

# Bridging Language Models and Knowledge Graphs with Controlled Natural Languages

Dhananjay Bhandiwad<sup>a,\*</sup>, Preetam Gattogi<sup>a</sup>, Ashish Kangen<sup>a</sup>, Marco Basaldella<sup>b</sup>, Sébastien Ferré<sup>c</sup>, Sahar Vahdati<sup>a</sup>, Jens Lehmann<sup>d</sup>

<sup>a</sup>*ScaDS.AI - Technical University of Dresden, Dresden, Germany*

<sup>b</sup>*Amazon, Cambridge, UK*

<sup>c</sup>*University Rennes, Rennes, France*

<sup>d</sup>*Amazon, Dresden, Germany*

---

## Abstract

Knowledge graphs provide a source of up-to-date structured knowledge, which makes them an ideal counterpart to LLMs. LLMs, by themselves, are not trained to run structured queries internally and can become stale without a source of up-to-date information. We hypothesize that knowledge graphs can be effectively connected to large language models via controlled natural languages. Unlike standard formal query languages, controlled natural languages (CNLs) offer a syntax close to human language. Yet, can be unambiguously converted into formal languages such as SPARQL. In this article, we explore the premise that the extensive pre-training of LLMs on diverse textual data enables them to perform semantic parsing into controlled natural languages more accurately than parsing directly into formal query languages. To evaluate our hypothesis, we constructed a dataset facilitating the comparison between a standard formal language and two controlled natural languages.

Our findings show a significant accuracy improvement when using the same amount of controlled natural language training samples. Additionally, fewer samples are required to achieve a desired performance when using CNLs compared to standard query languages. The higher data efficiency of CNLs is particularly important to reduce the complexity and cost of the collection and curation. This enables a more efficient way for LLMs to query KGs.

*Keywords:* Large Language Models, Knowledge Graphs Question Answering, Semantic Parsing, Controlled Natural Language, Sample Efficiency, Entity Linking

---

## 1. Introduction

*Knowledge Graphs* (KGs) have become a crucial part of different AI applications by offering a means to capture factual information. At their core, KGs follow a triple-based structure to represent knowledge. They can capture cross-domain information like WikiData Vrandečić and Krötzsch (2014) and DBpedia Lehmann et al. (2015a), or are domain-specific such as Open Research Knowledge Graph<sup>1</sup>.

Semantically represented knowledge in KGs can be exploited to solve a wide range of problems in downstream NLP tasks such as recommendation or question answering Bellomarini et al. (2020). However, accessing information in knowledge graphs presents a significant challenge that can hinder their utility. To manually retrieve information from these graphs, users must navigate through them using specific query languages, such as SPARQL and Cypher, which demands a comprehensive understanding of both the query syntax and the schema of the graphs themselves.

Addressing this challenge, Knowledge Graph Question Answering (KGQA) systems have been developed to bridge the gap between the structured world of knowledge graphs and the intuitive simplicity of natural language. By allowing users to

pose questions in their own words (in natural language), KGQA systems aim to democratizing access to the rich stores of knowledge contained within KGs. They do this by translating natural language questions into formal queries that can retrieve specific information from the graph. This approach, also known as semantic parsing, significantly lowers the barrier for a wider audience to benefit from the knowledge stored in KGs.

The recent advances of Large Language Models (LLMs) are changing the perspective on the role of KGs: they now often serve as a knowledge backbone for LLMs, e.g. for a) facts that emerged after the LLM was trained, hence that are unknown to the model, b) facts that are not “popular”, i.e. not “head” knowledge, as the LLMs might not be able to learn them during training Sun et al. (2023), and c) facts that are out of the LLMs’ training scope, like domain-specific or organisation-specific knowledge, and d) aggregations of facts obtained by running structured queries.

One approach to connect LLMs with knowledge graphs is to employ them for semantic parsing themselves and perform the query execution as an action Yao et al. (2022). This allows LLMs to ground their responses with factual knowledge stored in a knowledge graph Peng et al. (2023); Yasunaga et al. (2021); Liu et al. (2025). Given that LLMs can hallucinate when uncertain and lacking the required factual knowledge to answer a query, this form of grounding can effectively reduce hallucinations.

---

\*Corresponding author.

E-mail address: dhananjay.bhandiwad@tu-dresden.de

<sup>1</sup><https://orkg.org/>

However, LLMs usually need fine-tuning to perform semantic parsing effectively on a specific KG. This is because they might not be proficient in the required query language or the target KG’s schema due to a lack of task-specific pretraining data.

Training datasets typically consist of pairs of natural language questions and their formal query counterparts. Developing these datasets requires considerable effort and resources, posing a substantial obstacle to incorporating them into LLM-based workflows. Despite attempts to mitigate these obstacles through semi-automated dataset generation techniques Dubey et al. (2019); Shi et al. (2020); Trivedi et al. (2017), the solutions often fall short in generalizability: many of these approaches rely heavily on templates. While beneficial for dataset expansion, inherently limit the diversity and real-world applicability of the training data. Such template-based datasets struggle to encapsulate the variability and complexity of natural language queries encountered in QA systems, thus hindering the ability of the models to generalize beyond the template’s scope.

We propose employing controlled natural languages as a bridge between LLMs and knowledge graphs. Our rationale is that CNLs closely mirror the natural language structure of input queries, facilitating a more intuitive translation process. They align more closely with the extensive textual data on which LLMs are pre-trained, potentially enhancing the efficacy of semantic parsing. In a previous publication of our research Lehmann et al. (2023), empirical evidence from our experiments confirmed the proposed hypothesis. In this work, we further extend our contributions in each of the following aspects as main components of the proposed approach:

- **Dataset:** we provide a new dataset called Mintaka-1k where the number of questions and their corresponding queries in several languages have been roughly doubled leading to 1000 questions. Each natural language question contains an equivalent SPARQL query as a standard KG query language, SQUALL and Sparklis as two CNLs, with two different versions for each language.
- **Prompt:** we propose four different Entity-Linking (EL) variations: *No EL* (no entity linking), *Implicit EL* (entity linking information incorporated into the training prompt), *Explicit EL* (post-training entity adjustment using an index map), and *Hybrid EL* (combination of Implicit and Explicit EL methods).
- **Language Model:** we test a wider combination of closed and open source models - Mistral-v0.2, Zephyr-7b-beta, Yi, and GPT-3.5-turbo in addition to the previous ones, i.e, T5, GPT-neo, GPT2-xl, GPT2, Bloom to achieve better results.
- **Target Language:** we introduce different versions for each target language, namely Label-based and ID-based, that demonstrate a significant improvement in execution accuracy when using natural language Wikidata labels instead of Wikidata IDs (ex: Using Barack Obama as label in the query surface forms instead of Q76). Further explanation can be found in the Section 4.

The remainder of this paper consists of the following structure: first, we introduce preliminaries and related work in Section 2. We introduce and discuss the proposed approach in Section 3. The evaluation setup and the development of the Mintaka-1k dataset are introduced in Section 4. An extensive discussion of the results is provided in Section 5, together with an error analysis in Section 6. We discuss the limitations of the work and future directions in Section 7 and conclude the contributions in Section 8.

## 2. Preliminaries and Related Work

This section provides an overview of the necessary preliminaries and related work for our research. We cover four key topics: Knowledge Graphs for Question Answering, Semantic Parsing, Integrating Large Language Models with Knowledge Graphs, and Controlled Natural Languages for Semantic Parsing. For each topic, we first introduce the essential background concepts, followed by a review of significant related work, ensuring a comprehensive understanding of the foundational and current related work.

### 2.1. Knowledge Graphs for Question Answering (KGQA)

Question answering (QA) systems aim to provide accurate and factual answers to questions in Natural Language (NL). Early QA systems primarily relied on information retrieval techniques to find relevant passages in unstructured text corpora Voorhees et al. (1999). Over the years, QA systems have evolved from rule-based approaches Riloff and Thelen (2000) to more advanced methods by integrating machine learning and deep learning techniques. These advancements have enabled QA systems to process more complex questions and achieve better performance by learning from large datasets and exploiting semantic relationships between entities and relations (properties).

The rise of knowledge graphs (KGs) has further revolutionized the field of QA, enabling a new paradigm known as knowledge graph question answering (KGQA) Yih et al. (2015), Fu et al. (2020). KGQA systems take advantage of the structured information encoded in KGs to provide more accurate and relevant answers to the NL question. By mapping NL questions to structured queries in the KG, these systems can retrieve relevant entities and relationships to retrieve factual answers.

Knowledge graphs serve as a structured representation of real-world entities and their relations. KGs such as DBpedia Lehmann et al. (2015b) and Wikidata Vrandečić and Krötzsch (2014) have been widely used in KGQA systems. These KGs typically model relationships as edges connecting nodes (entities) in a graph structure. These representations, however, can be limiting for complex relationships involving additional properties or attributes. To address this limitation, hyperrelational knowledge graphs, also known as hypergraphs or property graphs, have been introduced Angles and Gutierrez (2008). Hyperrelations KGs extend the knowledge representation by allowing relationships themselves to have properties, enabling the capture of more nuanced information about the nature of the connections between entities

In a hyperrelational KG, a statement  $t = (s, p, o, Q)$  is composed of a subject  $s \in E$ , a property  $p \in P$ , an object  $o \in (E \cup L)$ , and a set of qualifiers  $Q \subseteq P \times (E \cup L)$ , where  $E = e_1 \dots e_{n_e}$  is a set of entities,  $L$  is the set of all literal values, and  $P = \{p_1, \dots, p_{n_p}\}$  denotes a set of properties, also referred to as relations. The property  $p$  defines a relationship between the subject  $s$  and object  $o$  and the set of qualifiers  $Q$  further defines this relationship.

The Mintaka 1k dataset (refer to Section 4), introduced in this paper, leverages hyperrelational KGs to represent complex relationships. Several questions in the dataset employ hyperrelations via qualifiers. In the example, "Who was the 42nd president of the United States and former governor of Arkansas?", from the Mintaka-1k dataset, the KGQA system needs to identify the entities "President of the United States" and "Governor of Arkansas" and the qualifier "series ordinal" = 42.

## 2.2. Integrating LLMs with Knowledge Graphs

In recent years, Large Language Model (LLMs) such as OpenAI's GPT-4 Achiam et al. (2023), have significantly advanced the field of NLP, enabling a broad spectrum of applications including KGQA. Alongside GPT-4, other models including Mistral (7 billion parameters) Jiang et al. (2023), Zephyr (7B) Tunstall et al. (2023), and Yi (6B) have also made substantial contributions to enhancing NLP capabilities. A key highlight is that these models are open-sourced, making them accessible for research and development. Unlike the presumed 175B model GPT-3 Davinci, these models are relatively smaller, yet they demonstrate superior performance in various benchmarks. This advancement over LLMs used in our previous work Lehmann et al. (2023), including models like GPT-3 Davinci and Curie, sets the stage for improved performance in this paper.

Despite LLMs being pre-trained on extensive datasets, they still face the issue of hallucinations, inaccuracies, and outdated information in parametric knowledge.

Recent research Sen et al. (2023), Baek et al. (2023) has integrated KGs with LLMs, extracting triples from KGs and converting them into textual format to enhance LLM-based question-answering with knowledge-augmented prompts. Despite this progress, significant challenges in reasoning and parsing remain. For example, the increased complexity of question answering tasks demands more advanced reasoning capabilities and efficient exploitation of Knowledge Graphs to produce answers that are both natural and grounded Liu et al. (2025). This challenge is mostly seen in complex, multi-hop questions, which require retrieving and connecting multiple facts. LLMs often struggle with this type of reasoning, as they can easily get lost during the retrieval process. Furthermore, LLMs do not directly support KG-specific parsing without access to the entities present in the knowledge graph, a topic which is also analysed in depth in this paper 3. They, hence, require augmentation that facilitates entity retrieval. Further study by Wu et al. (2023) introduced a KG-to-Text corpus generation method for high-quality annotations and finetuned LLMs on this corpus, aiming to transform question-related subgraphs into knowledge text, thereby improving KGQA performance. The pipeline approach in KGQA systems, also described in Taffa and Usbeck (2023)

sequentially processes queries. It first identifies and maps relevant entities and relationships to a knowledge graph, then formulating and executing a SPARQL query to retrieve answers. Taffa and Usbeck (2023) introduces a KGQA system that directly translates questions into SPARQL queries in few-shot setting. Unlike JarvisQA Jaradeh et al. (2020), which relies on converting tabular data into text, and DBLP-QuAD Banerjee et al. (2023) requiring extensive training for new schemas.

## 2.3. Semantic Parsing

The task of translating an occurrence in Natural Language (NL) into a machine-interpretable representation  $q$  (query), which preserves the original meaning in a given formal language is termed semantic parsing. The logical form  $q$  is considered correct if it accurately captures the meaning of the input NL utterance.

Semantic parsing finds use in several applications; for instance, studies such as Xuan (2020), Meng et al. (2023) have contributed to enhancing dialog systems with efficient parsing of user utterances into structured commands. In question answering tasks, including Schneider et al. (2024), Gu et al. (2022) explore the utilization of semantic parsing to interpret and answer complex queries by leveraging large-scale knowledge bases. For information extraction, Li et al. (2022) showcases how semantic parsing can be augmented to extract relevant information from texts. In machine translation, Song et al. (2019) investigates the use of Abstract Meaning Representation (AMR) to enhance translation quality by preserving the original semantics when interpreting across languages.

Semantic parsing is also supported by large language models such as GPT-3 Brown et al. (2020), BLOOM Scao et al. (2022) or Mistral-7B Jiang et al. (2023).

They show promise in reducing the need for high amounts of training data with zero or few training examples, especially after incorporating several improvements such as few-shot learning Brown et al. (2020), Efficient attention mechanism (sparse attention Child et al. (2019), locality-sensitive hashing Kitaev et al. (2020), and knowledge distillation Hinton et al. (2015) that reduces the inference cost. Our focus is more specific to semantic parsing in the context of knowledge graphs. We refer to Chakraborty et al. (2021) for an introduction and survey of neural network-based KGQA semantic parsing. Modern KGQA semantic parsing methods mostly rely on sequence-to-sequence architectures, i.e., encoder-decoder framework with attention mechanisms. For example, in Rongali et al. (2020), an encoder translates a natural language instance into an intermediate state, which is then translated by the decoder into a logical form. Pointer generator networks See et al. (2017) enable copying some part of the input, e.g. a data value, to the output. Such approaches are demonstrably accurate when trained on large amounts of high-quality training data. However, obtaining training data is a challenge, and updating the training dataset to include new features or cover more domains requires significant effort. As a result, there was a surge of interest in training semantic parsers with less data; for instance, Rony et al. (2022a) explores an unsupervised approach for semantic parsing. While the results achieved are promising, these

approaches do not meet the performance criteria required for production-level systems and cannot be easily tuned to do so.

#### 2.4. *Controlled Natural Languages for Semantic Parsing*

According to Kuhn (2013), a Controlled Natural Language (CNL) is “a constructed language that is based on a certain natural language, being more restrictive concerning lexicon, syntax, and/or semantics, while preserving most of its natural properties.” CNLs aim to bridge the gap between the lack of ambiguity of formal languages such as SPARQL, and the high-level syntax of natural languages Kuhn (2013). CNLs enable abstraction from the low-level aspects of formal language such as bindings and relational algebra. Instances of SPARQL-oriented CNLs are Sparklis Ferré (2017) and SQUALL Ferré (2014) which are used to query and update knowledge graphs. For instance, SQUALL possesses an extensive coverage of SPARQL allowing both data querying and manipulation. It combines the expressivity of SPARQL, and simultaneously, the readability of natural languages. Its primary limitation is that the content words (e.g., proper nouns, verbs) must be unambiguous references to entities and relations. SQUALL is based on Montague grammars Montague (1970), which integrate context-free grammars, first-order logic, and  $\lambda$ -calculus. In case of Sparklis, the SPARQL queries are fully concealed behind a CNL and are constructed through an interactive endpoint explorer.

CNLs have served as a target for semantic parsing in contexts other than KGQA. For instance Rongali et al. (2022) has explored the idea of utilising a special-purpose CNL to mitigate training data requirements which then demonstrated improvements on the Overnight dataset Wang et al. (2015) in several smaller domains. Shin and Van Durme (2021) used a constrained decoding procedure for semantic parsing on the same (and other) datasets employing the GPT-3 and Codex models. They made a comparison between a CNL and a standard logical form language which showed the accuracy benefits of using a CNL. Although these studies serve as motivation for our investigation, they were performed on smaller datasets and not on knowledge graphs. The Overnight domain, for instance, has at most 45 relations, sharply contrasting with the over 10,000 relations (and in excess of 100 million entities) present in the Wikidata knowledge graph Vrandečić and Krötzsch (2014) that we use. Shin et al. (2021) explored the use of language models such as BART, GPT-2, and GPT-3 for constrained decoding of synchronous context-free-grammars. They followed a similar methodology as we do by first converting a NL question into a canonical human-like utterance which is further translated into the target logical form. These are again evaluated on Overnight and related datasets. Nie et al. (2022) introduces a new language GraphQ-IR, which is an intermediate representation that aims to unify the semantic parsing of graph query languages. The intermediate representation language used is in principles similar to CNL with less emphasis on naturalness compared to the SQUALL language that we investigated. Further, they use a BART-based encoder-decoder model in contrast to our approach which makes use of the generative capabilities of large language models.

#### 2.5. *Correctness Evaluation Metrics*

In Semantic parsing it is difficult to formalize the word “meaning”, hence several metrics are used to evaluate the correctness of semantic parsing. Among the set of metrics considered, execution metrics (e.g., Hits@1) measure whether the execution of the logical form provides the desired result. However, execution metrics alone are not a sufficient measure of correctness. This is because multiple, representations of the input question can produce the exact same result. This can occur when the metrics do not accurately represent the semantic meaning of the input query. However, this might be rare occurrence, since the datasets would not contain such spurious logical forms for the model to learn in the first place. For instance, translating “What is two times two” to  $2+2$  provides the correct result but is not the right translation. Text similarity metrics such as BLEU Papineni et al. (2002) or METEOR Banerjee and Lavie (2005) are generally used to quantify the closeness of the semantic parse with respect to the gold standard query. These are, however, not as significant as execution metrics for KGQA because there are often multiple correct logical forms for an input question. For example, the correct translation for “Who is the author of the book “1984”?” in Wikidata could be querying the property “author” directly linked to ‘1984’ (Q208460) or via a more complex path involving “written work” and “creator” properties.

### 3. *The BridgeKGLM Approach*

The primary research objective of this work is to assess the effectiveness of LLMs in semantic parsing across different target languages i.e, SQUALL, Sparklis, and SPARQL. To achieve this, we have developed an approach dubbed *BridgeKGLM*, which is uniformly applicable across these three languages. The workflow of *BridgeKGLM* is illustrated in Figure 1. As shown, the workflow is initiated by an input natural language question. Followed by an entity linking (EL) process for a given question. Next, We perform supervised fine-tuning on the LLMs with the Mintaka-1k dataset that is described in Section 4 incorporating specific entity linking information provided by the entity linking variation described in Section 3.1. Finally, the received question is translated into the required target language, according to the specified instructions in the prompt.

The NLP task of entity Linking involves assigning each entity mentioned in an input a unique identity which refers to a unique entity (node) in the underlying knowledge graph. The *BridgeKGLM* approach is not dependent on any specific entity linker; in our experiments, we use ReFinED Ayoola et al. (2022). We choose this tool as it is runtime efficient and it produces state-of-the-art results on Wikidata.

#### 3.1. *Entity Linking*

In this section, we describe the process of constructing an entity map that assigns unique labels to entity IDs. Following this, we detail our experiments with four different variations of entity linking. Entity linking is initially performed in order for the

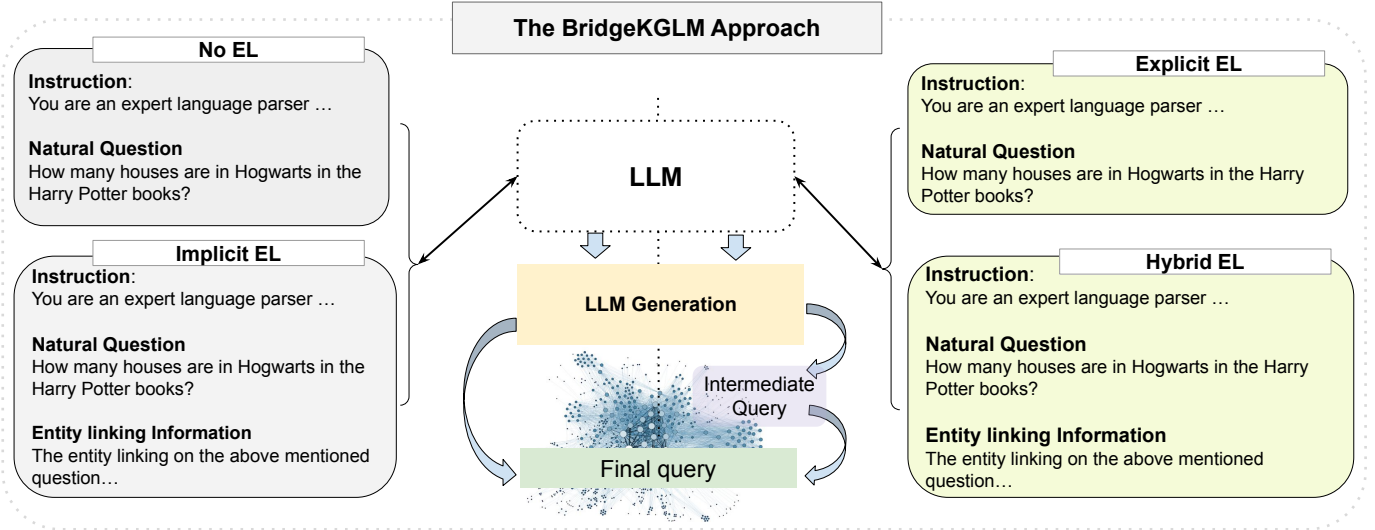


Figure 1: The Graphical Abstract of *BridgeKGLM* approach: First, Entity Linking (EL) information is retrieved. The entities with additional information are fed into the prompt construction for Implicit and Hybrid EL. For Explicit EL and Hybrid EL the LLM generates an intermediate query whose entities are adjusted to give the final query.

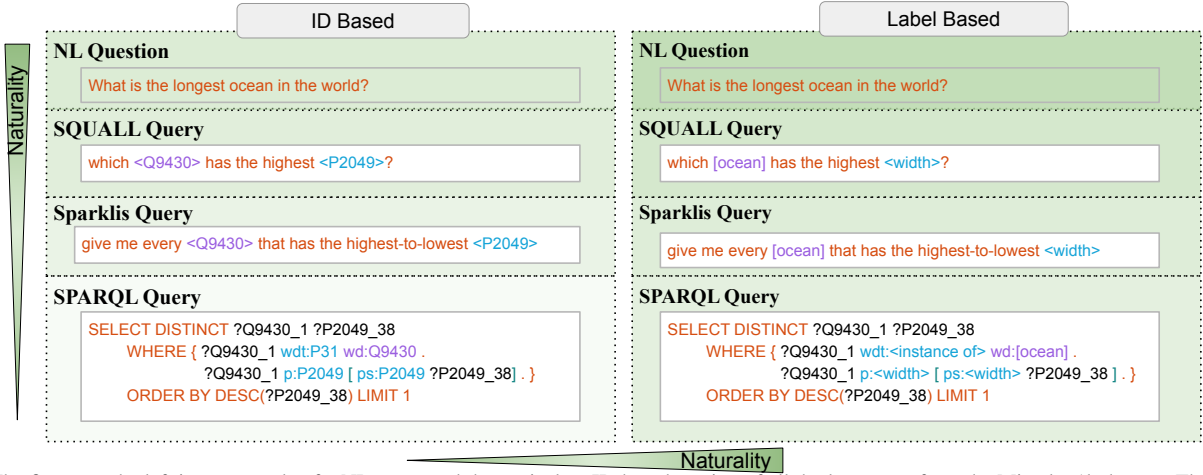


Figure 2: The figure on the left is an example of a NL query and the equivalent ID-based version of all the languages from the Mintaka-1k dataset. The figure on the right is an example of an NL query and the equivalent Label-based version of all the languages from the Mintaka-1k dataset.

model to have information about the entities present in the input natural language question. This information, in turn is used by the language model to formulate the query in the target languages (SQUALL, Sparklis, and SPARQL). For the cases with entity disambiguation i.e; when different QIDs share the same label, an *EntityMap* is constructed to facilitate the disambiguation of these entities in our dataset.

**EntityMap.** A dictionary is constructed for the entities within the Mintaka-1k dataset, where the keys are the Wikidata labels and the corresponding QIDs are the values. In some cases, different QIDs share the same label. These labels are disambiguated by appending a unique number during the dictionary creation process. This is performed as we pass through the Mintaka-1k dataset, which also ensures that all the keys in our dictionary are unique. It should be noted that we also add entities from ReFinED entity linker to the dictionary as we make a pass through the dataset. These entities are added only if their scores, as returned by the ReFinED entity linker and are above

the predetermined threshold of 0.1. The constructed dictionary is referred to as “EntityMap” in this paper.

**Entity Linking Variations.** We consider four variations of entity linking for Label-based versions of logical forms. These variations are a combination of (a) whether entity linking information is passed to the LLM through the prompt; and (b) whether the queries generated by the LLM are post-processed for entity adjustment.

1. **Without Entity Linking (w/o EL):** For the baseline, we do not include any entity linking information to the models in the prompt during the supervised fine-tuning. The input prompt consists of only an instruction to convert the natural language question into the respective query languages.
2. **Explicit Entity Linking (Explicit EL):** In this setup, no entity information is included in the prompt while performing supervised fine-tuning. There is, however, a post-

ad-hoc entity adjustment step that performs entity linking on the model-generated output using EntityMap. For this, initially, all the entities are extracted from the model output. Then, sentence transformer (all-MiniLLM-L6-v23) is used to embed both the keys from the EntityMap and the entities from the model output into a 384-dimensional vectors. As a final step, cosine similarity is calculated between these embeddings (the keys from EntityMap and the entities from the model output). Finally, each entity in the model output is replaced with the most similar key from the EntityMap.

3. **Implicit Entity Linking (Implicit EL):** In this setup, all the entities above a specified probability threshold value are retrieved by ReFinED for a given input; this threshold is considered as a tunable hyperparameter in the proposed semantic parsing approach. We retrieve the label, Entity ID, likelihood and description for every entity. The LLM prompting step however, receives only a triple i.e. (label or Entity ID, likelihood, and description), for each entity in the question as an input. The choice to use either the label or the Entity ID depends on the language versions being used. This is further defined in Section 4.

The underlying LLM uses this supplementary information provided in the prompt while performing supervised fine-tuning for semantic parsing.

For example, the probability in the entity linking prompt could cause the LLM to favor the highly likely entities over the less likely ones. Additionally, the entity label and description in the prompt can help the LLM to disambiguate entities by providing more context which is an implicit task in semantic parsing. It should also be noted that the entity label by itself does not guarantee uniqueness. For example (“Harry Potter” label has multiple pages in Wikidata, such as a book, series, or fictional character).

4. **Hybrid Entity Linking (Hybrid EL):** In this setup, a combined version of entity linking, integrating Implicit EL and Explicit EL is implemented. More precisely, the entity linking information is passed to the prompt of the LLM as mentioned in Implicit EL. A post-ad-hoc entity adjustment step is performed as mentioned in Explicit EL. Combining both Implicit EL and Explicit EL leads to an added advantage of mitigating hallucinated entities by correcting entities generated by the LLM.

### 3.2. Relation Linking

The NLP task of relation linking involves assigning each relationship that is relevant to an input utterance a unique identity. We allow the LLM to create the relation as part of the query creation process rather than employing a separate relation linker like in case of entities. The reason for these actions are as follows:

1. Relations have a large range of surface forms, and can even be implicit in the input question. Therefore, relation linking is typically less accurate than entity linking. Considering how hard this process is, the LLM itself might be a good solution for this task, as it is plausible that the LLM might have seen enough relations in its pre-training data.

Table 1: Models used for fine-tuning and their respective parameter count.

Model name	Provided by	Release date	#Parameters
Mistral-v0.2	Mistral AI	September 2023	7 B
Zephyr-7B Beta	Hugging Face	October 2023	7 B
Yi-6B	01-ai	November 2023	6 B
GPT-3.5 Turbo	OpenAI	August 2023	presumably(175B)
GPT-2	OpenAI	February 2019	0.12 B
T5-Large	Google	February 2020	0.77 B
GPT Neo	EleutherAI	March 2021	1.3 B
GPT-2 XL	OpenAI	November 2019	1.5 B
Bloom	BigScience	July 2022	1.7 B

2. Compared to entities, knowledge graphs often have fewer relations. As of May 2024, Wikidata approximately has over 100 million entities but only about 11,000 relations. Therefore, the training data will only comprise a tiny fraction of existing entities, even though it may contain an adequate amount of frequent relations.

### 3.3. Prompting

The designed prompt consists of three main components in the LLM prompt: instruction, natural question, and entity linking information. the latter includes a list of entities and their information (for Implicit EL and Hybrid EL). We note that, for the variants w/o EL and Explicit EL, the entity linking information is not included. The instruction is as follows: “*You are an expert language parser and you are able to convert from Natural Question to \$language query. Below are the <Question> and <Entity Linking Information> defined for you to create the query*”, where \$language  $\in$  {SPARQL, Sparklis, SQUALL} represents the target language. The LLM receives the natural question, followed by the triple (label or Entity ID, likelihood, description) for the entities in the manner detailed in the section 3.1. Figure 3 showcases the Label-based prompt, while Figure 4 indicates the ID-based version prompt. Our studies prioritize the performance difference between various target languages rather than achieving performance optimization, for this reason, we do not perform prompt engineering. While this choice may impact the absolute ceiling performance, its significance is unlikely to be substantial between the target languages.

Table 2: Entity linking variations and their TA results for the Label-based version of the languages

Model Name	Language	w/o EL	Explicit EL	Implicit EL	Hybrid EL
Zephyr-7B Beta	SQUALL	0.55	0.58	0.58	0.60
Zephyr-7B Beta	Sparklis(EM)	0.25	0.25	0.30	0.30
Zephyr-7B Beta	SPARQL	0.45	0.47	0.53	0.54
Mistral	SQUALL	0.52	0.55	0.58	0.60
Mistral	Sparklis(EM)	0.29	0.29	0.29	0.29
Mistral	SPARQL	0.53	0.56	0.50	0.52
Yi-6B	SQUALL	0.51	0.54	0.53	0.55
Yi-6B	Sparklis(EM)	0.25	0.25	0.27	0.27
Yi-6B	SPARQL	0.47	0.51	0.46	0.47

```

###
You are an expert language parser and you are able
to convert from Natural Question to SQUALL query.
Below are the <Question> and <Entity Linking
Information> defined for you to create the query.

###Question:
Which Iron Man movie was not directed by Jon
Favreau?

###Entity Linking Information:
The entity linking on the above mentioned question
gave the following entities as results:
For the text span Iron Man (2008 film) the candidates
within square brackets are below:
[The entity <Iron Man 3>, a 2008 film directed by
Jon Favreau, was identified with a probability 0.3713],
[The entity <Iron Man>, a fictional character in Marvel
Comics, was identified with a probability 0.105]

For the text span Jon Favreau the candidates within
square brackets are below:
[The entity <Jon Favreau>, an American actor and
filmmaker (born 1966), was identified with a probabili-
ty 1.0]

###Answer:

```

Figure 3: Label-based prompt including the instruction to the model, with the Natural Question, followed by entity linking information related to the specific question. The model is expected to output the Label-based version of SQUALL query.

#### 4. Dataset and Experimental Setup

**Development of the Mintaka-1k Dataset.** As previously mentioned, one of the contributions of this work is the preparation and development of a dataset dubbed Mintaka-1k. The original Mintaka Sen et al. (2022) dataset serves as the fundamental dataset for our research. Specifically designed for KGQA, Mintaka is a question-answering dataset that is characterized as being both natural and multilingual in nature and has 20,000 handwritten English NL queries in it. The Mintaka dataset is further characterized by a large number of queries and does not contain questions that are, at least in part, automatically generated. Furthermore, for the development of the Mintaka-1k dataset, the following procedure was established.

- **Sample Selection** The following criteria were used to se-

Table 3: Comparison between SPARQL and CNL-based Approaches

Feature	SPARQL	CNL-based Approaches
Syntax Naturality	Low – Formal query language	High – Close to natural language
Learning Curve	Steep – Requires understanding of formal syntax and KG schema	Gentle – More intuitive for non-technical users
Pre-training Benefits	Some benefit from LLM pre-training	Better leverages LLM pre-training on natural text
Error Rates	Higher rates of syntactic and semantic errors	Lower error rates, especially for syntactic errors
Accessibility	Requires technical expertise	More accessible to non-technical users

lect questions from Mintaka to generate our dataset: a) inclusion of diversity in question themes, b) varying degree of question complexity, c) questions with multiple entities and d) each category contains at least three samples of similar question.

- **Query Generation** The two equivalent CNL’s i.e; SQUALL and Sparklis along with SPARQL logical forms were developed after sampling relevant Mintaka questions. The first step involved utilizing Sparklis’ query creation interface<sup>2</sup> to convert these questions into Sparklis queries. Despite the fact that creating Sparklis was taking several minutes (up to 3 mins) per query on average, it significantly speeded up the process of creating SPARQL queries compared to manually creating them as described below. By conducting several workshops, the Sparklis queries were created. The workshop involved three domain expert researchers and one student.
- **Query Conversion** The obtained Sparklis queries from the above step were automatically converted into SPARQL queries using Sparklis’ interface. The SPARQL query were then converted deterministically into a SQUALL query using the backward translation tool<sup>3</sup>. SQUALL, in general, tends to offer greater expressiveness compared to Sparklis in scenarios where certain features are not immediately supported, such as selecting the  $n$ -th element based on an order or comparing relation values like height and length. Therefore, some of the SPARQL queries produced by Sparklis tool required manual correction, e.g. adding OFFSET and LIMIT clauses, in order to benefit from the higher expressiveness of SQUALL. Moreover, SQUALL permits a direct conversion into SPARQL, thereby enabling us to verify if both queries yielded identical results.

In the original version of the Mintaka dataset, questions are classified into seven categories: count, intersection, superlative, difference, comparative, generic, and ordinal. The yes/no question category from the original Mintaka dataset was not included in this work due to Sparklis’ inability to handle SPARQL ASK queries. A total of 1000 questions were enriched from the Mintaka dataset, and the questions are spread over the different categories as follows: 154, 140, 142, 137, 109, 133, and 185, with three distinct target forms created for each, specifically, SPARQL, SQUALL, and Sparklis. This dataset was split into 700 train and 300 test questions respectively. Two sub-datasets namely *Label-based* and *ID-based* versions for each of the logical forms were created. This was done to further analyse the performance across different representations of language naturalness. In the ID-based versions, the Entities and Relations in each target form are enclosed in angular brackets (<>) as QIDs and PIDs. An example of ID-based version in SQUALL would be; *which <Q11424> has <P179> <Q194318> and has the highest <P2047>?.* This format is less intuitive or human-readable but more precise, as it directly references the unique

<sup>2</sup>Available online at <http://www.irisa.fr/LIS/ferre/sparklis/>

<sup>3</sup><https://bitbucket.org/sebferre/squall2sparql/>

```

###
You are an expert language parser and you are able to convert from Natural Question to SPARQL query. Below are the
<Question> and <Entity Linking Information> defined for you to create the query.

###Question:
Which Iron Man movie was not directed by Jon Favreau?

###Entity Linking Information:
The entity linking on the above-mentioned question gave the following potential results:
For the text span Iron Man (2008 film) the candidates within the square brackets are below:
[The entity <Q192724>, a 2008 film directed by Jon Favreau, was identified with a probability 0.3713], [The entity
<Q180704>, an fictional character in Marvel Comics, was identified with a probability 0.105]

For the text span Jon Favreau the candidates within square brackets are below:
[The entity <Q295964>, an American actor and filmmaker (born 1966), was identified with a probability 1.0]

###Answer:

```

Figure 4: ID-based prompt including the instruction to the model, with the Natural Question, followed by entity linking information related to the specific question. The model is expected to output the ID-based version of the SQUALL query.

Table 4: The below table shows evaluation results on the Mintaka-1k dataset for Large Language Models for different types of target languages, i.e. SPARQL, SQUALL and Sparklis with two variations, i.e. ID-based and Label-based. The last column, for a given model, shows the difference in Translation Accuracy (TA) between different languages and ID-based SPARQL  $\Delta_L$  TA

Model	Language	L-Type	BLEU	METEOR	ROUGE	Exact Match	Hits@1(Mintaka)	TA	$\Delta_L$ TA
Zephyr-7B Beta	SQUALL	Label-based	0.67	0.82	0.76	0.40	0.40	0.58	+0.14
Zephyr-7B Beta	SQUALL	ID-based	0.55	0.75	0.66	0.36	0.35	0.50	+0.06
Zephyr-7B Beta	Sparklis	Label-based	0.65	0.82	0.75	0.30	-	-	-
Zephyr-7B Beta	Sparklis	ID-based	0.56	0.74	0.67	0.26	-	-	-
Zephyr-7B Beta	SPARQL	Label-based	0.58	0.76	0.70	0.21	0.38	0.53	+0.09
Zephyr-7B Beta	SPARQL	ID-based	0.49	0.70	0.63	0.19	0.31	0.44	-
Mistral-v0.2	SQUALL	Label-based	0.65	0.82	0.75	0.39	0.41	0.58	+0.14
Mistral-v0.2	SQUALL	ID-based	0.55	0.75	0.67	0.35	0.33	0.47	+0.03
Mistral-v0.2	Sparklis	Label-based	0.65	0.81	0.75	0.29	-	-	-
Mistral-v0.2	Sparklis	ID-based	0.57	0.75	0.67	0.26	-	-	-
Mistral-v0.2	SPARQL	Label-based	0.55	0.72	0.67	0.21	0.35	0.50	+0.06
Mistral-v0.2	SPARQL	ID-based	0.48	0.68	0.62	0.20	0.32	0.44	-
Yi-6B	SQUALL	Label-based	0.64	0.80	0.73	0.40	0.35	0.53	+0.17
Yi-6B	SQUALL	ID-based	0.52	0.73	0.64	0.33	0.30	0.44	+0.08
Yi-6B	Sparklis	Label-based	0.63	0.79	0.72	0.27	-	-	-
Yi-6B	Sparklis	ID-based	0.55	0.73	0.65	0.25	-	-	-
Yi-6B	SPARQL	Label-based	0.55	0.74	0.67	0.20	0.31	0.46	+0.10
Yi-6B	SPARQL	ID-based	0.47	0.68	0.60	0.15	0.22	0.36	-
GPT-3.5-0125	SQUALL	Label-based	0.62	0.80	0.72	0.32	0.30	0.48	+0.19
GPT-3.5-0125	SQUALL	ID-based	0.50	0.72	0.62	0.28	0.49	0.42	+0.13
GPT-3.5-0125	Sparklis	Label-based	0.57	0.77	0.69	0.22	-	-	-
GPT-3.5-0125	Sparklis	ID-based	0.51	0.71	0.63	0.19	-	-	-
GPT-3.5-0125	SPARQL	Label-based	0.54	0.73	0.66	0.21	0.32	0.46	+0.17
GPT-3.5-0125	SPARQL	ID-based	0.39	0.63	0.54	0.09	0.20	0.29	-

identifiers of Wikidata. Conversely, in the Label-based versions of the logical forms, the QID and PID were substituted with the corresponding Wikidata labels, which adopts a more natural and human-readable format. In these instances, entity labels were enclosed in square brackets ([ ]), while property labels were enclosed in angle brackets (<>). An example of a Label-based query in SQUALL would be; *which [film] has <part of the series> [Pirates of the Caribbean] and has the highest <duration>?*.

**Language models and Fine-Tuning.** The language models of different sizes shown in Table 1 were chosen for appropriate generative natural language processing applications. For fine tuning procedure, we used prompt as described in section 4 and the actual target query was used as the completion for all models. To allow for special tokens, the tokenizer was changed and the model’s token embeddings were scaled for all models except GPT-3.5 Turbo. The settings mentioned in Rony et al. (2022b) were followed during the fine-tuning phase: 30 epochs,

Table 5: The table below shows the evaluation results on the Mintaka-1k dataset for Language Models for different types of target languages, i.e. SPARQL, SQUALL and Sparklis with two variations, i.e. ID-based and Label-based. The last column, for a given model, shows the difference in TA (Translation Accuracy) between different languages and ID-based SPARQL  $\Delta_L$  TA.

Model	Language	L-Type	BLEU	METEOR	ROUGE	Exact Match	Hits@1(Mintaka)	TA	$\Delta_L$ TA
T5	SQUALL	Label-based	0.65	0.81	0.75	0.39	0.35	0.54	+0.12
T5	SQUALL	ID-based	0.56	0.76	0.68	0.36	0.32	0.47	+0.03
T5	Sparklis	Label-based	0.60	0.80	0.71	0.28	-	-	-
T5	Sparklis	ID-based	0.51	0.72	0.63	0.23	-	-	-
T5	SPARQL	Label-based	0.57	0.76	0.69	0.20	0.39	0.51	+0.07
T5	SPARQL	ID-based	0.50	0.72	0.63	0.20	0.31	0.44	-
<hr/>									
GPTNeo	SQUALL	Label-based	0.63	0.79	0.72	0.32	0.35	0.47	+0.11
GPTNeo	SQUALL	ID-based	0.50	0.72	0.63	0.28	0.24	0.37	+0.01
GPTNeo	Sparklis	Label-based	0.61	0.79	0.71	0.23	-	-	-
GPTNeo	Sparklis	ID-based	0.55	0.73	0.65	0.20	-	-	-
GPTNeo	sparql	Label-based	0.55	0.73	0.67	0.19	0.29	0.41	+0.05
GPTNeo	sparql	ID-based	0.47	0.68	0.61	0.16	0.27	0.36	-
<hr/>									
GPT2-XL	SQUALL	Label-based	0.61	0.79	0.71	0.33	0.32	0.47	+0.14
GPT2-XL	SQUALL	ID-based	0.52	0.72	0.64	0.31	0.28	0.40	+0.07
GPT2-XL	Sparklis	Label-based	0.63	0.79	0.72	0.23	-	-	-
GPT2-XL	Sparklis	ID-based	0.53	0.72	0.63	0.18	-	-	-
GPT2-XL	SPARQL	Label-based	0.54	0.72	0.66	0.20	0.27	0.38	+0.05
GPT2-XL	SPARQL	ID-based	0.47	0.68	0.61	0.16	0.23	0.33	-
<hr/>									
GPT2	SQUALL	Label-based	0.58	0.75	0.68	0.30	0.28	0.37	+0.05
GPT2	SQUALL	ID-based	0.48	0.70	0.61	0.26	0.25	0.38	+0.06
GPT2	Sparklis	Label-based	0.57	0.76	0.68	0.20	-	-	-
GPT2	Sparklis	ID-based	0.50	0.69	0.61	0.14	-	-	-
GPT2	SPARQL	Label-based	0.52	0.70	0.64	0.15	0.23	0.32	+0.00
GPT2	SPARQL	ID-based	0.45	0.66	0.59	0.17	0.24	0.32	-
<hr/>									
BLOOM	SQUALL	Label-based	0.57	0.75	0.67	0.26	0.25	0.36	+0.07
BLOOM	SQUALL	ID-based	0.48	0.70	0.60	0.23	0.19	0.30	+0.01
BLOOM	Sparklis	Label-based	0.57	0.76	0.68	0.18	-	-	-
BLOOM	Sparklis	ID-based	0.51	0.70	0.61	0.13	-	-	-
BLOOM	SPARQL	Label-based	0.46	0.65	0.58	0.07	0.11	0.16	-0.13
BLOOM	SPARQL	ID-based	0.45	0.67	0.59	0.14	0.20	0.29	-

$2.5e-5$  as the constant learning rate, and a batch size of 2. Extensive hyperparameter optimisation was not carried out, and the same configurations were kept for each target language.

In the decoding step, SQUALL, Sparklis, and SPARQL were all subjected to a temperature value of 0.7. At this temperature value, it was found that the GPT-3.5-Turbo model produced the best inference. To achieve optimal generation during inference, the temperature was set to 0.01 for three Large open-source models: Mistral-v0.2, Zephyr-7b-beta, and Yi. Prior to assessment, special tokens were eliminated from the LLM output. A threshold of 0.1 was chosen for the Entity linking. Two Nvidia A100 GPUs running on Linux were used for open-source models and OpenAI servers were used for the GPT-3.5 Turbo fine-tuning. Python 3.9 was used to write the code.

**Evaluation Metrics.** For evaluation, a combination of translation quality metrics and execution metrics were employed (see Section 2.5): BLEU Cumulative Papineni et al. (2002), METEOR Banerjee and Lavie (2005), ROUGE Lin and Hovy (2003) 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 a granular level. We also use Exact Match (EM) accuracy to evaluate the percentage of queries matching the ground truth.

Further, Hits@1(Mintaka) and TA(Translation Accuracy) were used as execution metrics. Hits@1(Mintaka) is computed by comparing the answers returned by executing model generated queries on Wikidata with the gold standard answers from the Mintaka-1k dataset. When direct execution metrics like Hits@1(Mintaka) were used, as in our previous work, there were several sources of failure that reduced the execution hits. Some of these were due to the query being correct and the Mintaka answer being outdated or incorrect, Wikidata not returning an answer or Wikidata having the wrong answer. In order to eliminate sources of error that depend on the dataset and knowledge graph, Translation Accuracy (TA) metric was introduced to evaluate the effectiveness of the semantic parse. TA is a union of two components,  $A_1$  and  $A_2$ .  $A_1$  is computed by comparing the answers returned by executing model-generated queries on Wikidata with the answers of the executed gold standard SPARQL queries from the Mintaka-1k dataset. For the

cases where the model-generated query and the gold standard query both return no entities,  $A_1$ , however, would not provide a reliable evaluation. For instance, the predicted query could be semantically different from the gold one. In such a case, we use  $A_2$  as a fallback, which checks whether the model-generated query is the same as the gold standard query from the Mintaka-1k dataset.

Essentially, TA metric would deem an execution correct if it has managed to capture the semantics of the original NL query accurately. Furthermore, we believe TA is a stronger measure than Hits@1(Mintaka), as a correct execution is yielded if the semantic parse is accurate

## 5. Results and Discussion

This section illustrates the results obtained from several experiments we have performed to demonstrate the viability of CNLs and the effect different entity-linking variations have on our *BridgeKGLM* approach. In the subsequent sections, we first evaluate the performance of various entity linking variants whose results are shown in Table 2. The analysis of our quantitative evaluations for the LLMs is detailed in Table 4 comparing the performance over four leading models on both label and ID based over three target languages. SQUALL consistently performs better in terms of TA compared to SPARQL. Label-based variants outperform ID-based variants across all languages, validating our hypothesis that natural representation improve LLM performance. Column  $\Delta_L$  TA column reveals the magnitude of improvement CNLs offer over SPARQL. SQUALL shows the most consistent advantage, with improvements ranging from +0.06 to +0.19, while GPT-3.5 exhibits the largest gains (+0.19 for Label-based SQUALL). Additionally, we also assess the sample efficiency of CNLs and SPARQL, with the results shown in Tables 6, 7 and 9. Note that the Implicit EL entity linking variation is used in our *BridgeKGLM* approach for the results displayed in these tables. This is done as the cosine similarity post-hoc correction step is only valid for the labels of the entities in our dataset, not the QIDs themselves, which precludes its usage on the ID-based variants of the languages in our study. Table 5 extends our analysis to smaller Language Models (LMs), evaluating T5, GPTNeo, GPT2-XL, GPT2, and BLOOM across the same language variations. The similar performance gain could be seen in the smaller LMs as well. Bloom, GPT2 here are an anomaly, where for example, BLOOM: the ID-based SPARQL (0.29 TA) outperforms Label-based SPARQL (0.16 TA). This suggests that the smaller models may struggle more with complex tokenization requirements of label-based SPARQL. Further, we attribute this to the fact that the smaller models are less intelligent. Given that they are likely trained on SPARQL, they lack the ability to translate the QIDS to their respective textual representation. The  $\Delta_L$  TA values generally show positive improvements for CNLs, with T5 and GPTNeo exhibiting particularly strong advantages for SQUALL (+0.12 and +0.11 respectively).

### 5.1. Exploring Impact of Entity Linking Variations on KGQA Semantic Parsing Accuracy

This section explores the impact of Entity Linking variants on KGQA Semantic Parsing accuracy. As shown in Table 2, comparing the performance of our Semantic Parsing(SP) pipeline w/o EL and Explicit EL, it is observed that parsing with Explicit EL consistently performs on par, and often better than, w/o EL. This aligns with our hypothesis that for a given question, the parsing accuracy is improved when substituting incorrectly generated labels with the correct ones from the EntityMap, based on similarity. This substitution increases the likelihood of generating executable queries compared to w/o EL. Further, when comparing Explicit EL and Implicit EL, it is observed that Explicit EL outperforms Implicit EL in three instances, while Implicit EL outperforms Explicit EL in four instances. This can be explained by considering the case of entity disambiguation. In the Implicit EL case, the model must learn to disambiguate between the provided entities, which may not be efficiently learned by all the models. Instead, Explicit EL does not directly address entity disambiguation and allows the models to generate potentially incorrect labels, which are then corrected using our ad-hoc post-processing step with cosine similarity, as described in Section 3.

Two notable exceptions are observed when comparing Implicit EL to the w/o EL case: Mistral SPARQL and Yi SPARQL. In the w/o EL case, Mistral SPARQL outperforms Mistral SQUALL. The w/o EL models outperforming the Implicit EL models can be attributed to the overlap between the entity set of the train and test dataset. Despite not being provided with explicit entity information, we believe these models likely learned entity labels from the training data and could have already learned some entities during pretraining.

This inference is easier in Label-based SPARQL queries, as the natural language entity labels stand out from the rest of the query. Since some test dataset questions refer to the same entities as the training dataset, the models only need to formulate the syntax and substitute the correct entities observed during fine-tuning to obtain the correct semantic parse. The syntax formulation is further aided by the fact that these models have already been pre-trained on SPARQL, leading the models to largely learn entity substitution and also, likely, to learn some very common Wikidata entities. In the Implicit EL case, the model has to additionally learn entity disambiguation along with syntax formulation in order to obtain the right semantic parse which is a more demanding task. This behaviour is present only in SPARQL, not SQUALL and Sparklis, explaining why w/o EL Mistral SPARQL outperforms w/o EL SQUALL.

In the case of Hybrid EL, it can be seen that it gives optimal performance across all entity-linking variations with the notable exception of Mistral SPARQL and Yi SPARQL. These exceptions can be explained by the reasoning with the following example. For the question: "What was Daniel Craig's first Bond movie?", here is the Label-based SPARQL query in case w/o EL case: `SELECT DISTINCT ?Q11424.8 WHERE ?Q11424.8 wdt:<instance of> wd:[Film] . ?Q11424.8 p:<part`

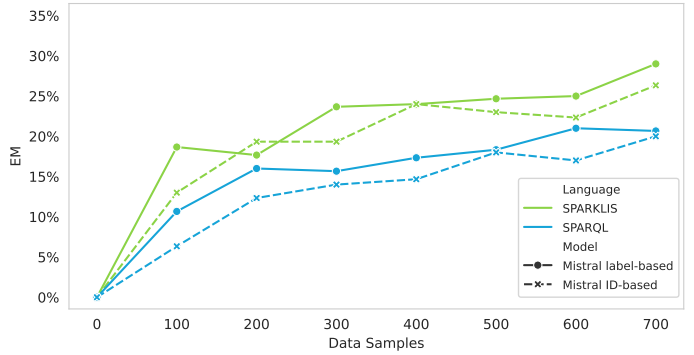
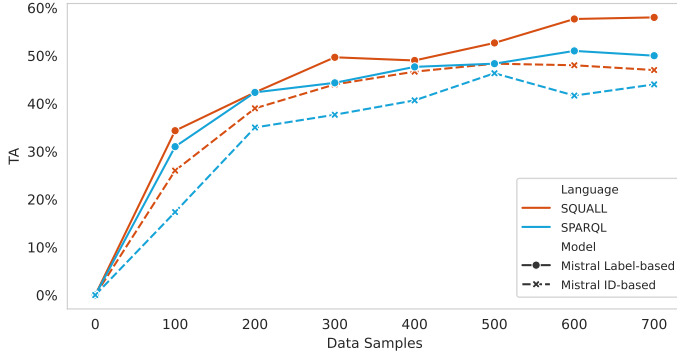


Figure 5: On the left Fig. 5a is an illustration of TA (y-axis) varying as a function of the size of the fine-tuning dataset for the Label-based and ID-based versions of SQUALL and SPARQL for Mistral. On the right Fig. 5b is an illustration of EM (y-axis) varying as a function of the size of the fine-tuning dataset for the Label-based and ID-based versions of Sparklis and SPARQL for Mistral.

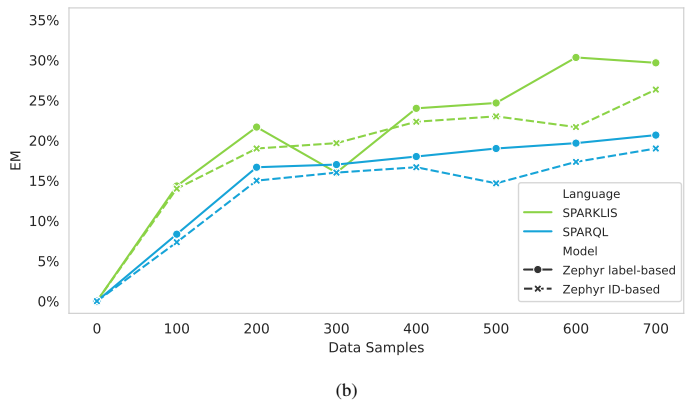
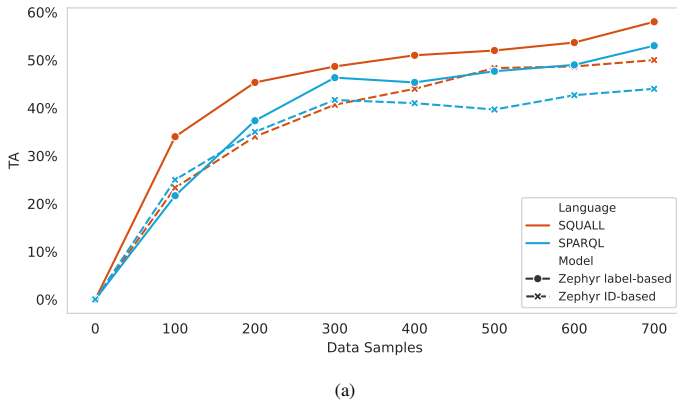


Figure 6: On the left Fig. 6a is an illustration of TA (y-axis) varying as a function of the size of the fine-tuning dataset for the Label-based and ID-based versions of SQUALL and SPARQL for Zephyr. On the right Fig.6b is an illustration of EM (y-axis) varying as a function of the size of the fine-tuning dataset for the Label-based and ID-based versions of Sparklis and SPARQL for Zephyr.

of the series> [ ps:<part of the series> wd:[James Bond] ] .  
 ?Q11424\_8 p:<cast member> [ ps:<cast member> wd:[Daniel Craig] ] .  
 ORDER BY ASC(?Q11424\_8) LIMIT 1.

Also, a query in case of Hybrid EL: SELECT DISTINCT  
 ?Q11424.8 WHERE ?Q11424.8 wdt:<instance of> wd:[film]  
 . ?Q11424.8 p:<publication date> [ ps:<publication date>  
 ?P577.45 ] . ?Q11424.8 p:<cast member> [ ps:<cast member>  
 wd:[Daniel Craig] ] . ORDER BY ASC(?P577.45) LIMIT 1.

It is seen that in the case of w/o EL, the model didn't have to perform any entity disambiguation. It simply learns to utilize the entity "James Bond" seen either in the training data or in its pretraining to generate a correct query. Whereas, in case of Hybrid EL the model bypasses over this entity, since it also has to learn to do entity disambiguation to arrive at an answer. Similar argument also holds true in case of Implicit EL as it has to understand to perform entity disambiguation before arriving at the correct query.

Overall the superiority of Hybrid EL is anticipated, as the best of both methods are combined i.e. Implicit EL and Explicit EL to provide the context of the entity into the prompt for enhancing entity disambiguation by the LLM and also mitigate the hallucinations via post-ad-hoc entity adjustment. Also, in

the case of Sparklis, Hybrid EL is at least on par, if not better than other entity-linking variations, though this might be an underestimate since we are only comparing the EM and not actual TA.

### 5.2. Effect of CNLs on KGQA Semantic Parsing Accuracy

This section deals with the effect of CNLs compared to query languages on the accuracy of KGQA semantic parsing. To show this, the results for SQUALL and SPARQL were compared using TA as a reference. It is observed that for all models, the ID-based version of SQUALL outperforms the ID-based version of SPARQL. The same is observed for the Label-based version of the languages. A similar trend is observed for the BLEU and METEOR metrics. There are, however, some exceptions (eg BLEU for Mistral ID-based SQUALL vs Mistral ID-based SPARQL) concerning the BLEU metric. Overall the TA is still consistently greater for SQUALL over SPARQL. As the TA is a direct measure of whether the semantics of an NL query is preserved during translation to a target query language, this can be interpreted as the models being better equipped to translate an NL query to SQUALL rather than SPARQL. We hypothesize that the relative naturalness of the syntax of SQUALL compared

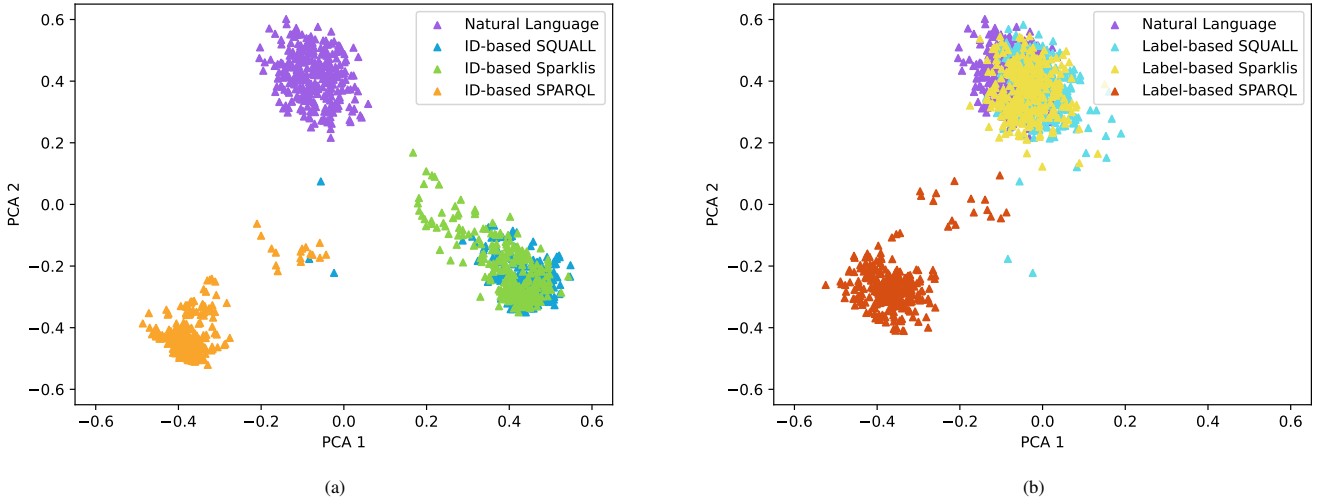


Figure 7: PCA visualisation of sequence embeddings of SQUALL, Sparklis and SPARQL for the Mintaka-1k test dataset. The figure on the left shows the clustering of the ID-based versions of SQUALL, Sparklis and SPARQL. The figure on the right shows the Label-based version of the same.

Table 6: Baseline comparison of GPT-3.5 Zero-Shot setting against Zephyr, Mistral for ID-based SPARQL and SQUALL

Model Name	Zero-Shot SPARQL	Fine-Tuned SPARQL	Zero-Shot SQUALL	Fine-Tuned SQUALL
GPT-3.5	7%	29%	0%	42%
Zephyr	0%	44%	0%	50%
Mistral	0%	44%	0%	50%

Table 7: The two tables measure the Latency in terms of average tokens per output query on the test dataset (avg\_tok/query) for both the top-performing models, Mistral and Zephyr-7b-beta. On the left, we have the Latency table of model Mistral on all versions of languages, i.e. SQUALL, Sparklis, SPARQL. On the right we have the Latency table of model Zephyr-7b-beta on all versions of languages, i.e; SQUALL, Sparklis, SPARQL

Model Name	Language	L-Type	Latency (avg_tok/query)
Mistral	SQUALL	Label-based	29.23
Mistral	SQUALL	ID-based	35.17
Mistral	Sparklis	Label-based	36.92
Mistral	Sparklis	ID-based	45.55
Mistral	SPARQL	Label-based	97.65
Mistral	SPARQL	ID-based	94.64

Model Name	Language	L-Type	Latency (avg_tok/query)
Zephyr-7B Beta	SQUALL	Label-based	29.47
Zephyr-7B Beta	SQUALL	ID-based	35.14
Zephyr-7B Beta	Sparklis	Label-based	35.42
Zephyr-7B Beta	Sparklis	ID-based	45.00
Zephyr-7B Beta	SPARQL	Label-based	105.67
Zephyr-7B Beta	SPARQL	ID-based	99.18

to SPARQL enables this, as the models already have a sufficient understanding of natural language due to their pre-training corpus. It can hence be concluded in regards to SQUALL and SPARQL that there is a positive effect on accuracy when using SQUALL as a CNL, especially for the smaller language models. In our study of Sparklis, EM was adopted as an alternative to TA for evaluating the performance. This choice is due to the design of the Sparklis queries, which are constructed interactively with a Sparklis tool Ferré, thereby lacking a direct translation to SPARQL. When considering EM, since there are multiple ways the semantics of a natural language question can be represented in the target language, it does not accurately represent the model’s understanding of the query’s semantics. In our experiments, Sparklis typically outperforms SPARQL in all cases with the exception of GPT2 and BLOOM. Although not

as strong a claim as with SQUALL, this implies that the models can represent the semantics of the NL question better in Sparklis and can formulate Sparklis better than SPARQL.

To further support this claim, we also compare the performance on our two best performing models (Zephyr-7b, Mistral-7b) over SPARQL, SQUALL with ID and label based variants. In this experiment, we note the performance over varying data samples as shown in the Figures 5a and 6a with TA being the evaluation metric. We observe that for the dataset size of 100, SPARQL outperforms SQUALL, however for all the other cases SQUALL outperforms SPARQL. This holds for both models. It can then be inferred that a larger fine-tuning dataset size would be required by SPARQL in order to obtain the same accuracy as SQUALL, conversely, a smaller amount of SQUALL fine-tuning data would achieve the same accuracy

as SPARQL. Similarly when we consider EM as the evaluation metric 5b and 6b, we observe for Mistral, except at 200 samples, both the versions of Sparklis outperform SPARQL. There is a smaller difference observed between the ID-based and Label-based versions of Sparklis rather than SPARQL. A similar observation can be made for Zephyr where, except at 100 samples, both Sparklis versions outperform SPARQL with a smaller difference for the Label-based version. We can explain this in light of the reasoning provided in Section 5.2. More precisely, EM is an underestimate on the hits@1 execution metric, as all exact matches are executable. Moreover, EM doesn't capture model's ability to preserve the semantics equivalent to Mintaka-1k ground truth Sparklis effectively. However, it is plausible, given what we observe with SQUALL, that an executable form of Sparklis would have a higher hits@1 when compared to SPARQL. This would be due to the presence of executable queries that preserve the semantics but do not match the Mintaka-1k gold standard exactly.

### 5.3. Effect of Naturality of Target Language on KGQA Semantic Parsing Accuracy

This section explores the effects of different versions of target languages on the accuracy of KGQA semantic parsing. Namely, the ID-based and Label-based versions of each language. To illustrate this the delta between the ID-based and Label-based versions of each language in the rightmost column of Table 4 and 5 is considered. A similar trend as before is observed, where the TA, BLEU, and METEOR metrics are consistently higher for the Label-based version of the language. Some exceptions are seen in GPT2 SQUALL and BLOOM SPARQL.

It is also observed that the Label-based version of SPARQL outperforms, or performs as well as, the ID-based version of SQUALL for all models except GPT2 and BLOOM in terms of TA. We hypothesize that this is due to the model's exposure to SPARQL in the pre-training data as opposed to limited or non-existent exposure to both SQUALL and Sparklis for the same, and the relative naturalness of Label-based SPARQL. The former can be illustrated by the zero-shot case results where it was generally observed that both the high-performing models, such as Mistral and Zephyr, generate different formulations of SPARQL, whereas the models were unable to generate SQUALL or Sparklis. On the contrary, generated random SQL or SPARQL surface forms.

Figure 7a and 7b compare the embeddings of NL sentences with respectively ID-based versions (left) and Label-based versions (right). The ID-based versions are grouped in two clusters separate from the NL cluster, the two CNLs at the bottom right and SPARQL at the bottom left. For the Label-based versions, the CNLs are also in a separate cluster than SPARQL but this time, the CNL cluster mostly coincides with the NL cluster. This reveals several things about how the model sees the different versions. First, only versions that combine a natural syntax and a natural lexicon (entities and relations), i.e. Label-based versions of CNL, are seen in the same cluster as natural language. Second, the CNLs are always found similar to each other, and clearly distinct from SPARQL. Further, the distance from the centroid of the NL cluster to ID-based SQUALL

and the distance from the centroid of the NL cluster to that of Label-based SPARQL was calculated. The distances obtained were 0.84 and 0.74 respectively. From these distances, it is evident that the sentence embedding cluster of Natural Language is closer to the cluster of Label-based SPARQL embeddings than the cluster of ID-based SQUALL embeddings. Even though the ID-based version of SQUALL, syntactically resembles Natural language more than the Label-based version of SPARQL, it does have a lot of non-natural tokens due to the presence of PIDs and QIDs in the query construction. The Label-based version of SPARQL, on the other hand, does have a significant amount of natural language tokens but does not possess natural language syntax. Overall it appears that the presence of natural language tokens (i.e Wikidata Labels) in Label-based SPARQL contributes more to its naturalness than the natural language syntax of ID-based SQUALL. The presence of SPARQL in the pre-training data further enhances the model's ability to construct syntactically correct Label-based SPARQL queries better than ID-based SQUALL queries.

### 5.4. Sample Efficiency of CNLs and SPARQL

In this section, the sample efficiency of CNLs and SPARQL are studied, beginning with an analysis of baseline comparison in the Table 6, where a comparison between the zero-shot setting of GPT3.5 against our two best-performing models (Zephyr-7b-beta, Mistral-v0.2) is made. From the table, it is observed that GPT-3.5 ID-based SPARQL has a TA of 7%, which is notably higher compared to other models, which have a TA of 0%. This 7% performance difference can be attributed to the extensive pre-training data (Wikidata, DBpedia) of GPT3.5 compared to the other models. However, even after fine-tuning, the performance of ID-based SPARQL is lower compared to ID-based fine-tuned SQUALL. For example, fine-tuned SPARQL achieves only 29%, whereas fine-tuned SQUALL reaches 42% in case of GPT-3.5, further demonstrating that the CNL SQUALL is learnt faster (from scratch) compared to SPARQL with the same amount of training data. Additionally, the sample efficiency of CNLs and SPARQL, is demonstrated with the plots constructed in Figures 5a and 6a. Again, the best-performing models are considered, specifically Mistral-v0.2 and Zephyr-7b-beta, and the size of the dataset required for fine-tuning is varied. This is performed for all language variants. These models are then evaluated on the same test dataset of 300 questions.

First, the difference between SQUALL and SPARQL is considered for comparison. For the ID-based and Label-based versions of Mistral-v0.2, it is observed that for the dataset size of 100, SPARQL outperforms SQUALL, however for all the other cases SQUALL outperforms SPARQL. In the case of Zephyr, Label-based SQUALL appears to outperform Label-based SPARQL from the very beginning whereas the ID-based SQUALL outperforms ID-based SPARQL from a dataset size of 300. A more direct comparison of sample efficiency can be observed by looking at different data points in Figures 5a and 6a between languages. Specifically, Mistral-v0.2, ID based SPARQL has a peak performance of in TA of 0.46 at full training set of 500 examples. In contrast, Label-based

Table 8: Error categories based on hierarchical approach observed on the test set of 300 examples w.r.t TA for Zephyr-7b-beta fine-tuned on 700 examples.

Error Category	Category Definition	#SQUALL	#SPARQL
(1) Entity linking	at least one entity required for building a correct query not found by ReFinED	21	35
(2) Entity selection	required entities provided by the ReFinED, but at least one selected incorrectly	6	7
(3) Wrong Relations Generation	at least one relation generated in the predicted truth is not present in the gold standard	21	30
(4) Hallucinated Entities	the entities are hallucinated by the LLM and not provided by ReFinEd in the context.	12	21
(5) Syntactically incorrect query	For SQUALL query cannot be parsed by the SQUALL $\rightarrow$ SPARQL converter / For SPARQL the query didn't execute on Wikidata Query service	17	-
(6) Semantically incorrect query	query does not reflect meaning of the input question	19	16
(7) Query not matching KG structure	relationships used in the query don't match the KG structure	22	29
(8) Implicit entities	the query requires an implicit entity (e.g. "Game of Thrones book" = "A song of ice and Fire") such that ReFinED cannot pick it up and the LLM misses it as well	2	6
(9) Hits@1 automation error	automation of Hits@1 results in some correctly executed queries being labelled as incorrect	7	5
(10) LLM Completion Error	LLM is unable to generate a complete query in case of SPARQL	-	20

Table 9: Tokens generated by Zephyr-7B, for each language with label, ID variants. **Note:** irrelevant tokens are truncated with (...) for readability

Language L-Type	Tokenized Example
SQUALL Label-based	'which'... '<', 'position', 'held', '>' ... , '<', 'height', '>?'
SQUALL ID-based	'which'... , '<', 'P', '3', '9', '>' ... , '<', 'P', '2', '0', '4', '8', '>?'
Sparklis Label-based	'give' ... '[' , 'human', ']' ... , '<', 'height', '>'
Sparklis ID-based	'give' ... , '<', 'Q', '5', '>' ... , '<', 'P', '2', '0', '4', '8', '>'
SPARQL Label-based	'SE', 'LECT', 'D', 'IST', 'IN', 'CT' ... , 'w', 'd', ':', '[', 'Fil', 'm', ']' ... , '<', 'part', 'of', 'the', 'series', '>' ... , 'w', 'd', ':', '[', 'J', 'on', 'F', 'av', 're', 'au', ']', ']', ':', ']', ']', ']'
SPARQL ID-based	'SE', 'LECT', 'D', 'IST', 'IN', 'CT' ... , 'w', 'd', ':', 'Q', '1', '1', '4', '2', '4', 'P', '1', '7', '9' ... , 'w', 'd', ':', 'Q', '2', '9', '5', '9', '6', '4', ']', ']', ']', ']', ']'

SQUALL achieves similar performance at approximately 230 samples, requiring approximately 54% less training data to match SPARQL's peak performance. A very similar trend can also be observed in case of Zephyr-7b-beta, where ID-based SPARQL achieves a maximum TA of 0.44 at 700 samples, while the Label-based SQUALL achieves this or even surpasses it (approximately 0.45 TA) at just 200 examples, requiring approximately 71% less training data compared to SQUALL. It can therefore be inferred that a larger fine-tuning dataset size would be required by SPARQL in order to obtain the same accuracy as SQUALL, conversely, a smaller amount of SQUALL fine-tuning data would achieve the same accuracy as that of SPARQL.

Second, the difference between Sparklis and SPARQL is compared using EM in Figures 6b and 7b. In case of Mistral, it is observed that for all dataset sizes, both versions of Sparklis outperform SPARQL. A similar observation can be made for Zephyr where, except at 300 samples, both Sparklis versions outperform SPARQL with a smaller difference for the Label-based version. This can be explained by the reasoning provided in Section 5.2. To recap, EM is an underestimate or a subset of the TA execution metric, as all exact matches are executable. Moreover, EM doesn't effectively capture the model's ability to retain the semantics of the original natural language query in Sparklis. However, it is plausible, given what we observe with SQUALL, that an executable form of Sparklis would have a higher TA compared to SPARQL.

Table 7 provides latency measurements for both Mistral and Zephyr-7b models across all languages over different types (ID/Label based). Latency here is measured per output query on the test dataset. Generally it is seen that overall, both the CNLs i.e. SQUALL and Sparklis, have a lower latency than SPARQL,

with the lowest latency being in the case of SQUALL. It is further observed in the case of SQUALL and Sparklis, that the Label-based version of the languages appears to have a lower latency than the ID-based versions. This, however, is not the case for SPARQL. This can be explained by considering Table 9, the example of the Label-based and ID-based versions of Sparklis. The entity <Q5> requires 4 tokens i.e., ('<', 'Q', '5', '>') to represent, however [human] is represented using 3 tokens i.e., ('<', 'human', '>'). Similarly, <P2048> requires 7 tokens i.e., ('<', 'P', '2', '0', '4', '8', '>') while <height> requires only three i.e., ('<', 'height', '>'). However, in the case of SPARQL, as a result of introducing the tokens < > and [ ] only in Label-based SPARQL, the tokenizer tends to categorize them as an individual token. This in turn increases the number of tokens on average. For example consider the Label-based and ID-based version of SPARQL, the text <part of the series> is split into six tokens i.e., ('<', 'part', 'of', 'the', 'series', '>') whereas P179 is split in four i.e., ('P', '1', '7', '9'). On the other hand, this increase in the average number of tokens is not observed in the case of Sparklis and SQUALL (see Table 9), as both the ID-based and Label-based versions of languages have < > and [ ] tokens in their formulation. The overall performance increase can be attributed to the design of the CNLs, specifically, the compactness and naturality of the syntax as opposed to that of query-based language. Furthermore, our experiments show that CNLs are more sample efficient.

## 6. Error Analysis

We performed an error analysis for two cases, namely, the Label-based SQUALL and ID-based SPARQL queries generated by Zephyr-7b-beta, since it is the best performing model as

Table 10: Examples for correct (first two rows) and incorrect parses of Zephyr-7b-beta for Label-based-SQUALL. For the incorrect parses, we show an example for each category.

Input	Generated SQUALL Query	Comment
What was the first movie that featured 50 Cent?	which [film] has <cast member> [50 Cent] and has the lowest <publication date>?	successfully detects that 50 Cent as a cast member and "lowest" refers to first publication date
Which governors of Michigan since 2003 have not been Democrats?	which [human] has <position held> at a <start time> whose year is $\geq$ 2003 [Governor of Michigan] and has not <member of political party> [Democratic Party]?	successfully builds a compositional query combining two criteria
What was the first James Bond movie?	which [film] has <based on> [James Bond 3] and has the lowest <publication date>?	entity linking error: linked the book series instead of the film series (James Bond 3 is unambiguated entity)
What is the first book in the James Bond series?	which [Literary Work] has <part of the series> at <series ordinal> "1" [James Bond]?	entity selection error: picked the [James Bond] Film instead of [James Bond 3] book series.
Which movie did John Williams last compose for?	which [Film] has <score composer> [John Williams] and has the highest <publication date>?	Hallucinated Relation: generated <score composer> as a relation which is hallucinated
How many of Booth Tarkington's books won a Pulitzer Prize?	how many [Literary Work] has <author> [Edwin Arlington Robinson] and has a <award received> that has <instance of> [Pulitzer Prize]?	syntactically incorrect since there is "a" present after the "has", It should just be "has <award received>"
What is Al Pacinos full first name?	what is a <given name> at <series ordinal> 1 of [Al Pacino]?	semantically incorrect query asking for 1st name assuming it to be a part of Al Pacino series (movies or books)
When did the Achaemenid Empire end?	what is the year of a <end time> of [Achaemenid Empire]?	query not matching KG structure, which uses a property <end time> instead of <dissolved or abolished>
When was the last time Guillermo del Toro won an Academy Award for Best Director?	what is the highest thing at which <point in time> [Guillermo Del Toro] has <award received> [Academy Award For Best Director]?	our script for automation was unable to handle this case as the ground truth was "2018" and the query returned "1 January 2018"
When was Obama first elected?	what is a <start time> of [2008 United States Presidential Election]?	The model here hallucinated the entity [2008 United States Presidential Election]
What British queen never had children?	which [human] has <position held> [monarch of the United Kingdom of Great Britain and Ireland] and has not a <child>?	an implicit entity [monarch of England] was required and was not given by ReFinEd.

can be seen from Table 4. After error collection, and analysis over them, several patterns were visible which led to identifying error categories. In general, it is possible for a query to have multiple causes for failure. Therefore, a "categorical" approach (see Table 8) was followed in categorising the error.

With this approach, a single root cause for the error was established. First, we determine whether the entity linker provides the required entities to build the query. In our framework, all entities above a given threshold are fed to the LLM. The LLM then decides which one to select. Thus the entity disambiguation is not completely resolved by the entity linker. It is, however, required that the correct entities are among those ones provided by the entity linker. If this is not the case then it is categorised as an entity linking error. If all the entities are provided by the entity linker but the LLM chooses the wrong one, this is categorised as an entity selection error. If the correct entities are selected but the LLM generates a wrong relation, this is termed incorrect relation generation. If correct relations are selected but the LLM generates a wrong entity it is termed as hallucinated entities. The entity linking approach can be an arbitrary number, however, it limits the ceiling TA performance of the LLM to 88% with the given threshold.

In the case where entities and relations are selected correctly, we then investigate whether there are syntactic or semantic er-

rors. Although syntactic errors are generally not observed with LLMs, several of them were found owing to the variable structure of SQUALL which allows multiple expressions to map to the same query. As the training data for the LLM is limited, it has not completely resolved which of those variations are permitted. In the case of SPARQL, the queries threw an execution error on Wikidata. In the case of semantic errors, the query generated by the LLM has not captured the intent of the question.

We further identify several other categories that result in an error and these are 7) Wikidata has a different KG structure from that chosen by the LLM. This happens due to the model not having direct access to the actual structure of the KG, but still tries to generate a plausible path in Wikidata. 8) The implicit entities which are not picked up by ReFinEd. 9) Our automation of the Hits@1 execution missed a few correct queries, for instance, a mismatch between a QID and its label is treated as an incorrect execution 10) The query generated by the LLM is incomplete, interestingly this was observed only in case of SPARQL and not incase of SQUALL or Sparklis as they are more compact and natural compared to SPARQL.

### 6.1. CNL's resilience vs SPARQL brittleness

The errors can be further classified as either *intrinsic*-originating from the LLM's generative output or *extrinsic* error-

Table 11: Examples for correct (first two rows) and incorrect parses of Mistral for ID-based-SPARQL. For the incorrect parses, we show an example for each category.

Input	Generated SPARQL Query	Comment
Who was the director of Scarface?	SELECT DISTINCT ?P57 WHERE { wd:Q47075 p:P57 [ ps:P57 ?P57 ] . } LIMIT 200	successfully uses the right relation P57 and entity Q47075 to construct the query
Who was not British in Katrina and the Waves?	SELECT DISTINCT ?P463 WHERE { ?P463 p:P463 [ ps:P463 wd:Q934582 ] . FILTER ( NOT EXISTS { ?P463 p:P27 [ ps:P27 wd:Q145 ] . } ) } LIMIT 200	successfully builds a compositional query combining two criteria
When did the NBA 2K series first come out?	SELECT DISTINCT ?P577 WHERE { wd:Q512849 p:P577 [ ps:P577 ?P577 ] . } LIMIT 1	entity linking error: refined retrieves the incorrect entity Q512849 instead of Q2073660
Did Twilight come out before 1995?	SELECT DISTINCT ?P577 ?expr WHERE { wd:Q748194 p:P577 [ ps:P577 ?P577 ] . BIND (year(?P577) AS ?expr) } LIMIT 200	entity selection error: picked the Q748194 1998 film directed by Robert Benton instead of Q160071 2008 film directed by Catherine Hardwicke
Where do the Yankees play?	SELECT DISTINCT ?P38 WHERE { wd:Q213417 p:P38 [ ps:P38 ?P38 ] . }	Hallucinated Relation: generated P38(currency) as a relation which is hallucinated instead of P115(home venue)
What is the third Led Zeppelin album called?	SELECT DISTINCT ?Q482994 WHERE { ?Q482994 wd:P31 wd:Q482994 . ?Q482994 p:P175 [ ps:P175 wd:Q2331 ; n1:P1545 "3" ] . } LIMIT 200	semantically incorrect query: query incorrectly uses series ordinal 3 to filter for a third item, instead of retrieving the specific performer wd:Q2331, and ordering by publication date
In what year did Ellen de Generes and Portia de Rossi get married?	SELECT DISTINCT ?expr WHERE { wd:Q483325 p:P26 [ ps:P26 ?P26 ] . wd:Q215366 p:P26 [ ps:P26 ?P26 ] . BIND (year(?P26) AS ?expr) }	query not matching KG structure, query does not consider P580, while it is expected to use a qualifier (ex: n1:P580 ?P580 ; ps:P26 wd:Q215366) to refine the search for point in time
When was Obama first elected?	SELECT DISTINCT ?expr WHERE { wd:Q76 p:P39 [ n1:P580 ?P580 ; ps:P39 wd:Q11696 ] . BIND (year(?P580) AS ?expr) }	our script for automation was unable to handle this case as the ground truth was "20 January 2000" and the model returned "2000"
Fenway Park is home to what MLB team?	SELECT DISTINCT ?Q17156793 WHERE { ?Q17156793 wd:P31 wd:Q17156793 . ?Q17156793 p:P118 [ ps:P118 wd:Q49136 ] . }	the model hallucinated the entity wd:Q17156793 (American football team)
What British queen never had children?	SELECT DISTINCT ?Q5 WHERE { ?Q5 wd:P31 wd:Q5 . ?Q5 p:P39 [ ps:P39 wd:Q111722535 ] . ?Q5 p:P27 [ ps:P27 wd:Q145 ] . } LIMIT 1 ?	an implicit entity Q18810062 (monarch of England) was required and was not given by ReFined.
How many universities are located in Oxford?	SELECT DISTINCT COUNT DISTINCT ?q875538 as ?number_of_749 where { ?q875538 wd:P31 wd:Q875538 . ?q875538 p:P155 [ ps:P155 wd:Q	LLM completion error: the model fails to completely generate the query

originating outside the LLM’s control. As shown in Table 8, categories 1, 7, 8, and 9 are of extrinsic type and the categories 2, 3, 4, 5, 6, and 10 are of intrinsic type. Critical investigation of these two types give some insight on how the EL bottlenecks are handled between the two languages SPARQL and SQUALL.

Firstly, SPARQL recorded 35 Entity Linking Failures - Category 1, significantly higher than SQUALL’s errors which is 12, despite the initial entity candidates being identical for both target languages. This demonstrates the brittleness of SPARQL; for e.g. a missing single digit in the QID of the entity (wd:Q76 instead of wd:Q75) or a keyword (e.g.WHERE) results in complete failure. Its rigid syntax is highly susceptible to complete failure when a component like EL misses a required entity, since post-adhoc correction is less effective in case of QIDs as opposed to Labels. On the contrary, SQUALL due to its structural compactness and proximity to natural language exhibit greater semantic resilience, allowing the LLM to more successfully infer intent due to their pre-training. This allows the LLM to bypass the immediate syntactic consequences of EL bottleneck and still generate a correct query with noisy or missing EL context using the post-adhoc correction step with Labels.

Second, the complexity in the structure of SPARQL magnifies the intrinsic LLM failures. For example, in LLM comple-

tion error cases (category 10) SPARQL registers 20 errors compared to SQUALL which registers 0, reinforcing the idea that a small and compact CNL (on average 29.23-29.47 for SQUALL) helps in reducing truncation inherent to the SPARQL (on average 94.64-99.18) queries, refer Table 7. Additionally, SPARQL generates 21 hallucinated entities (category 4) compared to SQUALL which has 12, reinforcing the idea that a small, compact and naturalness aids in generating a better query.

## 6.2. Limits of Static Threshold:

Raising the Threshold: would improve the precision and reduce the entity selection errors (category 2) by removing the low confidence candidates and reducing the noise for selecting the entity by the LLM. However, this would dramatically reduce the recall of the EL and increase the errors in Entity linking errors- category 1 by filtering out correct entities which have low confidence during retrieval using ReFinED. Given that category 1 (21–35 errors) is already significantly larger than category 2 (6–7 errors), raising the threshold would likely lower the TA and degrade overall performance.

Lowering the Threshold: would increase the recall of the ReFinED, potentially decreasing the EL errors- category 1. However, lowering the threshold would drastically flood the LLM’s context window with a large number of low confidence enti-

ties. For models like Mistral or Zephyr this could overwhelm the disambiguation capacity and hence potentially increase errors in other categories like Entity selection errors- category 2 or Hallucinated entities error- category 4. This would likely reduce the TA of the model.

In conclusion, a simple shift in of static threshold would not be enough to shift the ceiling of EL as it is constrained by precision-recall trade-off. However, part of this trade-off is handled by the post-adhoc correction in the Hybrid EL approach. An optimal strategy would be to have a dynamic or adaptive threshold which remains part of the future work.

In Tables [10, 11], we show instances of correct and incorrect parsing for both SQUALL and SPARQL. It was observed that the best-performing LLM Zephyr-7B Beta was able to perform a correct semantic parsing for the following cases:

- where the entities involved were not the highest ranked according to ReFinEd,
- where the relations were not observed in the training data,
- where ReFinED did not return all entities related to the input query.

It is also observed that even though our training dataset’s size has increased compared to Lehmann et al. (2023), the LLM still has trouble generalising to unseen topics. We shall note that, normally semantic parsers are trained on much more training data than our study Unger et al. (2012), Trivedi et al. (2017), Yih et al. (2014).

We, however, still recommend including training data that covers a broad range of topics enabling the LLM to observe several KG structures to learn the relations and entities.

## 7. Limitation and Future Works

The design of CNLs aims at a trade-off between naturality and simplicity. Indeed, a more natural CNL requires a more complex grammar. The two factors are important for human use but also for training LLMs. A more natural CNL will be closer to natural language, hence easier to learn by the LLM. However, a more complex CNL will require larger training samples to enable the LLM to catch all language constructs. This trade-off can be a limiting factor in the performance of the translation from NL to CNL by the LLM. With this trade-off in mind, future work could concentrate on NL constructs that are missing in the CNL yet common in end-user questions. For example, comparative and yes/no questions in Sparklis; or questions involving proportions in both SQUALL and Sparklis. A not-so-natural aspect of our CNL forms is the rendering of relations from their labels. Those labels can be nouns or verbs (or other syntactic categories) but are always used as nouns in the CNL forms because the linguistic information about the labels is missing. Given a target dataset (e.g., Wikidata), a lexicon of relations could be defined, mapping PIDs to both a surface form and a syntactic category. LLM may even be used for the task. That lexicon would then be used to generate more natural

CNL forms, which could improve the learning of the CNL by the LLM.

A rich and natural CNL inevitably exhibits syntactical ambiguities, i.e. a CNL query may have different interpretations that translate to different SPARQL queries. In practice, such ambiguities are resolved in CNLs by following rules similar to operator priorities in arithmetic expressions. These disambiguation rules are based on the syntax whereas, ideally, disambiguation should be based on semantics and available data. As disambiguation can be seen as a choice between different semantic interpretations, an interesting perspective would be to involve the LLM, possibly in combination with checking the actual data.

The current CNL-based approaches are only limited SQUALL and Sparklis, but extending it to other languages Cypher, Neo4j’s query language, offers a more natural syntax than SPARQL while maintaining formal query capabilities. Our study’s EL approach could also be enhanced by incorporating techniques like BLINKLedell Wu (2020) (Bi-encoder and cross-encoder architecture) or more recent retrieval-augmented entity linking methods that leverage contextual information from the knowledge graph itself. Additional limitations include the challenge of multilingual support, as the current approach was only tested with English queries despite Wikidata being multilingual, creating a significant barrier for non-English users. The computational efficiency of the pipeline remains a future work, as the hybrid entity linking approach introduces additional processing steps that may impact real-time application performance. In future research, we plan to implement a translation layer to convert Sparklis queries into executable SPARQL. This will enable a more direct comparison of target languages using the Translation Accuracy (TA) metric. Finally, the evaluation of our approach is further planned to be extended for complex temporal queries and questions requiring reasoning across multiple hops in the knowledge graph. Research into temporal knowledge graph embeddingsZhang et al. (2024) highlights the importance and complexity of capturing time-evolving information. This, in turn, highlights the necessity of enhancing CNL-based semantic parsing for time-dependent knowledge.

## 8. Conclusion

In conclusion, our work introduces a generalizable approach, termed *BridgeKGLM*, which leverages Controlled Natural Language (CNL) as a medium to bridge the gap between the intuitive expressiveness of natural language and the structured precision of formal query languages for LLM and Knowledge Graphs (KGs). Through the innovative use of CNL for semantic parsing, we have demonstrated both the feasibility and effectiveness of this approach. Our experimental validations provide evidence for these claims. Beside the *BridgeKGLM* approach, we also developed and released a new dataset, *Mintaka-1k*.

Our findings suggest that *BridgeKGLM* enhances the accessibility of KGs, enabling more intuitive and accurate query formulations without the steep learning curve associated with traditional formal query languages. This research marks an impor-

tant step toward making complex knowledge databases more accessible to a broader audience, facilitating the integration of LLMs in the process.

## 9. Acknowledgments

This work was supported by the German Federal Ministry of Education and Research (BMBF, SCADS22B) and the Saxon State Ministry for Science, Culture and Tourism (SMWK) by funding the competence center for Big Data and AI "ScaDS.AI Dresden/Leipzig".

## References

- Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Alteschmidt, J., Altman, S., Anadkat, S., et al., 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774 .
- Angles, R., Gutierrez, C., 2008. Survey of graph database models. *ACM Comput. Surv.* 40. URL: <https://doi.org/10.1145/1322432.1322433>, doi:10.1145/1322432.1322433.
- Ayoola, T., Tyagi, S., Fisher, J., Christodoulopoulos, C., Pierleoni, A., Al, A.A., 2022. Refined: An efficient zero-shot-capable approach to end-to-end entity linking. *NAACL-HLT 2022* , 209.
- Baek, J., Aji, A.F., Saffari, A., 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering, in: Dalvi Mishra, B., Durrett, G., Jansen, P., Neves Ribeiro, D., Wei, J. (Eds.), *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)*, Association for Computational Linguistics, Toronto, Canada. pp. 78–106. URL: <https://aclanthology.org/2023.nlrse-1.7>, doi:10.18653/v1/2023.nlrse-1.7.
- Banerjee, D., Awale, S., Usbeck, R., Biemann, C., 2023. Dbp-quad: A question answering dataset over the dbp scholarly knowledge graph. arXiv:2303.13351.
- Banerjee, S., Lavie, A., 2005. 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.
- Bellomari, L., Sallinger, E., Vahdati, S., 2020. Knowledge graphs: the layered perspective. *Knowledge Graphs and Big Data Processing 12072*, 20.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33, 1877–1901.
- Chakraborty, N., Lukovnikov, D., Maheshwari, G., Trivedi, P., Lehmann, J., Fischer, A., 2021. Introduction to neural network-based question answering over knowledge graphs. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 11, e1389.
- Child, R., Gray, S., Radford, A., Sutskever, I., 2019. Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509 .
- Dubey, M., Banerjee, D., Abdelkawi, A., Lehmann, J., 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, October 26–30, 2019, *Proceedings, Part II* 18, Springer. pp. 69–78.
- Ferré, S., 2014. SQUALL: The expressiveness of SPARQL 1.1 made available as a controlled natural language. *Data & Knowledge Engineering* 94, 163–188.
- Ferré, S., 2017. Sparklis: An expressive query builder for sparql endpoints with guidance in natural language. *Semantic Web* 8, 405–418.
- Ferré, S., . Sparklis. <http://www.irisa.fr/LIS/ferre/sparklis/>. <http://www.irisa.fr/LIS/ferre/sparklis/> [Accessed 15-02-2024].
- Fu, B., Qiu, Y., Tang, C., Li, Y., Yu, H., Sun, J., 2020. A survey on complex question answering over knowledge base: Recent advances and challenges. arXiv preprint arXiv:2007.13069 .
- Gu, Y., Pahuja, V., Cheng, G., Su, Y., 2022. Knowledge base question answering: A semantic parsing perspective. arXiv:2209.04994.
- Hinton, G., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 .
- Jaradeh, M.Y., Stocker, M., Auer, S., 2020. Question answering on scholarly knowledge graphs. arXiv:2006.01527.
- Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., Casas, D.d.l., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., et al., 2023. Mistral 7b. arXiv preprint arXiv:2310.06825 .
- Kitaev, N., Kaiser, L., Levskaya, A., 2020. Reformer: The efficient transformer. arXiv preprint arXiv:2001.04451 .
- Kuhn, T., 2013. A survey and classification of controlled natural languages. *Computational Linguistics* .
- Ledell Wu, Fabio Petroni, M.J.S.R.L.Z., 2020. Zero-shot entity linking with dense entity retrieval, in: *EMNLP*.
- Lehmann, J., Gattogi, P., Bhandiwad, D., Ferré, S., Vahdati, S., 2023. Language models as controlled natural language semantic parsers for knowledge graph question answering, in: *European Conference on Artificial Intelligence (ECAI)*, IOS Press. pp. 1348–1356.
- Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C., 2015a. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal* 6, 167–195. Outstanding Paper Award (Best 2014 SWJ Paper).
- Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al., 2015b. Dbpedia—a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web* 6, 167–195.
- Li, B., Fan, Y., Sataer, Y., Gao, Z., Gui, Y., 2022. Improving semantic dependency parsing with higher-order information encoded by graph neural networks. *Applied Sciences* 12. URL: <https://www.mdpi.com/2076-3417/12/8/4089>, doi:10.3390/app12084089.
- Lin, C.Y., Hovy, E., 2003. 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.
- Liu, Z., Li, J., Huang, Y., Cui, N., Pei, L., 2025. Knowledge-based natural answer generation via effective graph learning. *Knowledge-Based Systems* 316, 113288.
- Meng, X., Dai, W., Wang, Y., Wang, B., Wu, Z., Jiang, X., Liu, Q., 2023. Lexicon-injected semantic parsing for task-oriented dialog, in: *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. doi:10.1109/ICASSP49357.2023.10095358.
- Montague, R., 1970. Universal grammar. *Theoria* 36, 373–398.
- Nie, L., Cao, S., Shi, J., Sun, J., Tian, Q., Hou, L., Li, J., Zhai, J., 2022. 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.
- Papineni, K., Roukos, S., Ward, T., Zhu, W.J., 2002. 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.
- Peng, B., Galley, M., He, P., Cheng, H., Xie, Y., Hu, Y., Huang, Q., Liden, L., Yu, Z., Chen, W., et al., 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. arXiv preprint arXiv:2302.12813 .
- Riloff, E., Thelen, M., 2000. A rule-based question answering system for reading comprehension tests, in: *ANLP-NAACL 2000 workshop: reading comprehension tests as evaluation for computer-based language understanding systems*.
- Rongali, S., Arkoudas, K., Rubino, M., Hamza, W., 2022. Training naturalized semantic parsers with very little data. arXiv preprint arXiv:2204.14243 .
- Rongali, S., Soldaini, L., Monti, E., Hamza, W., 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing, in: *Proceedings of The Web Conference 2020*, pp. 2962–2968.
- Rony, M.R.A.H., Chaudhuri, D., Usbeck, R., Lehmann, J., 2022a. Tree-kgqa: an unsupervised approach for question answering over knowledge graphs. *IEEE Access* 10, 50467–50478.
- Rony, M.R.A.H., Kumar, U., Teucher, R., Kovriguina, L., Lehmann, J., 2022b. Sgpt: A generative approach for sparql query generation from natural language questions. *IEEE Access* 10, 70712–70723.
- Scao, T.L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A.S., Yvon, F., Gallé, M., et al., 2022. Bloom: A 176b-parameter

- open-access multilingual language model. arXiv preprint arXiv:2211.05100 .
- Schneider, P., Klettner, M., Jokinen, K., Simperl, E., Matthes, F., 2024. Evaluating large language models in semantic parsing for conversational question answering over knowledge graphs. arXiv:2401.01711.
- See, A., Liu, P.J., Manning, C.D., 2017. Get to the point: Summarization with pointer-generator networks. arXiv preprint arXiv:1704.04368 .
- Sen, P., Aji, A.F., Saffari, A., 2022. 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.
- Sen, P., Mavadia, S., Saffari, A., 2023. Knowledge graph-augmented language models for complex question answering, in: Dalvi Mishra, B., Durrett, G., Jansen, P., Neves Ribeiro, D., Wei, J. (Eds.), Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE), Association for Computational Linguistics, Toronto, Canada. pp. 1–8. URL: <https://aclanthology.org/2023.nlrse-1.1>, doi:10.18653/v1/2023.nlrse-1.1.
- Shi, J., Cao, S., Pan, L., Xiang, Y., Hou, L., Li, J., Zhang, H., He, B., 2020. Kqa pro: A large-scale dataset with interpretable programs and accurate sparqls for complex question answering over knowledge base. arXiv preprint arXiv:2007.03875 .
- Shin, R., Lin, C., Thomson, S., Chen Jr, C., Roy, S., Platanios, E.A., Pauls, A., Klein, D., Eisner, J., Van Durme, B., 2021. Constrained language models yield few-shot semantic parsers, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 7699–7715.
- Shin, R., Van Durme, B., 2021. Few-shot semantic parsing with language models trained on code. arXiv preprint arXiv:2112.08696 .
- Song, L., Gildea, D., Zhang, Y., Wang, Z., Su, J., 2019. Semantic neural machine translation using amr. Transactions of the Association for Computational Linguistics 7, 19–31.
- Sun, K., Xu, Y.E., Zha, H., Liu, Y., Dong, X.L., 2023. Head-to-tail: How knowledgeable are large language models (llm)? aka will llms replace knowledge graphs? arXiv preprint arXiv:2308.10168 .
- Taffa, T.A., Usbeck, R., 2023. Leveraging llms in scholarly knowledge graph question answering. arXiv:2311.09841.
- Trivedi, P., Maheshwari, G., Dubey, M., Lehmann, J., 2017. 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, Springer. pp. 210–218.
- Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., von Werra, L., Fourrier, C., Habib, N., Sarrazin, N., Sanseviero, O., Rush, A.M., Wolf, T., 2023. Zephyr: Direct distillation of lm alignment. arXiv:2310.16944.
- Unger, C., Bühmann, L., Lehmann, J., Ngonga Ngomo, A.C., Gerber, D., Cimiano, P., 2012. Template-based question answering over rdf data, in: Proceedings of the 21st international conference on World Wide Web, pp. 639–648.
- Voorhees, E.M., et al., 1999. The trec-8 question answering track report., in: Trec, pp. 77–82.
- Vrandečić, D., Krötzsch, M., 2014. Wikidata: a free collaborative knowledge-base. Communications of the ACM 57, 78–85.
- Wang, Y., Berant, J., Liang, P., 2015. 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.
- Wu, Y., Hu, N., Bi, S., Qi, G., Ren, J., Xie, A., Song, W., 2023. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. arXiv:2309.11206.
- Xuan, C., 2020. Improving sequence-to-sequence semantic parser for task oriented dialog, in: Bogin, B., Iyer, S., Lin, V., Radev, D., Suhr, A., Panupong, Xiong, C., Yin, P., Yu, T., Zhang, R., Zhong, V. (Eds.), Proceedings of the First Workshop on Interactive and Executable Semantic Parsing, Association for Computational Linguistics, Online. pp. 18–22. URL: <https://aclanthology.org/2020.intexsempar-1.3>, doi:10.18653/v1/2020.intexsempar-1.3.
- Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., Cao, Y., 2022. React: Synergizing reasoning and acting in language models. arXiv preprint arXiv:2210.03629 .
- Yasunaga, M., Ren, H., Bosselut, A., Liang, P., Leskovec, J., 2021. 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.
- Yih, S.W.t., Chang, M.W., He, X., Gao, J., 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base, in: Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP.
- Yih, W.t., He, X., Meek, C., 2014. Semantic parsing for single-relation question answering, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 643–648.
- Zhang, Y., Kong, X., Shen, Z., Li, J., Yi, Q., Shen, G., Dong, B., 2024. A survey on temporal knowledge graph embedding: Models and applications. Knowledge-Based Systems 304, 112454.

## Supplementary Material

A. Semantic Parsing sample examples for SPARQL, SQUALL and Sparklis.

### SPARQL Sample Example

###

You are an expert language parser and you are able to convert from Natural Question to SPARQL query. Below are the <Question> and <Entity Linking Information> defined for you to create the query.

###Question:

Which Matrix movie did Lilly Wachowski not direct?

###Entity Linking Information:

The entity linking on the above mentioned question gave the following potential results:

For the text span The Matrix (franchise) these are the following candidates below within square brackets:

[Q1210827 is a 1999–present films directed by The Wachowskis with a probability 0.8262]

For the text span Lilly Wachowski these are the following candidates below within square brackets:

[Q9544977 is a Polish-American film director with a probability 0.9355]

###Answer:

```
SELECT DISTINCT ?Q11424_1 WHERE ?Q11424_1 wdt:P31 wd:Q11424 . wd:Q1210827 p:P527 [ ps:P527 ?Q11424_1 ] .  
?Q11424_1 p:P57 [ ps:P57 [] ] . FILTER ( NOT EXISTS ?Q11424_1 p:P57 [ ps:P57 wd:Q9544977 ] . )
```

### SQUALL Sample Example

###

You are an expert language parser and you are able to convert from Natural Question to SPARQL query. Below are the <Question> and <Entity Linking Information> defined for you to create the query.

###Question:

Which Matrix movie did Lilly Wachowski not direct?

###Entity Linking Information:

The entity linking on the above mentioned question gave the following entities as results:

For the text span The Matrix (franchise) these are the following candidates below within square brackets:

[The Matrix series is a 1999–present films directed by The Wachowskis with a probability 0.8262]

For the text span Lilly Wachowski these are the following candidates below within square brackets:

[Lilly Wachowski is a Polish-American film director with a probability 0.9355]

###Answer:

which [Film] is a <has part(s)> of [The Matrix Series] and has a <director> and has not <director> [Lilly Wachowski]?

## Sparklis Sample Example

###

You are an expert language parser and you are able to convert from Natural Question to Sparklis query.  
Below are the <Question> and <Entity Linking Information> defined for you to create the query.

**###Question:**

Which Matrix movie did Lilly Wachowski not direct?

**###Entity Linking Information:**

The entity linking on the above mentioned question gave the following entities as results:

For the text span The Matrix (franchise) these are the following candidates below within square brackets:

[The Matrix series is a 1999–present films directed by The Wachowskis with a probability 0.8262]

For the text span Lilly Wachowski these are the following candidates below within square brackets:

[Lilly Wachowski is a Polish-American film director with a probability 0.9355]

**###Answer:**

give me every [Film] that has as a <part of the series> [The Matrix Series] and that not has as a <director> [Lilly Wachowski]