management.*
[10] Stanislav Malyshev, Markus Krötzsch, Larry González, Julius Gonsior, and Adrian Bielefeldt. 2018. Getting the Most Out of Wikidata: Semantic Technology Usage in Wikipedia's Knowledge Graph. In *The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, Oct. 8-12, 2018, Proc., Part II (LNCS, Vol. 11137).* Springer, 376–394.
[11] Seonyeong Park, Soonchoul Kwon, Byungsoo Kim, and Gary Geunbae Lee. 2015. ISOFT at QALD-5: Hybrid Question Answering System over Linked Data and Text Data. In *Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015 (CEUR Workshop Proceedings, Vol. 1391).* CEUR-WS.org.
[12] Tommaso Soru, Edgard Marx, Diego Moussallem, Gustavo Publio, Andre Valdestilhas, Diego Esteves, and Ciro Baron Neto. 2017. SPARQL as a Foreign Language. In *Proc. of the Posters and Demos Track of the 13th International Conference on Semantic Systems - SEMANTiCS2017, Amsterdam, The Netherlands, Sept. 11-14, 2017 (CEUR, Vol. 2044).* CEUR-WS.org.
[13] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. 2012. Template-based question answering over RDF data. In *Proc. of the 21st World Wide Web Conference 2012, WWW 2012, Lyon, France, Apr. 16-20, 2012.* ACM, 639–648.
[14] Christina Unger, André Freitas, and Philipp Cimiano. 2014. An Introduction to Question Answering over Linked Data. In *Reasoning Web. Reasoning on the Web in the Big Data Era - 10th International Summer School 2014, Athens, Greece, Sept. 8-13, 2014. Proc. (LNCS, Vol. 8714).* Springer, 100–140.
[15] Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.
[16] Shujun Wang, Jie Jiao, Yuhan Li, Xiaowang Zhang, and Zhiyong Feng. 2020. Answering Questions over RDF by Neural Machine Translating. In *Proc. of the ISWC 2020 Demos and Industry Tracks: From Novel Ideas to Industrial Practice, Globally online, Nov. 1-6, 2020 (UTC) (CEUR, Vol. 2721).* CEUR-WS.org, 189–194.
[17] Xiaoyu Yin, Dagmar Gromann, and Sebastian Rudolph. 2021. Neural machine translating from natural language to SPARQL. *Future Gener. Comput. Syst.* 117 (2021), 510–519.

Received 3 February 2023