# Synthetic dataset generation and validation with LLMs for text2sparql in Wikidata graphs

Anton Bulle Labate, Breno William Santos Rezende de Carvalho, Sandro Rama Fiorini and Viviane Torres da Silva

*IBM Research Brazil*

Abstract

Fine-tuning large language models (LLMs) for question answering tasks typically requires large datasets and generating representative questions often involves crowdsourcing, which is both expensive and time-consuming. In this paper, we present a method for automatically regenerating questions from the LC-QuAD 2.0 dataset using LLMs, without any human intervention. We show that the regenerated dataset maintains a similar level of question quality compared to the original, by evaluating the quality of the rephrasings and the performance of model fine-tuned with it on a text2sparql task for Wikidata graphs. We demonstrate that fine-tuning LLMs on either dataset yields comparable performance on a text2sparql task. These findings support the hypothesis that LLMs can effectively contribute to the creation of question-answering datasets.

## 1. Introduction

Large Language Models (LLMs) have demonstrated remarkable performance across a wide range of tasks, including text classification, sentiment analysis, and question answering [1, 2]. Among their many applications, LLMs are particularly promising for translating natural language instructions into SPARQL queries — a task known as text-to-SPARQL (text2sparql) — in Wikidata-based knowledge graphs.

While zero- and few-shot prompting techniques have shown potential for text2sparql [3, 4], the current state-of-the-art approaches still rely heavily on pre-training and fine-tuning strategies [5], which in turn require high-quality, representative datasets. Constructing high-quality datasets for question answering is a challenging task, particularly when it comes to generating relevant and fluent natural language questions. Several datasets have been developed for text2sparql in Wikidata, including the QALD series [6] and LC-QuAD 2.0 [7].

The LC-QuAD 2.0 dataset is particularly valuable due to its sufficiently large training split of relatively complex SPARQL queries, which supports fine-tuning of LLMs. The authors of LC-QuAD 2.0 employed crowdsourcing to produce human-written rephrasings of questions derived from Normalized Natural Question Templates (NNQT) [8], which were automatically generated from Wikidata using instantiation rules (see Table 1 for an example). These NNQT questions are typically composed of entity labels and fixed syntactic patterns, resulting in

*Wikidata'25: Wikidata workshop at ISWC 2025*

anton.labate1@ibm.com (A. B. Labate); brenow@ibm.com (B. W. S. R. d. Carvalho); srfiorini@ibm.com (S. R. Fiorini); vivianet@br.ibm.com (V. T. d. Silva)



CEUR Workshop Proceedings (CEUR-WS.org)

**Table 1**
Example sample in LC-QuAD 2.0.

| | |
|---|---|
| **Template** | E REF ?F |
| **NNQT question** | What is <albedo> of <Saturn> ? |
| **Rephr. question** | What is the total solar radiation reflected off of Saturn? |
| **SPARQL** | select distinct ?answer where { wd:Q193 wdt:P4501 ?answer } |

rigid structures that often lack grammatical fluency and naturalness. They are insensitive to variations such as pluralization or article usage, and frequently fail to conform to standard grammar rules.

Manual rephrasing of NNQT questions is resource-intensive and does not always yield high-quality results. In fact, even in LC-QuAD 2.0, some rephrased questions from crowdsourcing exhibit limited fluency and unclear semantics. This highlights a broader issue: human-generated datasets are not immune to quality concerns [9], and dataset quality often matters more than its origin.

In this paper we test whether it is feasible to automatically generate human-like rephrasings from NNQT questions in LC-QuAD. To show that, we describe two experiments. In the first, we use LLMs to regenerate the paraphrases in LC-QuAD 2.0 (Figure 1) and apply an automatic quality assessment method to evaluate the syntactic and semantic validity of the regenerated questions. We prompt four different LLMs to act as evaluators, scoring the questions based on a shared rubric. We then analyze both individual and aggregated scores to iteratively refine our method. Additionally, we validate our automatic assessment by comparing it with human judgments on a small sample of the dataset

In the second experiment, we demonstrate that a text2sparql LLM fine-tuned on our automatically regenerated dataset can achieve a pass@1 performance comparable to that of a model trained on the original, human-curated LC-QuAD 2.0. To this end, we employ a straightforward training pipeline that incorporates embedding-based entity disambiguation and triple sampling. Our findings suggest that human-like question rephrasings can be generated automatically and used effectively for downstream tasks, potentially reducing the reliance on costly and inconsistent manual annotation efforts.

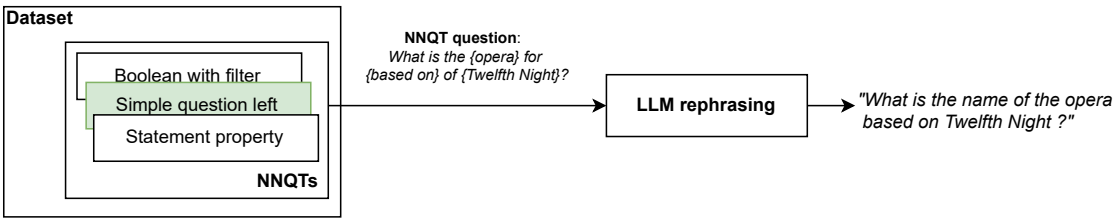


**Figure 1:** Diagram for an example dataset generation. LC-QuAD 2.0is composed of a collection of NNQT questions. We take each of those and use LLMs to rephrase them in human-like question.

## 2. Related work

With the recent and fast advances of the capabilities of Large Language Models, research has been intense on how to leverage their abilities for data generation in various fields. For instance, Borisov et al. [10] use them for generating tabular data. For instruction data generation, Peng et al. [11] use LLMs to generate textual instruction following data for LLM fine-tuning, while Liu et al. [12] use it to produce language-image instruction-following data. For topic classification, Meng et al. [13] and Ye et al. [14] generate the dataset guided only by prompts, in a zero-shot fashion. Also, Yu et al. [15] propose using zero-shot generation for changing attributes in the generation prompt for more diversity in the produced data. In this same task, ProGen [16] provides examples from an influential subset, in a in-context learning manner, in the prompt to guide the generation.

Regarding the construction of text2sparql datasets, one of the largest available datasets is LC-QuAD 2.0 [7]. This dataset has 30,000 question-sparql queries pairs, compatible with both Wikidata and DBpedia 2018 knowledge graphs. The questions in this dataset have varying complexity levels and paraphrases made through crowdsourcing. However, due to this latter aspect, some of the questions do not present the necessary quality for training a model (e.g. some of them have an intent difficult to understand). In MK-Squit [17], the authors resort to automatic dataset generation of text2sparql pairs, instead of crowdsourcing, for three types of templates, using context-free grammar annotation to generate possible variations of them. With the knowledge graph's entities, the authors then fill their templates. Nevertheless, as dataset samples are produced solely by generating different question templates and populating them with appropriate entities and properties, examples derived of the same template are likely to exhibit limited variation and remain highly similar, regardless of the inserted content. Moreover, because only certain parts of the template are substituted (i.e., the placeholders for entities and properties), the resulting samples share rigid structural patterns. This can lead to unnatural phrasing, such as inconsistent use of articles, which in turn may cause the text to appear truncated and lack fluency.

In contrast, with the presented method, we post-process the raw template-infilled questions by asking a LLM to rephrase them so that they sound more human. With this, not only we correct eventual syntactic inconsistencies and lack of fluency due to the rigid template structure, but we also ensure syntactic variability between samples of the same type.

## 3. Experiment: rephrasing

Our goal is to test whether using LLMs to rephrase text2sparql datasets can produce results comparable to a human rephrasing (i.e., done by crowdsourcing). In the first experiment, we regenerate the rephrases of LC-QuAD 2.0 and evaluate them. In the second one (Section 4), we demonstrate that the regenerated dataset can produce a finetuned model for text2code that has similar performance to the model finetuned with original dataset.

## 3.1. Materials and methods

### 3.1.1. Dataset

LC-QuAD 2.0 (LC-QuAD for short, [7]) has 30226 samples (24180 training and 6046 validation) distributed across a broad range of question and query complexity, from 1-hop to 3-hop queries with varied restrictions and aggregation operators. The dataset was originally created by instantiating a collection of (question) templates. A question template defines a type of question-SPARQL pair with variables to be instantiated by graph entity labels and ids, respectively. For our purposes, each sample of the dataset is derived from a question template, and consists of a NNQT question form, a paraphrased version of that and a SPARQL query (see Table 1 for an example).

As a side effect of their construction, NNQT questions exhibit low syntactic and words variability, possibly with poor fluency of text (such as missing determinants) or even standard norm flaws for some cases. In order to approximate the NNQT questions to what users might actually ask, the authors of LC-QuAD crowdsourced the production of the paraphrased questions[1].

For our experiments, we created an updated version of LC-QuAD where we filtered out: (a) questions with invalid SPARQLs or with SPARQLs that take too long to run; and (b) questions in the test split that do not return a valid result in the version of Wikidata which we are using. For all experiments, we used an Wikidata dump from August 2023[2]. That resulted in a new base dataset with 21504 training and 5361 testing samples, 88.6% and 88.4% of the original splits, respectively. When we refer to LC-QuAD dataset in the following, we are referring to this new, cleaned version.

### 3.1.2. Question rephrasing

In our experiment, we generated new rephrasings from the NNQT questions in LC-QuAD. For that, we used a LLM, which we prompted in a zero-shot manner (see Appendix A.2) to rephrase the NNQT questions and generate questions in natural language, similar to those produced by humans, while meeting the English's standard norm. In Figure 1, for the input in NNQT "What is the {opera} for {based on} of {Twelfth Night}? ", the output of the LLM is "What is the name of the opera based on Twelfth Night?", which is more aligned to a question that a person would make. For this purpose, we used Llama 3.1-70b-Instruct [2], with sample decoding and temperature 0.4 to have more variability, and zero-shot prompting. With this process, we end up with two training datasets: (a) one with the original question/SPARQL pairs using the human-created field `question` of LC-QuAD; and (b) another with our rephrased questions paired with the original SPARQLs.

### 3.1.3. Quality assessment and evaluation

Before we used our own dataset to train a model, we evaluated its quality. We prompted 4 language models to act as independent judges and evaluate each generated question rephrase.

[1] The authors of LC-QuAD refer to the paraphrases as verbalizations. In the dataset files, these correspond to the 'question' key.

[2] https://archive.org/details/addshore-wikidata-jnl

The prompt included the NNQT sentence from which it was generated and five evaluation criteria for the judges to check: (a) the rephrased sentence sounds natural and like a human, (b) it is clear in its request, (c) it is well written, (d) the key terms of the NNQT question are the same in the generated question and (e) the rephrasing did not change the original meaning of the NNQT question. We asked the LLMs to give a score from 1 to 5 taking into account all the criteria. The complete evaluation prompt can be found in the Appendix A.1. As evaluator models, we used Llama 3.1-70b-Instruct [2], Granite 3.0-8b-Instruct [18], Mistral-Large-Instruct [19] and Qwen 2.5-72b-Instruct [20].

In order to verify whether we can trust the LLM judges, we created three baselines. In the first one, we submitted the NNQT questions to the same assessment done by the LLM judges described above. The goal was to check whether the LLM judges would give lower scores to them, given their inherent grammatical issues and lack of naturalness. For the second baseline, we checked how many of the *exact* entity labels where preserved in the generated questions, as a more strict, alternative measurement to the evaluation criterion (d) in the list above. For that, we used a simple metric, which we refer to as *simple match*, where we match each of the entity terms in the NNQT question (i.e., the terms in between braces in Figure 1) to words in the generated question. If the term is in the new sentence, we add one to an accumulated sum, until we have checked every term. Then, we divide this sum by the number of terms. We measure the recall of the reference sentence's key terms with the simple match metric. This metric goes from 0 to 1, in which 1 means exact recall. In the third baseline, we created a golden evaluation standard by sampling 3 questions from each question type (69 questions in total) in LC-QuAD and asking 3 human evaluators to grade both the original, human-created question and our LLM-generated question. The goal was to measure how well the automatic metrics relate to human evaluations.

### 3.2. Results and discussion

Table 2 shows the quality assessment result and two of our baselines. Comparing the results in Table 2 for the human and for the generated questions, through our automatic evaluation, we can verify that, for every LLM judge, the generated questions obtain a higher score, which indicates that they are more clear and sound than the original human questions in LC-QuAD (and adherent to the original NNQT). We have seen questions in LC-QuAD with misspelling problems and ones without fluency, which also justifies this.

In relation to the baselines, Table 2 shows that, as expected, all LLM judges gave the NNQT questions a low grade indicating that in fact they recognize low quality questions. Furthermore, the high score for generated questions under the simple match metric indicates that our method can better preserve terms in the produced questions, with less hallucination.

Finally, the results for our third remaining baseline are shown in Table 3. It shows the correlation between assessment done by our LLM judges and the evaluation made by the three human evaluators, both on the original human questions and in the generated ones. We can observe that our LLM judges' scores are somehow tied to the human evaluation, in the view of the fact that they have a considerable correlation. In fact, for most of the LLMs the correlation with the average of the scores given by human judges is higher than 0.5.

Taken as a whole, these results indicate that the generated questions are at least adequate

**Table 2**
Evaluation of the original LC-QuAD and generated datasets, per LLM and their average score over all samples (and standard deviation), and simple match metric. The simple match metric for the NNQT questions is 1.0 by definition.

| Evaluator | LC-QuAD (NNQT) | | LC-QuAD (Human) | | Generated | |
|---|---|---|---|---|---|---|
| | Avg. | St. Dev. | Avg. | St. Dev. | Avg. | St. Dev. |
| Llama 3.1-70b-Instruct | 1.87 | 0.78 | 3.71 | 1.12 | **4.39** | 0.72 |
| Granite 3.0-8b-Instruct | 1.53 | 0.72 | 3.01 | 1.26 | **3.78** | 0.80 |
| Mistral-Large-Instruct | 1.47 | 0.58 | 3.01 | 1.29 | **3.93** | 1.08 |
| Qwen 2.5-72b-Instruct | 1.38 | 0.61 | 3.48 | 1.34 | **4.40** | 0.95 |
| Average LLMs | 1.56 | 0.53 | 3.30 | 1.10 | **4.13** | 0.70 |
| Simple Match | - | - | 0.64 | 0.30 | **0.73** | 0.26 |

substitutes for the questions created by crowdsourced ones. This is a positive result, given the difficulties and costs involved in hiring and creating crowdsourced datasets.

**Table 3**
Correlation between LLM judges and human evaluators (#1, #2 and #3) for a subset of LC-QuAD. The columns "Avg." denote the correlation of the average score of the human judges with each automatic judge.

| | LC-QuAD (Human) | | | | Generated | | | |
|---|---|---|---|---|---|---|---|---|
| **Model** | #1 | #2 | #3 | Avg. | #1 | #2 | #3 | Avg. |
| Llama 3.1-70b-Instruct | 0.52 | 0.53 | 0.29 | **0.58** | 0.41 | 0.28 | 0.30 | 0.43 |
| Granite 3.0-8b-Instruct | 0.29 | 0.30 | 0.42 | 0.44 | 0.45 | 0.31 | 0.56 | **0.57** |
| Mistral-Large-Instruct | 0.43 | 0.43 | 0.27 | 0.48 | 0.52 | 0.52 | 0.28 | **0.58** |
| Qwen 2.5-72b-Instruct | 0.56 | 0.56 | 0.40 | **0.65** | 0.45 | 0.38 | 0.45 | **0.56** |
| Average LLMs | 0.52 | 0.52 | 0.40 | **0.62** | 0.58 | 0.48 | 0.50 | **0.68** |

# 4. Experiment 2: Model fine-tuning

To further assess whether using LLMs to rephrase text2sparql questions produced valid and useful results, we did a second experiment where we fine-tuned two LLM models for a text2sparql task in Wikidata: (a) one based on the original LC-QuAD questions rephrased by humans and (b) another based on our LLM-generated rephrasings. We then compared both models' performance on the test split of both original LC-QuAD and our generated version.

## 4.1. Materials and methods

For both models, we used the original dataset as described in Section 3.1.1 and the generated dataset resulting from Experiment 1.

The text2sparql inference approach we aimed for is similar to existing approaches (i.e., [3], [21]), which involve a retrieval step for helping the model disambiguate question terms, but using fine-tuning to improve SPARQL generation (i.e., [22]). We calculated all Wikidata entities mentioned in LC-QuAD and retrieved a list of all their labels and alternate labels, resulting

in a list of 88279 terms. We embedded each term using `slate-125m-english-rtrvr-v2` embedding model [23] and populated a vector database with these embeddings and their corresponding wiki ids, segregated by entity type; i.e., segregated items from properties. Given an input question, we start by running the question through an LLM (Mixtral-8x7B-Instruct-v01 [24], with greedy decoding) with a prompt asking it to extract terms referring to items and properties. We then create two versions of the question: (a) a q-question, where we concatenate the item terms back to the original question; and (b) a p-question, where we do the same with property terms. For example, considering the sentence in "In which country is Paris located?", its q-question would be "In which country is Paris located? *country Paris*" and its p-question would be "In which country is Paris located? *located*". We subsequently use the embedding of the q-question to retrieve the 10 most cosine-similar items and similarly the p-question to retrieve the 10 most cosine-similar properties. Whilst counterintuitive, we found that the term extraction and subsequence concatenation helps increasing recall in the disambiguation phase. Following that, for each sample, we used this list of 20 entities to sample triples from the knowledge graph that involved those 20 items and properties, removing entities from the initial list for which triples could not be found (we also ignore triples with negative dates). With this restricted list and the sampled triples for each example, we created a prompt asking a LLM to generate the corresponding SPARQL query, using tag sequences to delimit the query. These prompts were constructed for all samples in both datasets. The template for this prompt and an instantiated example can be found in Appendix A.3.

We used these prompts to fine-tune Mistral-7B-Instruct-v0.2 [25] with LoRA [26]. For the tuning, we used a 4-bit quantization, with LoRA parameters $r = 32$ and $\alpha = 64$, as well as dropout $= 0.05$. We used $2.5e$-5 learning rate, 100 warm-up steps, in 3 epochs with 2 samples per batch and 4 gradient accumulation steps. We fine-tuned one adaptor model on each training dataset; i.e., on the original questions and on the generated ones.

For evaluation, we used two metrics. First, we evaluated the disambiguation phase used to build the prompts. For that we used the *perfect recall metric*, defined as the number of samples in the dataset in which the retrieval returned all relevant Wikidata entities to the query.

For evaluating model inference, we calculated prompts using the same disambiguation and graph sampling strategy as in training. We used greedy search with 3 beams (i.e., beam-search decoding). While somewhat counter-intuitive given the deterministic nature of greedy search, we found that this technique improved results by eliminating answer variations, possibly to very small numeric instabilities in generation. We generated SPARQL queries for all samples and ran them into our version of Wikidata (see Section 3.1.1), ignoring any samples that did not work anymore given updates from when the original LC-QuAD 2.0 was created. We considered a generated query to be correct if its result set is the same as the golden, irrespective of projection and row orders. However, we did consider row order for ranking queries.

For evaluating the results, we used pass@k metric as implemented in HumanEval benchmark [27]. This metric is the *de facto* metric for code generation, being more strict and clearly defined than the commonly used F1. Furthermore, given that we used greedy search for inference, we only calculated pass@k for $k = 1$.

**Table 4**
Pass@1 results of our fine-tuned model with human questions (the control dataset) and our LLM-generated questions.

| | **Test dataset** | |
|---|---|---|
| **Fine-tuning dataset** | Human | LLM |
| Human | 0.51 | 0.49 |
| LLM | 0.50 | 0.54 |

### 4.2. Results and discussion

The inference result is dependent on the disambiguation phase responsible for building the prompts. In the original LC-QuAD dataset, the measured perfect recall was 0.53 for both train and test splits. For the generated dataset, it was 0.58 for the train split and 0.54 for the test split. While we could have used a less strict recall metric, the perfect recall puts in context the results of text2sparql inference. A given inference will give the model a harder time generating the correct SPARQL if the question terms where not disambiguated properly. Interestingly, both datasets have similar perfect recall measures, which indicates that both present the same level of difficulty for disambiguation.

For the inference evaluation, the scores are shown in Table 4. These results can be seen through two perspectives. Firstly, the results indicate that the generated training dataset is as representative and significant in model training for text2sparql as the original dataset, enabling the model to achieve similar learning and comprehension. The second perspective relates to the importance of the *crowdsourced test split* in LC-QuAD. If one considers this split as a gold-standard proxy for real-world queries — given it is human-curated — then the similar performance of both fine-tuned models on it suggests that using synthetic data for training text2sparql can be a cost-effective way to achieve state-of-the-art performance for real use cases.

Finally, the results of both experiments should be interpreted considering the possibility that LC-QuAD 2.0was part of the pre-training data and subsequently memorized by the LLMs used in our study. This hypothesis is difficult to verify without direct access to the original training corpus. Nevertheless, the higher quality scores achieved by the generated questions in the quality assessment (Table 2) suggest that, even if some memorization occurred, its impact was likely limited.

## 5. Conclusion

We presented an experiment for synthetic question generation for text2sparql tasks in Wikidata without human annotation or crowdsourcing. The experiment includes a question-evaluation step leveraging multiple LLMs to ensure syntactic, semantic, and fluency quality, reducing the need for manual verification. We successfully recreated a crowdsourced text2sparql dataset, achieving higher question quality and comparable model performance. We have verified the new dataset to be as representative as the original, by using it to fine-tune a LLM model with a similar performance than one fine-tuned on the original dataset. Future work includes applying our dataset generation methodology to a completely synthetic dataset built from scratch, which

could test the generality of the methodology, with less chance of memorization. Additionally, exploring alternative inference strategies and models using the generated dataset offers potential for further improvements. Another avenue for research is leveraging the results of our automatic evaluation process to enhance the quality of the generated questions.

## References


[1] T. B. Brown, Language models are few-shot learners, arXiv preprint arXiv:2005.14165 (2020).

[2] A. Grattafiori, et al., The llama 3 herd of models, arXiv preprint arXiv:2407.21783 (2024). URL: https://arxiv.org/abs/2407.21783. arXiv:2407.21783.

[3] C. V. S. Avila, V. M. Vidal, W. Franco, M. A. Casanova, Experiments with text-to-sparql based on chatgpt, in: 2024 IEEE 18th International Conference on Semantic Computing (ICSC), IEEE, 2024, pp. 277–284.

[4] J. D'Abramo, A. Zugarini, P. Torroni, Dynamic few-shot learning for knowledge graph question answering, 2024. URL: https://arxiv.org/abs/2407.01409. arXiv:2407.01409.

[5] Z. Li, S. Fan, Y. Gu, X. Li, Z. Duan, B. Dong, N. Liu, J. Wang, Flexkbqa: A flexible llm-powered framework for few-shot knowledge base question answering, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, 2024, pp. 18608–18616.

[6] R. Usbeck, X. Yan, A. Perevalov, L. Jiang, J. Schulz, A. Kraft, C. Möller, J. Huang, J. Reineke, A.-C. Ngonga Ngomo, et al., Qald-10–the 10th challenge on question answering over linked data, Semantic Web (2023) 1–15.

[7] M. Dubey, D. Banerjee, A. Abdelkawi, J. Lehmann, Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia, in: The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part II 18, Springer, 2019, pp. 69–78.

[8] P. Trivedi, G. Maheshwari, M. Dubey, J. Lehmann, Lc-quad: A corpus for complex question answering over knowledge graphs, in: The Semantic Web–ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II 16, Springer, 2017, pp. 210–218.

[9] T. Hosking, P. Blunsom, M. Bartolo, Human feedback is not gold standard, arXiv preprint arXiv:2309.16349 (2023).

[10] V. Borisov, K. Seßler, T. Leemann, M. Pawelczyk, G. Kasneci, Language models are realistic tabular data generators, arXiv preprint arXiv:2210.06280 (2022).

[11] B. Peng, C. Li, P. He, M. Galley, J. Gao, Instruction tuning with gpt-4, arXiv preprint arXiv:2304.03277 (2023).

[12] H. Liu, C. Li, Q. Wu, Y. J. Lee, Visual instruction tuning, Advances in neural information processing systems 36 (2024).

[13] Y. Meng, J. Huang, Y. Zhang, J. Han, Generating training data with language models: Towards zero-shot language understanding, Advances in Neural Information Processing Systems 35 (2022) 462–477.

[14] J. Ye, J. Gao, Q. Li, H. Xu, J. Feng, Z. Wu, T. Yu, L. Kong, Zerogen: Efficient zero-shot learning via dataset generation, arXiv preprint arXiv:2202.07922 (2022).

[15] Y. Yu, Y. Zhuang, J. Zhang, Y. Meng, A. Ratner, R. Krishna, J. Shen, C. Zhang, Large language model as attributed training data generator: A tale of diversity and bias, 2023. URL: https://arxiv.org/abs/2306.15895. arXiv:2306.15895.

[16] J. Ye, J. Gao, J. Feng, Z. Wu, T. Yu, L. Kong, Progen: Progressive zero-shot dataset generation via in-context feedback, arXiv preprint arXiv:2210.12329 (2022).

[17] B. A. Spiegel, V. Cheong, J. E. Kaplan, A. Sanchez, Mk-squit: Synthesizing questions using iterative template-filling, 2020. URL: https://arxiv.org/abs/2011.02566. arXiv:2011.02566.

[18] I. Granite Team, Granite 3.0 language models, https://github.com/ibm-granite/granite-3.0-language-models/, 2024. Accessed: 2025-01-08.

[19] Mistral AI Team, Mistral large, https://mistral.ai/news/mistral-large/, 2024. Accessed: 2025-01-08.

[20] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, Z. Qiu, Qwen2.5 technical report, 2025. URL: https://arxiv.org/abs/2412.15115. arXiv:2412.15115.

[21] H. Wmid, Making Question-Answering Systems Smarter with Knowledge Graphs Using FrOG: A Wikidata Research Fund 2024 Highlight, Technical Report, Wikimedia Indonesia, 2025. URL: https://diff.wikimedia.org/2025/07/23/making-question-answering-systems-smarter-with-knowledge-graphs\-using-frog-a-wikidata-research-fund-2024-highlight/.

[22] D. Banerjee, P. A. Nair, J. N. Kaur, R. Usbeck, C. Biemann, Modern baselines for sparql semantic parsing, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 2260–2265.

[23] IBM WatsonX Team, Ibm slate-125m-english-rtrvr-v2 model card, https://dataplatform.cloud.ibm.com/docs/content/wsj /analyze-data/fm-slate-125m-english-rtrvr-v2-model-card.html?context=wx#expand, 2024. Accessed: 2025-01-08.

[24] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. d. l. Casas, E. B. Hanna, F. Bressand, et al., Mixtral of experts, arXiv preprint arXiv:2401.04088 (2024).

[25] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, W. E. Sayed, Mistral 7b, 2023. URL: https://arxiv.org/abs/2310.06825. arXiv:2310.06825.

[26] Y. Yu, C.-H. H. Yang, J. Kolehmainen, P. G. Shivakumar, Y. Gu, S. R. R. Ren, Q. Luo, A. Gourav, I.-F. Chen, Y.-C. Liu, et al., Low-rank adaptation of large language model rescoring for parameter-efficient speech recognition, in: 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), IEEE, 2023, pp. 1–8.

[27] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford,

M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, W. Zaremba, Evaluating large language models trained on code, 2021. URL: https://arxiv.org/abs/2107.03374. arXiv:2107.03374.

# A. Used prompts

## A.1. Evaluation prompt

The evaluation prompt used for the LLMs to assess the quality of the generated sentences was (arrows denote line breaks for fitting long lines in this page):

*You are a great and very strict English teacher, who pays close attention to details. You will be given a reference sentence and a sentence that rephrases this first sentence. Your task is to rate from 1 to 5 the second phrase according to the score rubric provided:*

*###Score Rubrics:*

*Given the following criteria, give the rephrased sentence a grade.*
*1: Does the rephrased sentence sounds natural and like a human?*
*2: Does the new phrase maintain the terms in brackets?*
*3: Does the rephrased sentence has the same meaning of the first sentence without adding information to the reference sentence? Pay attention if the semantics of the two phrases are the same.*
*4: Is the rephrased sentence clear in its intention and about what it requests?*
*5: Does the rephrased sentence has correct syntax and grammar without misspelling?*

*If the rephrased sentence fails in any of these criteria, penalize it in your final score.\n Provide only the score ranging from 1 to 5 as answer, with feedback or notes. If no rephrase sentence is provided, give a score of 1. Format your answer as follows:\n ```\n### Feedback: <<Insert your feedback>> \n ### Score: <<Insert your score>>\n ```\n*

## A.2. Prompt for rephrasing

For generating the natural language questions, we prompted a LLM to rephrase the NNQT questions. The system prompt that we use to guide the model through the task is the following:

*You are a great English teacher and grammar corrector. It is given to you a reference phrase, with key terms in curly brackets.*
*Your task is to rewrite the given sentence in a more human–like and natural question, while maintaining the phrase's original semantics. Make sure you follow the guidelines below when generating your answer:*

1. *You need to maintain the phrase's original semantics. Pay close attention to the relations between the terms in the reference phrase to reflect them also in the generated sentence;*
2. *Try to be as concise as possible;*
3. *You may modify the phrase's structure and syntax to make it sound more natural, AS LONG AS you maintain the semantics of the reference sentence;*

*4. Do not add new information in the generated phrase;*
*5. Please rephrase without any comment or notes and answer only with the new phrase.*

## A.3. Fine-tuning prompt

Finetuning prompt carried out in all experiments used the following prompt template:

*<s>[INST] You are a code assistant. Your task is to generate a valid SPARQL query based on the given context and input:*

*### Input:*
*{question}*

*### Context:*
*{equivalences}*

*{subgraphs}*

*Just generate SPARQL query that implement the Input based on the Context, without explanation: [/INST] {query} </the-code></s>*

where *question* refers to the rephrased question of a given sample; *equivalences* refer to the relevant entities calculated with our retrieval method for that sample; *subgraphs* refer to the sampled triples related to those entities ; and *query* refer to the ground truth SPARQL query. The prompt also include instruct tokens specific for mistral and a specific end-of-sequence tag to mark the end of the (query) code sequence generated. Arrows denote line breaks for making long lines fit in this page. Below there is an instantiated prompt for the sample 19719 of LC-QuAD 2.0:

*<s>[INST] You are a code assistant. Your task is to generate a valid SPARQL query based on the given context and input:*

*### Input:*
*What periodical literature does Delta Air Lines use as a moutpiece?*

*### Context:*
*Q32396 = American Airlines, AAL, AA, American Air Lines, American Airlines Group, American Airlines, Inc., American Airways, AMR Corporation*
*P432 = callsign of airline, airline callsign, airl aéronefine code, airline call sign*
*Q188920 = Delta Air Lines, Delta Air Lines, Inc., DL, Delta, DAL*
*Q46970 = airline, air carrier*

*American Airlines callsign of airline AMERICAN*
*Delta Air Lines callsign of airline DELTA*

*Just generate SPARQL query that implement the Input based on the Context, without explanation: [/INST] select distinct ?obj where { wd:Q188920 wdt:P2813 ?obj . ?obj wdt:P31 wd:Q1002697 } </the-code></s>*

Note that in this particular example, the context has imperfect recall. Our retriever was not able to retrieve entities Q1002697 and P2813 that were necessary to build the query.