ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230411

Language Models as Controlled Natural Language Semantic Parsers for Knowledge Graph Question Answering

Jens Lehmann [a;*], Preetam Gattogi[b,c], Dhananjay Bhandiwad[b], Sébastien Ferré[d] and Sahar Vahdati[b]

[a]Amazon
[b]Institute for Applied Informatics (InfAI)
[c]Dresden University of Technology
[d]University Rennes
ORCiD ID: Jens Lehmann https://orcid.org/0000-0001-9108-4278,
Preetam Gattogi https://orcid.org/0009-0001-4604-2009,
Dhananjay Bhandiwad https://orcid.org/0009-0005-6955-5439,
Sébastien Ferré https://orcid.org/0000-0002-6302-2333, Sahar Vahdati https://orcid.org/0000-0002-7171-169X

Abstract. We propose the use of controlled natural language as a target for knowledge graph question answering (KGQA) semantic parsing via language models as opposed to using formal query languages directly. Controlled natural languages are close to (human) natural languages, but can be unambiguously translated into a formal language such as SPARQL. Our research hypothesis is that the pre-training of large language models (LLMs) on vast amounts of textual data leads to the ability to parse into controlled natural language for KGQA with limited training data requirements. We devise an LLM-specific approach for semantic parsing to study this hypothesis. To conduct our study, we created a dataset that allows the comparison of one formal and two different controlled natural languages. Our analysis shows that training data requirements are indeed substantially reduced when using controlled natural languages, which is relevant since collecting and maintaining high-quality KGQA semantic parsing training data is very expensive and time-consuming.

1 Introduction

Over the past decades, large amounts of structured knowledge in the form of knowledge graphs have been published [11, 29, 32]. To access the information in such knowledge graphs, query languages like SPARQL[1] are used. The use of formal queries to access knowledge graph poses difficulties for non-expert users as they require the user to understand the syntax of the formal query language, as well as the underlying structure of entities and their relationships. KG question answering (KGQA) systems, therefore, aim to provide the users with an interface to ask questions in natural language, using their own terminology, to which they receive a concise answer generated by querying the KG. Recently, KGQA systems have also been used as an auxiliary system (commonly called a "tool" or "plugin") for LLMs in chatbots, since they allow LLMs to ground their responses in factual knowledge stored in a knowledge graph [16, 34] to reduce LLM hallucinations.

KGQA systems typically use semantic parsing approaches in which natural language questions are transformed into formal language. Such approaches are often trained on large datasets of (natural language, formal query language) pairs. Creating such datasets can be very expensive since handwriting formal queries for natural language input queries is a time-consuming task. At the same time, machine learning techniques for semantic parsing are typically not sufficiently accurate when trained on smaller datasets. To enable the creation of large datasets, various semi-automated approaches have been used [5, 26, 31]. Although those datasets were helpful to advance the state-of-the-art, they are still largely template-based, do not generalise well outside of the scope of those templates and often do not reflect the traffic seen in QA systems in production. Therefore, we argue that it is important to investigate KGQA semantic parsing systems which require smaller amounts of training data.

As a step towards achieving this, we propose using a controlled natural language (CNL) as the target formal language. CNL expressions are close to (human) natural language while they can still be unambiguously translated into a formal language. Our research hypothesis is that using controlled natural language is more suitable than a formal query language for knowledge graphs, since CNLs are a) closer to the input query in natural language and b) closer to the high amounts of textual pre-training data of language models, that can be employed for semantic parsing. To the best of our knowledge, this research hypothesis has so far not been investigated for knowledge graph query languages. There are studies on controlled natural languages for simpler custom query languages, for example [20]. However, these studies do not yet cover the more complex compositional patterns of graph query languages, aggregation functions, and common hyperrelational graph structures. We believe the research hypothesis is interesting because while controlled natural language expressions are close to the large amount of human-authored text that LLMs have been trained on, they typically do not occur frequently in LLM pre-training data directly. In contrast to this, formal query languages like SPARQL or SQL are frequent in pre-training data and in small-scale scenarios work to some extent already in zero- and few-

* Corresponding Author. Email: jlehmnn@amazon.com.
[1] https://www.w3.org/TR/rdf-sparql-query/

shot settings. From this perspective, we were curiously interested which of those factors would have a bigger influence on semantic parsing performance.

To verify our research hypothesis, we conducted a study using the Mintaka dataset [25] based on Wikidata. Unlike most KGQA datasets, Mintaka contains very natural questions, as they were collected manually in a high-quality data collection process. However, due to the significant effort that would be required to collect (natural language, logical form) pairs, Mintaka only contains (natural language, response) pairs, i.e. a so called weak supervision setting. We prefer to investigate the strong supervision setting because of its widespread use in semantic parsing and its simplicity, which only requires LLM fine-tuning. To be able to investigate our research hypothesis, we labelled a subset of 550 questions of Mintaka with their corresponding logical forms. We provide three variants of those logical forms: SPARQL [10], Sparklis [7] and SQUALL [6]. The latter two are controlled natural languages. We then used several language models, specifically BLOOM [24], GPT Neo[2], GPT-2 [18], GPT-3 [3], T5 [19] and Llama 2 [30] on this dataset. We employ the state-of-the-art ReFinED [1] entity linker to enrich the LLM prompt with entity information. The specific prompting and fine-tuning techniques that we used are described in Section 4. We generally focused on comparing results across target languages rather than optimising the performance of the approach itself. Overall, we make the following contributions:

- Proposing controlled natural languages as target for KGQA semantic parsing, opposed to using formal query languages directly.
- Provision of a dataset containing 550 (NL question, formal KG query) pairs to the community.[3] While larger (semi-)automatically created datasets exist, this is one of the largest publicly available strong supervision datasets using a manually collected set of questions (similar to, e.g. [17]) and the largest for controlled natural language KGQA.
- Development of an LLM-based semantic parsing approach that builds on an external entity linker and leverages the capabilities of LLMs to hallucinate for relation linking.
- An evaluation of semantic parsing accuracy for three target languages on six different language models using different metrics.
- An analysis of the relationship between fine-tuning scale and semantic parsing accuracy across the three target languages.
- A qualitative result analysis on our test set of 150 queries.

2 Related Work

We first discuss related work in the area of knowledge graph question answering and then continue with controlled natural languages for semantic parsing.

2.1 KGQA Semantic Parsing

A range of different approaches have been proposed for semantic parsing, such as Combinatory Categorial Grammars (CCG), rule-based systems or neural-network based methods. [9] provides a general overview over such semantic parsing techniques. Since we focus more specifically on semantic parsing in the context of knowledge graphs, we refer to [4] as an introduction and survey for neural network-based KGQA semantic parsing.

Modern KGQA semantic parsing methods are often based on sequence to sequence architectures, e.g. [21] where an encoder translates the natural language utterance into an intermediate state and a decoder then translates this state into a logical form. Pointer generator networks allow to copy some part of the input, e.g. a data value, into the output. Those approaches have shown to be accurate when trained on large amounts of high quality training data. However, as described in the introduction obtaining training data is very challenging and updating the training set to incorporate new features or cover more domains requires substantial effort. For this reason, there was an increasing interest in training semantic parsers with less data. For example, [22] explores an unsupervised approach for semantic parsing. While achieving promising results, such approaches do not achieve the performance levels required for production level systems and cannot easily be tuned to do so.

Large language models such as GPT-3 [3], BLOOM [24] or Llama 2 [30] also support semantic parsing with zero or few examples and are a promising avenue towards reducing the need for high amounts of training data, in particular after several improvements have been made to reduce the inference costs of such models. However, LLMs do not support KG-specific parsing directly without access to entities in the underlying knowledge graph. For this reason, they need to be augmented with a method to retrieve entities. We are not aware of any publication that has used an LLM-based generative approach for knowledge graph question answering at the point of writing but point to some related work with controlled natural language as a target in the subsection below.

2.2 Controlled Natural Languages for Semantic Parsing

Controlled natural languages (CNLs) have been used as target for semantic parsing in applications outside of knowledge graph question answering. For example, [20] has already investigated the idea of using a special-purpose CNL to reduce training data requirements and showed improvements on the Overnight dataset [33] in several smaller domains. On the same (and other) datasets, [28] used a constraint decoding procedure for semantic parsing that employs the GPT-3 and Codex models. They compared a controlled natural language and a standard logical form language and could show accuracy benefits of using a controlled natural language. Although these previous studies are an indicator and motivation for our investigation, they were performed on datasets with a smaller scale and not on knowledge graphs. For example, an Overnight domain has at most 45 relations compared to more than 10,000 relations (and more than 100 million entities) in the Wikidata knowledge graph [32] that we use. Similarly, [27] investigated the use of language models (BART, GPT-2, GPT-3) for constrained decoding of synchronous context-free grammars (SCFG) following an idea similar to ours by first converting an NL question into a canonical human-like utterance which is then translated into the target logical form. These are again evaluated on Overnight and related datasets. [14] introduces a new language Graph-IR with the aim of unifying the semantic parsing of graph query languages with one intermediate representation. The principles of their intermediate representation language are similar to CNL but with less emphasis on naturalness compared to the SQUALL language which we are investigating. Moreover, they use a BART-based encoder-decoder model, whereas we focus on generative approaches using LLMs.

[2] https://github.com/EleutherAI/gpt-neo
[3] https://github.com/NIMI-research/CNL_KGQA

3 Preliminaries

3.1 Semantic Parsing

Semantic parsing is the task of translating a natural language (NL) utterance into a machine-interpretable representation q (query) of its meaning in a given formal language. The generated logical form q is correct if it captures the meaning of the input NL utterance. Given that it is difficult to formalize the concept of meaning, there are several metrics for measuring the correctness of semantic parsing:

One set of metrics relevant for semantic parsing are *execution metrics*, which measure whether the execution of the logical form, e.g. executing a query over a knowledge graph, yields the desired result. It should be noted that query execution is not sufficient for correctness because multiple queries can return the same result even if they do not correctly capture the meaning of the input question. We call such logical forms *spurious*. For example, translating "*What is two plus two*" to $2*2$ yields the correct result but is not the correct query.

Another set of metrics are text similarity metrics such as BLEU [15] or METEOR [2]. Those are primarily used as proxy metrics to assess how close the semantic parse is to the gold standard query, but are less significant than execution metrics for KGQA.

3.2 Knowledge Graphs

In this paper, we use hyperrelational knowledge graphs as a base. Hyperrelations are common, e.g., they are required to model the relationship properties in graph databases and the DBpedia [11] or Wikidata [32] knowledge graphs use variants of this data model. Using hyperrelations via so called qualifiers in Wikidata is indeed required for several questions in the Mintaka dataset which we evaluate on. Let $\mathcal{E} = \{e_1 \dots e_{n_e}\}$ be a set of entities, $\mathcal{L}$ be the set of all literal values, and $\mathcal{P} = \{p_1 \dots p_{n_r}\}$ be a set of properties (also called relations). A *statement* $t = (s, p, o, Q)$ comprises a subject $s \in \mathcal{E}$, a property $p \in \mathcal{P}$, an object $o \in (\mathcal{E} \cup \mathcal{L})$, and a set of qualifiers $Q \subseteq \mathcal{P} \times (\mathcal{E} \cup \mathcal{L})$. The property p relates the subject s to the object o, and the set of qualifiers Q further describes the relationship. An example of statement in Wikidata is *(The Twilight Saga: Breaking Dawn – Part 1, part of the series, The Twilight Saga, { (series ordinal, 4), (follows, The Twilight Saga: Eclipse)})*, which describes the film "Breaking Dawn - Part 1" as the fourth in the Twilight Saga, following the film "Eclipse". A KG $\mathcal{K}$ is a set of such statements.

3.3 Controlled Natural Languages for SPARQL

Generally, Controlled Natural Languages aim at bridging the high-level and natural syntax of natural languages, and the lack of ambiguity of formal languages such as SPARQL [10]. CNLs allow for abstracting from the low-level aspects of formal languages (such as bindings and relational algebra). Sparklis [7] and SQUALL [6] are examples of SPARQL-oriented CNLs for querying and updating knowledge graphs. SQUALL has a very high coverage of SPARQL queries and updates. It therefore combines the expressivity of SPARQL, and at the same time the readability of natural languages. Its main limit is that the content words (e.g., proper nouns, verbs) must be non-ambiguous references to entities and relations. SQUALL is based on Montague grammars [13] that combine context-free grammars, first order logic, and λ-calculus. Sparklis acts as an interactive SPARQL endpoint explorer that completely hides SPARQL behind a CNL. Sparklis' CNL is slightly less expressive and less natural than SQUALL, and it only covers SELECT queries.

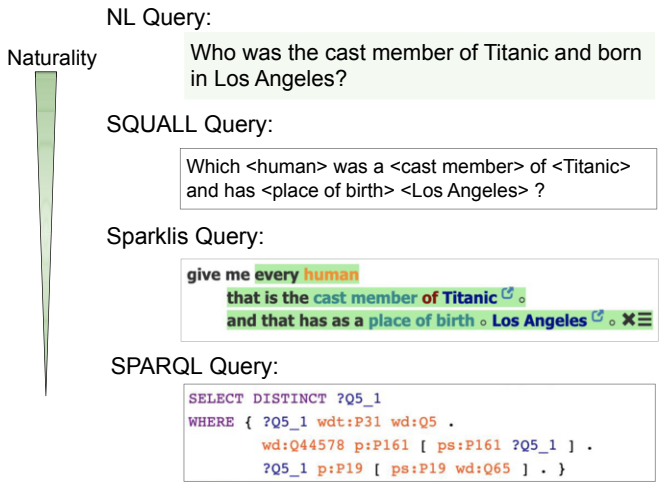

Figure 1. An example natural language query from the Mintaka dataset, and its equivalent queries written in the SQUALL and Sparklis controlled languages as well as SPARQL query language.

In counterpart, queries can be built incrementally through user interaction: at each step, Sparklis suggests some query refinements, among which the user selects the refinement that fits her information needs. This allows the user to discover the data model on-the-fly, and this prevents ill-formed queries. The main limitation is that, in case of a complex data model (like in Wikidata), it may be difficult for the user to find a valid sequence of refinements. Figure 1 shows an example natural language query that is taken from the Mintaka dataset. We showcase the corresponding SPARQL, Sparklis and SQUALL versions of this question.

4 Semantic Parsing Approach

4.1 Overview

Our main goal is to compare different target languages for semantic parsing using LLMs. For this reason, we devised an approach that follows the same steps for all target languages. Upon receiving a question as input, we first perform an entity linking step. We then instruct the LLM to convert the input query into the target language.

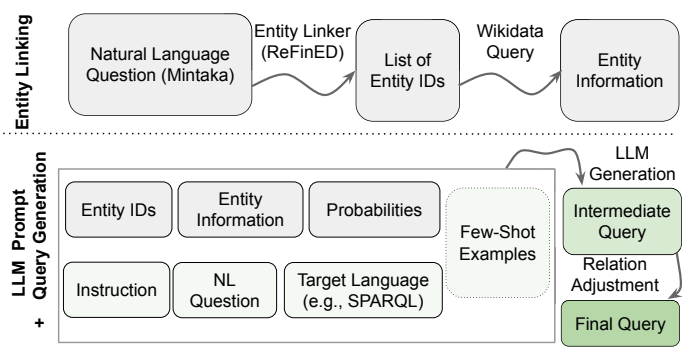

Figure 2. Runtime workflow: First, entity linking is performed. The linked entities with additional information are fed into the prompt construction. The LLM generates an intermediate query after which relations can be adjusted.

4.2 Entity Linking

Entity linking is the NLP task that assigns a unique identifier - in our case a node in the knowledge graph - to entities mentioned in an input utterance. Our approach is not specific to a particular entity linker. In our experiments, we decided to use the ReFinED [1] entity linker, since it achieves state-of-the-art results on Wikidata and is runtime efficient. For a given input utterance, we retrieve all entities above a threshold probability value returned by ReFinED. This threshold is a hyperparameter of our semantic parsing approach. For each returned entity, we obtain its label and description. For each entity the quadruple (entity ID, probability, label, description) is passed as input to the LLM prompting step. The additional information beyond the entity ID can be used by the LLM for the semantic parsing step. For example, the probability may lead to the LLM preferring highly probable entities over less probable entities. The label and description give the LLM additional context. It should be noted that the entity label alone is not unique, e.g. the label "The Hunger Games" exists multiples times in Wikidata (for the books, the movie series and the first movie in the series itself).

Relation linking is the NLP task that assigns a unique identifier to relations relevant to an input utterance. We do not employ a separate relation linker, but let the LLM generate the relation as part of the query generation. We do this for several reasons:

- First, relation linking is usually less accurate than entity linking, since relations are frequently implicit (i.e. not part of the question) or have a variety of surface forms. Given the difficulty of this task, the LLM itself may be the most suitable component for this task.
- Second, knowledge graphs usually contain much fewer relations than entities. For example, as of May 2023, Wikidata contains approximately 10 thousand relations but more than 100 million entities[4]. So while the training data may contain at least a relevant portion of frequent relations, it will only contain a minuscule fraction of existing entities.
- Third, we can use the hallucination of LLMs as a strength by letting it generate likely relations and then adjusting those in a post-processing step if needed (described in more detail later in this section). This is useful since most of the infrequent relations will not be in the training data.

4.3 Prompting

In our approach, the LLM prompt consists of three parts: the instruction, the input question, and the list of entities. As instruction, we use "Given the question and the entities generate a $language query!" where $\$language \in \{\text{SPARQL}, \text{Sparklis}, \text{SQUALL}\}$ is the target language. The question is passed to the LLM as is. For the entities, we pass the quadruples (entity ID, probability, label, description) as described in the previous section. An example of a prompt is given below. We did not perform prompt engineering since optimising performance is not the main scope of the experiments. While this may affect absolute ceiling performance, it is unlikely to substantially affect the performance difference between the different target languages that is our focus here.

```
Given the question and the entities generate a
SQUALL query!

Question: Which actor was the star of Titanic
and born in Los Angeles?

Entities:
[{'ID': 'Q44578', 'Label': 'Titanic',
'Description': '1997 film by James Cameron',
'Probability': 0.4532},
{'ID': 'Q25173', 'Label': 'Titanic',
'Description': 'British transatlantic passenger
liner, launched and foundered in 1912',
'Probability: 0.3498'},
{'ID': 'Q65', 'Label': 'Los Angeles',
'Description': 'largest city in California,
United States of America',
'Probability': 0.9623}]

SQUALL query:
```

4.4 Posthoc Relation Adjustment

Given that the number of fine-tuning training examples which we use is relatively small compared to the number of relations, there is a high chance that a question contains an unseen relation. To overcome this problem, we can make use of LLM hallucinations. Although hallucinations are usually seen as a negative phenomenon in LLMs, we can exploit them as a strength in our case. Even for unseen (or infrequently seen) relations, the LLM causal modeling objective incentivizes it to output tokens that are plausible in the given context if we give it the opportunity to do so. In order to achieve this, we let the LLM output not only the identifier of a relation (e.g. `P162`) but also the label (e.g. `producer`) separated by a colon. For frequently seen relations, the LLM will likely output a correct identifier. If the relation is unseen or infrequently seen, it may hallucinate an incorrect identifier and a plausible relation name, which may, however, not exist in Wikidata. We can then use this relation name to post-process the relation. To do this, we use Cosine similarity to calculate within the 384-dim vector space using the (all-MiniLLM-L6-v23) Sentence transformer from Hugging face [5]. The model embeds each label(both the generated label and the Wikidata label) into 384 dim vectors. After this the generated label vector is compared against all label vectors of Wikidata. Then The hallucinated label is then replaced with the highest scoring relation label from Wikidata.

5 The Mintaka-CNL Dataset

Mintaka [25] is a natural and multilingual question answering dataset containing 20k manually written NL queries in English. We used Mintaka as a base for our studies since it is the only KGQA dataset we are aware of that contains a high number of queries and does not contain questions that are, at least in part, automatically generated (e.g. "Is the WOEID of Tuscaloosa 14605?" in KQA Pro [26] or "1520.0 is the minimum width for which size rail gauge?" in GrailQA [8]). To create our dataset, we sampled questions from Mintaka using the following criteria: a) diversity in question themes, b) diversity in question complexity, c) preference of questions with more than one entity, and d) having at least three examples of the same type of question in each category. After sampling relevant questions from Mintaka, we constructed their equivalent CNL and SPARQL queries. For this, we first turned them into Sparklis queries using its query construction interface[6]. On average, it takes several minutes to create one query using the tool which is still a substantial speedup compared to manually writing SPARQL queries. The majority of this work was done during internal workshops. Three researchers and two student assistants were involved in creating the Sparklis queries.

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

[5] See https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.

[6] http://www.irisa.fr/LIS/ferre/sparklis/

The interface automatically converts queries into SPARQL queries. For SQUALL, the queries are manually written based on the Sparklis query. A semi-automatic conversion is subject to future work. Generally, SQUALL is slightly more expressive than Sparklis, where some features are not directly supported: e.g. selecting the n-th element according to an order or comparisons of relation values (e.g. height, length). SQUALL also supports a direct conversion into SPARQL such that we could verify that both queries return the same output modulo differences in language expressivity.

Table 1. Used models including parameter count.

Model Name	Developed by	Number of Parameters
GPT-2	OpenAI	0.12 B
T5-Large	Google	0.77 B
GPT Neo	EleutherAI	1.3 B
GPT-2 XL	OpenAI	1.5 B
Bloom	BigScience	1.7 B
GPT-3 Curie	OpenAI	6.7 B
Llama2 7B	Meta	7 B
GPT-3 Da Vinci	OpenAI	175B

Mintaka has seven categories of questions including (1) count, (2) intersection, (3) superlative, (4) difference, (5) comparative, (6) generic and (7) ordinal. Our questions are spread over the different categories as follows: 92, 85, 57, 49, 56, 74 and 137. The original Mintaka dataset also has another category named of yes/no questions which we did not consider in this work, since Sparklis does not support SPARQL ASK queries.

In total, we sampled 550 questions and created three different logical forms for each. This dataset is split into 400 train and 150 test questions.

6 Experimental Setup

6.1 *Models and Fine-Tuning*

We used the language models shown in Table 1. The models were selected to cover a range of different sizes suitable for generative NLP tasks. For all models, we used a fine-tuning procedure using the prompt as described in Section 4 and the actual target query as the completion target. For models outside of the GPT-3 family, we updated the tokenizer to include special tokens and re-sized the model's token embeddings. During the fine-tuning phase, the models were trained following parameters of [23] which states 25 epochs with a constant learning rate of 5e-05 and a batch size of 4. The maximum sequence length during training was set at 120. No extensive hyperparameter optimization was performed. The same settings are used for all target languages. For the decoding step, we use a greedy search strategy. Special tokens are excluded from the LLM output before evaluation. The threshold for entity linking was set to 0.1. The fine-tuning was done on 2 Nvidia A100 GPUs on Linux and on OpenAI servers for GPT-3. Code was written in Python 3.10. The fine tuning for Llama 2 [30] was done for 20 epochs, with constant learning rate of 1e-04 and batch size of 1. No extensive hyper- parameter optimization was performed.

6.2 *Metrics*

We employ a combination of translation quality metrics and execution metrics (see Section 3.1): BLEU Cumulative [15], METEOR [2], ROUGE [12] help in evaluating the translation quality, with BLEU cumulative focusing on n-gram precision, METEOR providing a balance between precision, recall and ROUGE-2 that focuses on bi-gram (2-gram) recall. These metrics help to assess the extent to which the generated queries preserve the structural and semantic information of the reference query at granular level. We also use Exact Match accuracy to evaluate the percentage of queries matching the ground truth. Additionally, we incorporate hits@1 as execution metric, which measures whether the first retrieved answer when actually querying the underlying knowledge graph matches the Mintaka gold standard. The Mintaka dataset reports exactly one correct result for each query, which is why we did not include further execution metrics such as micro precision and recall.

7 Quantitative Evaluation Results

We investigate the following research questions:

1. Does a controlled natural language as target for KGQA semantic parsing using LLMs have a positive effect on accuracy?
2. Are smaller amounts of training data sufficient to teach an LLM a controlled natural language as opposed to a formal language?
3. Does the answer to the above two questions depend on the choice of language model?
4. Does the answer to the first question depend on the question type?

We will walk through our research questions and connect them to the quantitative evaluation results we obtained. The first research question is whether using CNL can result in improved accuracy. We report our results in Table 2. Mintaka is generally a challenging dataset with hits@1 scores reported [25] to be between 0.12 and 0.2 on its test set for 3 different approaches. (Note that the Mintaka test set is a superset of our Mintaka-CNL test set and the reported approaches use weak supervision rather than strong supervision. Therefore, the results are not directly comparable.)

The hits@1 results for the two controlled natural languages, Sparklis and SQUALL, are both higher than the SPARQL baseline. Using SQUALL leads to higher accuracies for all models except T5, which could be due to it being even closer to natural language than Sparklis and generally being more compact. For the top-performing model, the accuracy using SQUALL is roughly twice as high compared to using SPARQL, so the choice of target language has a strong influence on the performance of the KGQA system. For the string similarity proxy metrics (BLEU, METEOR, ROUGE) SQUALL and Sparklis are similar, and consistently higher than SPARQL. For Sparklis, we used the exact match score as an estimate for hits@1 execution accuracy. This can be an overestimate in some cases since

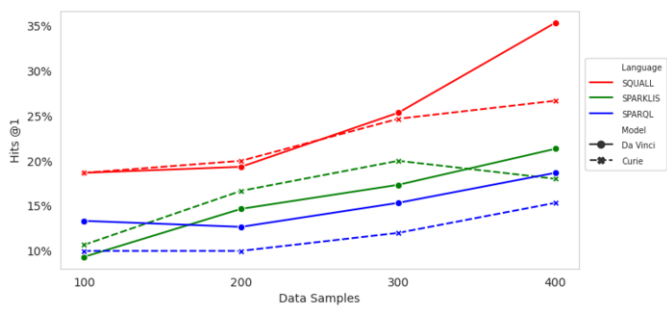

Figure 3. Illustration of hits@1 accuracy (y-axis) of the top performing models when using different amounts of fine-tuning examples (x-axis). Each line in the plot corresponds to a pair of language model and target language.

Table 2. Evaluation Results on the Mintaka-CNL dataset for different combinations of language models and target languages. The last column shows the absolute difference between hits@1 score of the controlled natural language and SPARQL.

Model	Setting	Language	BLEU	METEOR	ROUGE	Exact Match	Hits@1	Δ Hits@1
GPT-3 Davinci	Fine-Tuning-400	SPARQL	0.60	0.57	0.51	0.09	0.18	-
		Sparklis	0.67	0.71	0.61	0.20	0.20	+0.02
		SQUALL	**0.72**	**0.73**	0.60	0.18	**0.36**	**+0.18**
GPT-3 Curie	Fine-Tuning-400	SPARQL	0.60	0.58	0.52	0.09	0.15	-
		Sparklis	0.67	0.71	**0.62**	0.19	0.19	+0.04
		SQUALL	0.71	0.70	0.60	0.16	0.26	+0.11
Llama 2 7B	Fine-Tuning-400	SPARQL	0.63	0.65	0.46	0.05	0.11	-
		Sparklis	0.62	0.68	0.59	0.19	0.19	+ 0.08
		SQUALL	0.71	0.72	0.59	0.13	0.23	+0.12
T5-Large	Fine-Tuning-400	SPARQL	0.35	0.47	0.24	0.00	0.03	-
		Sparklis	0.66	0.7	0.61	**0.21**	0.21	**+0.18**
		SQUALL	0.7	0.7	0.58	0.11	0.19	+0.16
GPT Neo	Fine-Tuning-400	SPARQL	0.38	0.33	0.36	0.03	0.04	-
		Sparklis	0.54	0.61	0.55	0.08	0.08	+0.04
		SQUALL	0.64	0.65	0.68	0.18	0.18	+0.14
GPT-2 XL	Fine-Tuning-400	SPARQL	0.31	0.35	0.24	0.04	0.06	-
		Sparklis	0.54	0.6	0.51	0.12	0.12	+0.06
		SQUALL	0.58	0.63	0.54	0.11	0.15	+0.09
BLOOM 1.7B	Fine-Tuning-400	SPARQL	0.38	0.43	0.29	0.02	0.04	-
		Sparklis	0.61	0.58	0.57	0.10	0.10	+0.06
		SQUALL	0.59	0.64	0.63	0.09	0.13	+0.09
GPT-2 0.12B	Fine-Tuning-400	SPARQL	0.28	0.22	0.18	0.00	0.00	-
		Sparklis	0.43	0.51	0.47	0.04	0.04	+0.04
		SQUALL	0.41	0.39	0.33	0.08	0.12	+0.12

Table 3. Error categories we observed on the test set for GPT-3 (Da Vinci) fine-tuned on 400 examples.

Error Category	Category Definition	Count
(1) Entity linking	at least one entity required for building a correct query not found by ReFinED	18
(2) Entity selection	required entities provided by the ReFinED, but at least one selected incorrectly	6
(3) Relation generation	at least one relation in the ground truth not contained in the generated query	27
(4) Syntactically incorrect query	query cannot be parsed by the Squall → SPARQL converter	17
(5) Semantically incorrect query	query does not reflect meaning of the input question	12
(6) Query not matching KG structure	relationships used in the query don't match the KG structure	5
(7) Query correct, Mintaka result incorrect	generated query is correct, but Mintaka ground truth answer is not correct (outdated or was never correct)	2
(8) Query correct, Wikidata has no results	generated query is correct, but Wikidata does not return an answer	4

Table 4. Hits@1 scores for SQUALL and SPARQL query generation using GPT-3 Davinci fine-tuned on 400 examples.

Question Type	Count (test)	SPARQL hits@1	SQUALL hits@1	Improvement
ordinal	38	0.23	0.47	+ 0.24
difference	11	0.09	0.45	+ 0.36
generic	23	0.21	0.43	+ 0.22
intersection	21	0.28	0.33	+ 0.05
comparative	10	0.00	0.30	+ 0.30
count	28	0.25	0.21	– 0.04
superlative	18	0.00	0.11	+ 0.11

Sparklis is less expressive than SPARQL and SQUALL. This means that the ground truth queries may not capture all aspects of the input question (e.g. rather than selecting the "5th US president" the Sparklis query returns "US presidents" in increasing "point in time"). It can also be an underestimate since different query surface forms can lead to the same execution results. For both SPARQL and SQUALL, we observed that accuracy is usually much higher than the exact match score.

We also observed that the LLM sometimes generates SQUALL queries that are substantially different from the ground truth, but are semantically equally good and often lead to the same execution results. If we take this ability to generate correct but syntactically different queries as a proxy for the LLM obtaining a deeper understand-

Table 5. Examples for correct (top) and incorrect (bottom) parses of GPT-3 Davinci. For the incorrect parses, we describe one example per failure category.

Input	Generated SQUALL Query	Comment
What was the last Fast a Furious movie that Paul Walker starred in?	Which <film> is-<part of the series> <Fast & Furious> and has a <cast member> <Paul Walker> and has the latest <publication date>?	successfully detects that "Fast & Furious" is a series and "latest" refers to publication date
Who is a current senator of Utah and a former governor of Massachusetts?	Which <human> has <position held> <United States senator> at <electoral district> <Utah Class 1 senate seat> and has <position held> <Governor of Massachusetts> at an <end time>?	successfully builds a compositional query combining two criteria
What is Iron Man's real name?	What is the <birth name> of <Iron Man> ?	entity linking error: linked the movie instead of the fictional character
Which Indiana Jones movie made the most money?	Which <film> that is-<part of the series> <Indiana Jones> has the highest <box office>?	entity selection error: picked the figure Indiana Jones instead of the series
Which actors in Black Panther are no longer alive?	Which <human> that is-<part of the series> <Black Panther> has not a <cast member> <Black Panther>?	relation generation error: generated <cast member> instead of <cause of death>
Which Iron Man movie did Jeff Bridges not appear in?	Which <film> has <part of the series> <Iron Man> and has not <cast member> <Jeff Bridge>?	query correct, Mintaka lists the character instead of the movie as ground truth
Which member of The Police was not British?	Which <member> of <The Police> has not <country of citizenship> <United Kingdom>?	query correct, Wikidata has no results (anymore)
How many movies has Quentin Tarantino directed?	How many <film>-s has-<director> <Quentin Tarantino>?	syntactically incorrect since there is a dash after "has"
What basketball player became a senator?	Which <human> has <position held> <United States senator> at <series ordinal> 5?	semantically incorrect query asking for 5th term rather than for basketball occupation
How many Star Trek movies have Chris Pine in them?	How many <film>-s is-<part of the series> <Star Trek> and has <cast member> <Chris Pine>?	query not matching KG structure, which uses a specific entity type <Star Trek films>

ing of the target language (as opposed to simpler expression matching) then the pattern we see is: more natural target languages (see Figure 1) lead to a deeper understanding of the language by the large language model.

To answer the second research question, we varied the number of fine-tuning examples and plotted execution accuracies in Figure 3 for the two best performing LLMs. We observe that SQUALL is generally more accurate across all data sizes. For lower amounts of fine-tuning data SPARQL is more accurate than Sparklis for GPT-3 Davinci – probably because it was seen more in pre-training. Increasing the number of fine-tuning examples leads to Sparklis performing slightly better than SPARQL for GPT-3 Davinci with a constant margin. From our data, we cannot generally conclude that using a controlled natural language always reduces training data requirements - it is the case for SQUALL but to a lesser degree for Sparklis. So whether or not training data requirements for a CNL are substantially lower appears to be language-specific. SQUALL is both more natural and more compact than Sparklis, i.e. it can express a query in less tokens, which could explain the performance gap.

For the third research question, we computed the relative improvement for each language model in the execution accuracy of SQUALL compared to SPARQL in the last column of Table 2. It is interesting and surprising that the improvement from a formal language to a controlled natural language does not seem to depend much on the choice of the language model. Even for a small model like the GPT-2 124 million parameter model, which could not generate a single accurate SPARQL query, we observed substantial absolute improvements when using a controlled natural language.

For the fourth research question, we analyzed the different types of questions in Mintaka in terms of absolute performance and performance improvement when using SQUALL instead of SPARQL. The results are shown in Table 4. They indicate that the performance improvements are particularly high for difference, comparative, generic and ordinal questions. Some of those, e.g. comparative questions, are rather complex to phrase in SPARQL, which could explain the bigger performance differences for those types of questions. Generally, performance varies greatly depending on question type which could be explained by the types implying different query complexity.

8 Qualitative Evaluation and Failure Analysis

To analyse failure cases in more depth, we grouped them into categories as shown in Table 3. While generally a single generated query can have multiple failures, we followed a hierarchical approach to identify one root cause per query: We first checked whether the entity linker provides the entities required to build a correct query. Please note that we feed all entities above a threshold to the LLM, so in case of ambiguous cases the LLM needs to decide which one to pick and we do not require the entity linker to resolve the ambiguity completely. However, we do require (by the formulation of the prompt) that the correct entities are among those provided by the entity linker. If this is not the case, then we classify this as entity linking error. If this is the case, but the LLM does not use the correct entity, we classify this as entity selection error. If entities were selected correctly, but a relation is incorrect, we classified this as a relation generation error. The last column of Table 3 shows the frequency of those er-

rors on the test set for the best performing models. We can observe that entity linking and relation generation are major sources of errors. While the entity linking approach scales to millions of entities, it limits the ceiling performance of the LLM to roughly 88% hits@1 with the selected threshold. For larger training data sizes, the entity linking threshold should be moved further down.

Assuming that entities and relations were selected correctly, we then analyse whether there are syntactic or semantic errors. Syntactic errors should rarely happen with LLMs, but we observed several of them - most likely because SQUALL permits a variable structure allowing multiple expressions that map to the same query. With limited training data, the LLM has not fully captured which of those variations are allowed. Semantic errors are those where the generated query does not capture the intent of the question – in one case we could observe that the LLM hallucinated an additional irrelevant part of a query that was not correlated to the input question. The last 3 categories are queries that are not judged to be correct, because of a) Wikidata opting for a different KG structure than generated by the LLM, b) the Mintaka benchmark result not being correct or c) Wikidata not returning a result due to (never or no longer) having the data required for answering the query.

Table 5 shows some examples of correct and incorrect parses. We could observe that the most powerful LLM was able to perform correct semantic parses even for entities that were not top-ranked by the entity linker and for cases where the relations are unseen in the training data. However, we also observe that the limited training data set size makes it difficult for the LLM to adapt to unseen topics. While semantic parsers are often trained on orders of magnitude more training data than in our study, we would generally still recommend (as suggested by Figure 3) to include training data covering a wide range of topics allowing the LLM to observe KG modelling choices and relations. An interesting observation that we made is that the LLMs more frequently tend to generate repeated unnecessary patterns in SPARQL queries whereas this does not happen for SQUALL. This might be due to the model having a very good understanding of natural language and therefore also of controlled natural language whereas for SPARQL even GPT-3 can generate queries that are not meaningful, e.g. contain too many or repeated triple patterns.

9 Conclusions and Future Work

We verified our research hypothesis that using a controlled natural language as target for LLM-based KGQA semantic parsing provides advantages compared to translating into a logical form directly. This appears to hold despite the logical form being more prevalent in use and therefore in pre-training data. Our study with two different CNL shows that the more compact and natural SQUALL language leads to much better results compared to a more regular, verbose Sparklis language. Using the most powerful LLM in our test, the semantic parsing accuracy could be doubled by switching from SPARQL to SQUALL. We could also observe that the LLM likely obtains a better understanding of SQUALL, e.g. it can generate correct syntactic variations of the ground truth and very rarely generates meaningless or duplicated patterns. Interestingly, the benefits of CNLs appear to be relatively independent of Language Model size as similar improvements were observed for LM sizes ranging from 124 million to 175 billion parameters.

In future work, the CNL dataset could be extended and further target languages can be investigated. Moreover, we plan to investigate the usage of CNL beyond knowledge graph semantic parsing as interface for the LLM to external knowledge and services.

Acknowledgements

This work is partly supported by the BMBF (Federal Ministry of Education and Research) and DAAD (German Academic Exchange Service) in project SECAI (grant 57616814). It is also supported by EU projects CALLISTO (EU grant 101004152) and e-Vita (EU grant 101016453) as well as the ScaDS.AI (IS18026A-F) center for providing High Performance Computing infrastructure access at TU Dresden, Germany.

References

[1] Tom Ayoola, Shubhi Tyagi, Joseph Fisher, Christos Christodoulopoulos, Andrea Pierleoni, and Amazon Alexa AI, 'Refined: An efficient zero-shot-capable approach to end-to-end entity linking', *NAACL-HLT 2022*, 209, (2022).

[2] Satanjeev Banerjee and Alon Lavie, 'Meteor: An automatic metric for mt evaluation with improved correlation with human judgments', in *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72, (2005).

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al., 'Language models are few-shot learners', *Advances in neural information processing systems*, **33**, 1877–1901, (2020).

[4] Nilesh Chakraborty, Denis Lukovnikov, Gaurav Maheshwari, Priyansh Trivedi, Jens Lehmann, and Asja Fischer, 'Introduction to neural network-based question answering over knowledge graphs', *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **11**(3), e1389, (2021).

[5] Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann, '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, October 26–30, 2019, Proceedings, Part II 18*, pp. 69–78. Springer, (2019).

[6] Sébastien Ferré, 'SQUALL: The expressiveness of SPARQL 1.1 made available as a controlled natural language', *Data & Knowledge Engineering*, **94**, 163–188, (2014).

[7] Sébastien Ferré, 'Sparklis: An expressive query builder for sparql endpoints with guidance in natural language', *Semantic Web*, **8**(3), 405–418, (2017).

[8] Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su, 'Beyond iid: three levels of generalization for question answering on knowledge bases', in *Proceedings of the Web Conference 2021*, pp. 3477–3488, (2021).

[9] Aishwarya Kamath and Rajarshi Das, 'A survey on semantic parsing', *arXiv preprint arXiv:1812.00978*, (2018).

[10] T. Kuhn, 'A survey and classification of controlled natural languages', *Computational Linguistics*, (2013).

[11] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al., 'Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia', *Semantic web*, **6**(2), 167–195, (2015).

[12] Chin-Yew Lin and Eduard Hovy, 'Automatic evaluation of summaries using n-gram co-occurrence statistics', in *Proceedings of the 2003 human language technology conference of the North American chapter of the association for computational linguistics*, pp. 150–157, (2003).

[13] R. Montague, 'Universal grammar', *Theoria*, **36**, 373–398, (1970).

[14] Lunyiu Nie, Shulin Cao, Jiaxin Shi, Jiuding Sun, Qi Tian, Lei Hou, Juanzi Li, and Jidong Zhai, 'Graphq ir: Unifying the semantic parsing of graph query languages with one intermediate representation', in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5848–5865, (2022).

[15] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu, 'Bleu: a method for automatic evaluation of machine translation', in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, (2002).

[16] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al., 'Check your facts and try again: Improving large language mod-

els with external knowledge and automated feedback', *arXiv preprint arXiv:2302.12813*, (2023).
[17] Aleksandr Perevalov, Dennis Diefenbach, Ricardo Usbeck, and Andreas Both, 'Qald-9-plus: A multilingual dataset for question answering over dbpedia and wikidata translated by native speakers', in *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*, pp. 229–234. IEEE, (2022).
[18] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al., 'Language models are unsupervised multitask learners', *OpenAI blog*, **1**(8), 9, (2019).
[19] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, 'Exploring the limits of transfer learning with a unified text-to-text transformer', *The Journal of Machine Learning Research*, **21**(1), 5485–5551, (2020).
[20] Subendhu Rongali, Konstantine Arkoudas, Melanie Rubino, and Wael Hamza, 'Training naturalized semantic parsers with very little data', *arXiv preprint arXiv:2204.14243*, (2022).
[21] Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza, 'Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing', in *Proceedings of The Web Conference 2020*, pp. 2962–2968, (2020).
[22] Md Rashad Al Hasan Rony, Debanjan Chaudhuri, Ricardo Usbeck, and Jens Lehmann, 'Tree-kgqa: an unsupervised approach for question answering over knowledge graphs', *IEEE Access*, **10**, 50467–50478, (2022).
[23] Md Rashad Al Hasan Rony, Uttam Kumar, Roman Teucher, Liubov Kovriguina, and Jens Lehmann, 'Sgpt: A generative approach for sparql query generation from natural language questions', *IEEE Access*, **10**, 70712–70723, (2022).
[24] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al., 'Bloom: A 176b-parameter open-access multilingual language model', *arXiv preprint arXiv:2211.05100*, (2022).
[25] Priyanka Sen, Alham Fikri Aji, and Amir Saffari, 'Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering', in *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 1604–1619, (2022).
[26] Jiaxin Shi, Shulin Cao, Liangming Pan, Yutong Xiang, Lei Hou, Juanzi Li, Hanwang Zhang, and Bin He, 'Kqa pro: A large-scale dataset with interpretable programs and accurate sparqls for complex question answering over knowledge base', *arXiv preprint arXiv:2007.03875*, (2020).
[27] Richard Shin, Christopher Lin, Sam Thomson, Charles Chen Jr, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme, 'Constrained language models yield few-shot semantic parsers', in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7699–7715, (2021).
[28] Richard Shin and Benjamin Van Durme, 'Few-shot semantic parsing with language models trained on code', *arXiv preprint arXiv:2112.08696*, (2021).
[29] Claus Stadler, Jens Lehmann, Konrad Höffner, and Sören Auer, 'Linkedgeodata: A core for a web of spatial open data', *Semantic Web*, **3**(4), 333–354, (2012).
[30] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al., 'Llama 2: Open foundation and fine-tuned chat models', *arXiv preprint arXiv:2307.09288*, (2023).
[31] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann, 'Lc-quad: A corpus for complex question answering over knowledge graphs', in *The Semantic Web–ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II 16*, pp. 210–218. Springer, (2017).
[32] Denny Vrandečić and Markus Krötzsch, 'Wikidata: a free collaborative knowledgebase', *Communications of the ACM*, **57**(10), 78–85, (2014).
[33] Yushi Wang, Jonathan Berant, and Percy Liang, 'Building a semantic parser overnight', in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1332–1342, (2015).
[34] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec, 'Qa-gnn: Reasoning with language models and knowledge graphs for question answering', in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 535–546, (2021).