KGC-RAG: Knowledge Graph Construction from Large Language Model Using Retrieval-Augmented Generation

Thin Prabhong[1], Natthawut Kertkeidkachorn[2] and Areerat Trongratsameethong[1]

[1]Chiang Mai University, Chiang Mai, Thailand
[2]Japan Advanced Institute of Science and Technology (JAIST), Ishikawa, Japan

Abstract

The construction of Knowledge Graphs (KGs) has become increasingly important due to their ability to integrate and represent complex relationships across various domains, making them essential for applications like information retrieval and semantic search. Recently, Large Language Models (LLMs) have been utilized to enhance KGs creation by leveraging their advanced capabilities in understanding and generating human-like text. The Large Language Models for Knowledge Engineering (LLMKE) pipeline was introduced to combine knowledge probing with Wikidata entity mapping for Knowledge Engineering. Nevertheless, this approach has a limitation: it primarily relies on retrieval-augmented context drawn from the first paragraph and Wikipedia Infobox of the subject entity's page. This narrow focus can lead to incomplete knowledge representations, as relevant information is often spread throughout the text and linked pages. To address this issue, we propose the Knowledge Graph Construction from large language model using Retrieval-Augmented Generation method (KGC-RAG). This method leverages web scraping to retrieve documents from the subject entity's Wikipedia page and to extend the search to include linked pages, thereby increasing the likelihood of capturing comprehensive and contextually rich information. We further enhance this approach by using LLM in conjunction with cosine similarity to filter out irrelevant content, ensuring that only the most pertinent data are included in the relevant context. We conducted experiment on datasets from ISWC 2024 LM-KBC Challenge and applied the meta-llama/Meta-Llama-3-8B-Instruct model as our pre-trained large language model along with the all-MiniLM-L6-v2 as our vector embedding model. We set a relevant score threshold of 0.5 to filter Wikipedia URLs. Our approach achieved macro average F1-scores of 0.695 and 0.698 on the validation and test sets, respectively. The implementation is available at https://github.com/jaejeajay/LM-KBC2024.

1. Introduction

Knowledge graphs [1] store information in a Subject-Predicate-Object format, providing more efficient semantic data storage compared to relational. They are more easily understood by computers, offering flexibility and the ability to integrate diverse information, making them crucial for applications such as question answering and recommendation systems. A notable example of a significant knowledge graph repository is Wikidata [2], an open-source platform by the Wikimedia Foundation for storing factual data about the world.

KBC-LM'24: Knowledge Base Construction from Pre-trained Language Models workshop at ISWC 2024
thin.pra2013@gmail.com (T. Prabhong); natt@jaist.ac.jp (N. Kertkeidkachorn); areerat.t@cmu.ac.th (A. Trongratsameethong)
https://github.com/jaejeajay/LM-KBC2024 (T. Prabhong)

Constructing knowledge graphs (KGs) is highly challenging [3] due to the need for access to vast and diverse information sources. Large Language Models (LLMs) are gaining popularity for their ability to perform a wide range of tasks, such as answering questions, providing information, translating languages, and engaging in conversations. Since LLMs are trained on extensive datasets, they serve as comprehensive repositories of knowledge. Extracting knowledge from LLMs can yield valuable information for various applications, including the construction of knowledge graphs, where LLMs can provide foundational data and relationships needed for building these complex structures.

However, challenges in extracting knowledge from LLMs [4] include the generation of fabricated answers and outdated information. Over time, facts can change or be disproven, but the data used to train LLMs remain factual only as of the time of training. Thus, optimizing LLMs for accurate and reliable responses is crucial.

Retrieval-Augmented Generation (RAG) [5] offers an effective solution to these challenges. By combining retrieval and generation processes, RAG enhances the accuracy of LLM outputs by retrieving up-to-date information from external databases or sources, thereby providing the necessary context for generating more factual and reliable responses.

In this study, we explore the construction of knowledge graphs using knowledge extracted from LLMs. We optimize LLM performance with RAG by improving the quality of relevant context through web scraping and web crawling. The relevance score is determined by the path names in the Wikipedia URLs. We utilized datasets from the ISWC 2024 LM-KBC Challenge [1], where the task limited LLM parameters to 10B. We selected the meta-llama/Meta-Llama-3-8B-Instruct model for our study, and the results demonstrate that our approach outperforms the baseline.

2. Related Works

In recent research, Zhang et al. [6] explored the use of large language models (LLMs) for knowledge engineering tasks within the context of the ISWC 2023 LM-KBC Challenge. They utilized pre-trained LLMs to generate relevant objects in string format from given subject-relation pairs sourced from Wikidata, subsequently linking these objects to their respective Wikidata QIDs. The developed pipeline, known as Large Language Models for Knowledge Engineering (LLMKE), combines knowledge probing with Wikidata entity mapping and incorporates retrieval-augmented context to enhance predictions. This context is derived from external sources, such as Wikipedia, to refine the model's responses by comparing and integrating this information with the model's initial predictions.

Nonetheless, LLMKE operates under the assumption that the most relevant documents are primarily found in the first paragraph (Introduction) and the Wikipedia Infobox, which may not always be the case, as related information can be dispersed throughout various sections.

To address this limitation, we propose a method to enhance content extraction from Wikipedia by employing web scraping techniques to gather information from paragraphs, Infoboxes, and Wikitables on the Wikipedia page of the subject entity. Additionally, this approach extends the

[1] https://github.com/lm-kbc/dataset2024

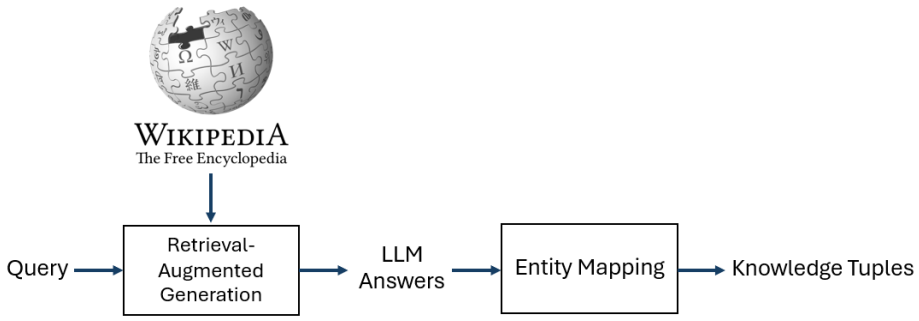

Figure 1: The overview of the KGC-RAG method

search to other linked Wikipedia pages, thereby capturing more comprehensive and relevant information.

One challenge of extending the search beyond the subject entity's Wikipedia page is the increased volume of documents, which can slow down the process. To mitigate this, we implemented a solution using LLM and cosine similarity scores to filter and prioritize relevant Wikipedia pages and documents, ensuring efficient and effective information retrieval.

3. Methods

The Knowledge Graph Construction from large language model using Retrieval-Augmented Generation method (KBC-RAG) provides an overview as shown in Figure 1. This method is composed of two processes: Retrieval-Augmented Generation (RAG) and Entity Mapping. In the RAG process, Relevant Contexts are identified and used to enhance the LLM's Knowledge Extraction. In the Entity Mapping process, the answers from the LLM are used to construct Knowledge Graphs by mapping through API functions from Wikidata, resulting in completed knowledge tuples. The details of each process are as follows:

3.1. Retrieval-Augmented Generation (RAG)

This process serves as a preliminary overview presented in Figure 2. The goal of this process is to extract knowledge from LLM using RAG as an LLM optimization method. The outcome of this process will be LLM-generated answers, which will then be used for subsequent Entity Mapping.

3.1.1. Entity ID query

This step involves locating the Wikipedia URL of the subject entity using the provided SubjectEntityID, which is essential for the web scraping process. The SubjectEntityID is used to

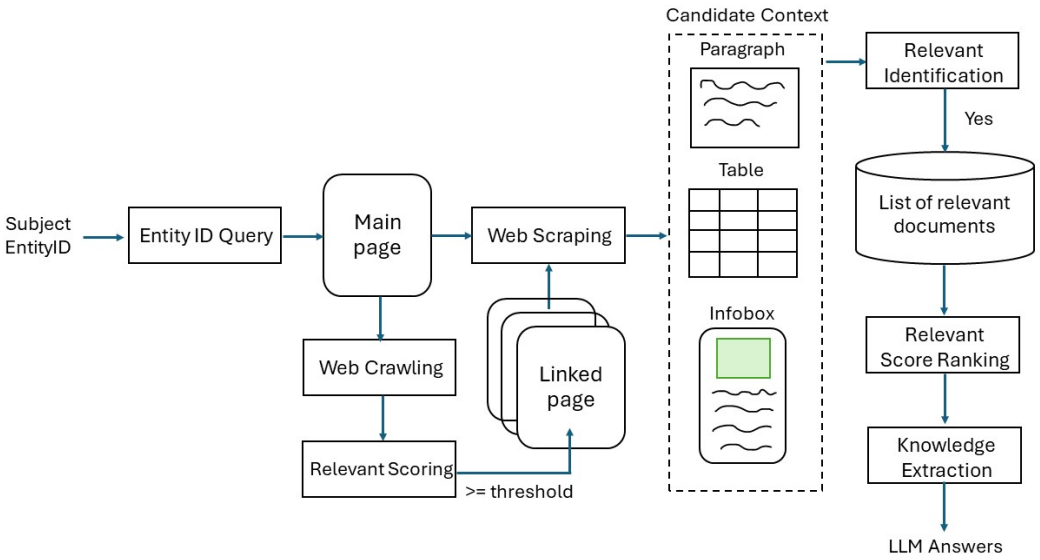

Figure 2: The overview of Retrieval-Augmented Generation (RAG)

search for the URL in Wikidata through an API call via SPARQLWrapper [2], with a preference for English URLs. The outcome of this step is the subject entity's Wikipedia URL, which is used to access the subject entity's Wikipedia page, referred to as the Main Page in this study.

3.1.2. Web Scraping

In this step, the focus is on scraping data from the Wikipedia page. The data targeted for scraping includes three tags: *p* (Paragraph), Infobox, and Wikitable. The output of this step is the scraped data from Wikipedia, which will later be assessed for relevance in the Relevant Identification step.

3.1.3. Web Crawling

This step is designed to broaden the scope of web scraping by extending the search from the Main Page to other linked Wikipedia pages, thereby increasing the chances of identifying Relevant Context. This is achieved by identifying <*a*> tags within the Main Page. The result of this step is a collection of linked Wikipedia URLs linked to the Main Page.

3.1.4. Relevant Scoring

The primary aim of this step is to filter out unnecessary linked URLs from the web crawling process by using a Relevant score, calculated via cosine similarity. This approach helps

[2]https://sparqlwrapper.readthedocs.io/en/latest/main.html

Table 1
Templates of Prompt for web scraping

Relation	Prompt for web scraping
awardWonBy	”Who has won ” + SubjectEntity + ” ?”
companyTradesAtStockExchange	”Which stock exchange does ” + SubjectEntity + ” trade on ?”
countryLandBordersCountry	”Which country share land border with ” + SubjectEntity + ” ?”
personHasCityOfDeath	”What is the city of death of ” + SubjectEntity + ” ?”
seriesHasNumberOfEpisodes	”How many episodes does series ” + SubjectEntity + ” has ?”

streamline the RAG process. The filtering method involves calculating the cosine similarity between the path name of each URL linked to the Main Page and the Prompt for web scraping, utilizing the vector model all-MiniLM-L6-v2. The templates of the Prompt for Web Scraping are shown in Table 1. For instance, from the URL https://en.wikipedia.org/wiki/List_of_Nobel_laureates_in_Physiology_or_Medicine, only the part ”List of Nobel laureates in Physiology” is used to calculate the Relevant Score. URLs with a cosine similarity score above the threshold are selected for web scraping. The outcome of this step is a refined list of linked URLs for web scraping.

3.1.5. Relevant Identification

This step focuses on filtering scraped data from Wikipedia by leveraging LLM responses to specific questions. The result is a list of relevant documents, which will be used in the subsequent step for determining the relevant score.

We begin by filtering each paragraph, Infobox, and Wikitable data using the Meta-Llama-3-8B-Instruct Model through question-based evaluation. An example question might be: *Is this information ’[Paragraph/Infobox/Wikitable]’ able to answer the question: ’[Prompt for web scraping]’?* Information deemed relevant by the LLM is added to the list of relevant documents.

3.1.6. Relevant Score Ranking

In this step, the task is to retrieve the Relevant Context from list of relevant documents. This is accomplished by calculating the cosine similarity between each document in the list and Prompt for web scraping, using the vector model all-MiniLM-L6-v2, as done in the Relevant Scoring step. The top K documents are then selected to form the Relevant Context. The result is a Relevant Context, which will be integral to the retrieval process in Knowledge Extraction.

3.1.7. Knowledge Extraction

The purpose of this step is to extract knowledge from LLM by using Relevant Context. The result of this step will be LLM-generated answers that are ready for entity mapping.

To ask questions using the Prompt for Knowledge Extraction, as shown in Table 2, and the obtained Relevant Context, the input message format consists of three parts:

1. **Relevant Context:** The first message provides the Relevant Context to the LLM in the following format:

Table 2
Templates of Prompt for Knowledge Extraction

Relation	Prompt for Knowledge Extraction
awardWonBy	"Provide a name only list of all award winners in " + SubjectEntity + " with no explanation and name with comma ?"
companyTradesAtStockExchange	"Which stock exchange does " + SubjectEntity + " trade on ? answering with no explanation and name with comma? If None, answer None."
countryLandBordersCountry	"Which countries share land borders with " + SubjectEntity + " with country name only with comma? If None, answer None."
personHasCityOfDeath	"What is the city of death of " + SubjectEntity + "? answering with one city name only with no explanation. If there is no place of death mentioned, answer None"
seriesHasNumberOfEpisodes	"How many total episodes of series " + SubjectEntity + " ? answering with only one number ?"

{"role": "system", "content": "Using this context to answer the question: " + [Relevant Context]}.

2. **Behavior Setting:** The second message sets the behavior of the Chatbot of LLM to ensure responses are as required:

   {"role": "system", "content": "You are a chatbot who always responds an answer in english with comma and no explanation. If u don't know the answer, answer None"}.

3. **Question Prompt:** The third message is used to ask the question and is formatted as:

   {"role": "user", "content": [Prompt for Knowledge Extraction]}.

This structured approach ensures that the LLM has the necessary context and guidance to provide accurate and concise answers.

3.2. Entity Mapping

The goal of this process is to construct a Knowledge Graph using the answers from the LLM obtained in the previous session. The result of this process will be completed knowledge tuples based on the given subject entity and relation.

We begin by mapping the answers to create a Knowledge Graph using *wbsearchentities*, an API function provided by the Wikidata Query Service [3]. This function searches for entities in Wikidata using labels or keywords and returns a list of matching entities ranked by relevance. The method used to select entities for linking as objects is 'Choose First' for all relations. After selecting an entity, its validity is verified by checking the range of each relation. As shown in Table 3, if the entity's 'instance of' is within the same range of relations as the subject, the entity is selected for mapping.

[3] https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service

Table 3
Domain and Range for each relation

Relation	Domain	Range
awardWonBy	Award	Human
companyTradesAtStockExchange	Company	Stock Exchange
countryLandBordersCountry	Country	Country
personHasCityOfDeath	Human	City
seriesHasNumberOfEpisodes	Series	Number

Table 4
Number of unique subject-entities in the data splits

Relation	Train	Val	Test	Special features
awardWonBy	10	10	10	Many objects per subject
companyTradesAtStockExchange	100	100	100	Null values possible
countryLandBordersCountry	68	68	68	Null values possible
personHasCityOfDeath	100	100	100	Null values possible
seriesHasNumberOfEpisodes	100	100	100	Object is numeric

In cases where responses from LLM exhibit ambiguity, it may be due to the LLM relying too heavily on context. For instance, in the Relation companyTradesAtStockExchange, stock exchange information for a company on Wikipedia is often presented in the format "Traded as", such as *"West Japan Railway Company traded as TYO: 9021"*, where TYO is the stock exchange name and 9021 is the stock exchange code. The LLM might provide an answer like "TYO: 9021". To address this ambiguity, we employ hard coding to filter and refine the responses from the LLM, thereby improving the accuracy and clarity of the output.

4. Experiments and Results

4.1. Datasets

The datasets used in this study were obtained from ISWC 2024 LM-KBC Challenge. Details of the datasets are shown in Table 4. Each dataset consists of 378 unique subject-entities and 5 relations, with object entities referenced from Wikidata. Additionally, each relation has special features that describe the characteristics of the object entities, which vary by relation. For example, the special feature of the relation *countryLandBordersCountry* is the possibility of Null values. For instance, New Zealand, being an island nation with no land borders, would have a Null value for the object entity corresponding to the given subject entity "New Zealand" and the given relation *countryLandBordersCountry*.

4.2. Experiment settings

The study utilizes meta-llama/Meta-Llama-3-8B-Instruct model as the large language model and the all-MiniLM-L6-v2 vector model for cosine similarity calculations. We selected Wikipedia

as our main data source. The relevant score threshold for filtering Wikipedia URLs is set at 0.5. Each document is chunked into segments of 4500 tokens. Use Top K, where K=20, to combine documents into a single context. The Relevant Identification process is configured with `max_new_tokens=1`, `temperature=0.1`, and `top_p=0.9`. For the Knowledge Extraction process, the parameters are set to `max_new_tokens=3000`, `temperature=0.1`, and `top_p=0.9`. We used these settings for both validation and test set. The implement is available at https://github.com/jaejeajay/LM-KBC2024.

4.3. Baseline

Our study used the baseline of meta-llama/Meta-Llama-3-8B-Instruct from ISWC 2024 LM-KBC Challenge[1]. The baseline was derived from the use of a Masked Language Model, an Autoregressive Language Model, and Llama-3 chat models.

4.4. Evaluation Metrics

In our study, we used the following statistical measures for evaluation:

- **Macro-Precision (Macro-P)**: The average precision score across all classes, calculated by taking the precision for each class and then averaging these values.
- **Macro-Recall (Macro-R)**: The average recall score across all classes, computed by taking the recall for each class and averaging these values.
- **Macro-F1**: The average F1 score across all classes, computed by taking the F1 score for each class and averaging these values.

4.5. Results

We conducted experiments on our system using the validation set, with the results presented in Table 5. The experimental results demonstrated that our approach outperformed the baseline, achieving an average F1-score of 0.695. For the test set, we submitted the results to the ISWC 2024 LM-KBC challenge. The test results, shown in Table 6, indicate an average F1-score of 0.698, which is consistent with the performance on the validation set.

5. Discussion

5.1. Data Source Quality

The source of context information plays a critical role in determining the coverage score of the context [7]. If a data source provides limited information, the coverage score will be lower, which subsequently affects the responses generated by the LLM [8]. ISWC 2024 KM-KBC Challenge this year is represented by five distinctive relations, particularly the *awardWonBy* and *companyTradesAtStockExchange* relations. For the *awardWonBy* relation, some subjects on Wikipedia do not display the award winners on the main award page, but instead on a separate "List of laureates" page. Similarly, for the *companyTradesAtStockExchange* relation, information about stock exchanges may not be directly provided or may be absent from the Wikipedia page. Therefore, utilizing effective data sources and retrieval methods is crucial.

Table 5
The results of our system and baseline (meta-llama/Meta-Llama-3-8B-Instruct) on validation set

Method	Relation	macro-p	macro-r	macro-f1
Baseline	awardWonBy	0.238	0.028	0.045
	companyTrades AtStockExchange	0.540	0.703	0.474
	countryLand BordersCountry	0.961	0.912	0.919
	personHas CityOfDeath	0.700	0.600	0.460
	seriesHas NumberOfEpisodes	0.493	0.160	0.155
	All Relations	**0.638**	**0.552**	**0.455**
KGC-RAG	awardWonBy	0.771	0.103	0.125
	companyTrades AtStockExchange	0.842	0.795	0.678
	countryLand BordersCountry	0.921	0.900	0.850
	personHas CityOfDeath	0.750	0.910	0.660
	seriesHas NumberOfEpisodes	0.725	0.700	0.697
	All Relations	**0.799**	**0.801**	**0.695**

Table 6
The results of our system on test set

Relation	macro-p	macro-r	macro-f1
awardWonBy	0.825	0.021	0.032
companyTradesAtStockExchange	0.797	0.755	0.638
countryLandBordersCountry	0.910	0.903	0.854
personHasCityOfDeath	0.695	0.920	0.647
seriesHasNumberOfEpisodes	0.800	0.770	0.770
All Relations	**0.792**	**0.810**	**0.698**

5.2. Knowledge Discrepancy

While performing the task, we observed that although the responses generated by the LLM were consistent with the context derived from Wikipedia, they did not align with the information referenced in Wikidata. This discrepancy affected the system's performance. For example, in the training data for the subject: *Love & Hip Hop: New York*, relation: *seriesHasNumberOfEpisodes*, the LLM predicted that the series had 143 episodes, based on Wikipedia. However, Wikidata indicated that the series had 82 episodes, which was outdated. Although both Wikipedia and Wikidata are open-source platforms where users can update information, the discrepancy in information between them still exists. Helping to update information or reporting issues to the

Table 7
The average coverage scores on validation set

Relation	Coverage
awardWonBy	0.137
companyTradesAtStockExchange	0.675
countryLandBordersCountry	0.967
personHasCityOfDeath	0.715
seriesHasNumberOfEpisodes	0.790

community could help reduce this discrepancy.

5.3. Relevant Context

High-quality context can improve the performance of question-answering in LLMs [9]. We employed LLMs and relevant scores to filter the quality of documents obtained from web scraping and web crawling. After consolidating documents into a single context, we evaluate the quality of the Relevant Context using the coverage score. The coverage score indicates how well the Relevant Context contains substrings that are object entities from the given subject entity and relation. It is calculated by dividing the number of object entity substrings found in the Relevant Context by the total number of object entities for that given subject entity and relation. The coverage scores for each relation on the validation set are shown in Table 7.

It was observed that for certain relations, particularly awardWonBy, the coverage score was not high. This may be due to the fact that Wikipedia often presents award winners in tables that include additional information. Although the award winner's name is present, the presence of other noise in the data can impact the relevant score [10, 11]. When comparing the coverage score with the macro F1 score for each relation in Table 5, it was found that the macro F1 score is generally lower or similar. This implies that increasing the coverage score of the context could potentially enhance the quality of responses generated by the LLM.

6. Conclusion

This study aims to construct Knowledge Graphs from large language model by using Retrieval-Augmented Generation (KGC-RAG) to optimize knowledge extraction from LLM and to involve the large language model in finding relevant documents using ISWC 2024 LM-KBC Challenge datasets. We achieved an F1 score of 0.695 for the validation set and 0.698 for the test set.

The results demonstrate that the quality of context is crucial for optimizing LLM performance in answering questions. Additionally, the LLM can effectively assist in screening relevant documents, which is a key factor in constructing an accurate and high-quality Knowledge Graph. For future investigations, it is recommended to explore the implementation of automatic relevant document retrieval instead of relying solely on question-answering combined with relevant scores.

References

[1] H. Paulheim, Knowledge graph refinement: A survey of approaches and evaluation methods, Semantic Web 8 (2017) 489–508.

[2] D. Vrandečić, M. Krötzsch, Wikidata: A free collaborative knowledge base, Communications of the ACM 57 (2014) 78–85. doi:10.1145/2629489.

[3] L. Jansen, R. van Hirtum, Constructing knowledge graphs from text: A survey of methods and tools, Journal of Computer Science and Technology 35 (2020) 1001–1020.

[4] R. Binns, V. Veitch, N. Shadbolt, Evaluating the reliability of large language models for knowledge extraction, in: Proceedings of the ACM Conference on Information and Knowledge Management (CIKM), 2021.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, in: Advances in Neural Information Processing Systems, volume 33, 2020, pp. 9459–9474. doi:10.48550/arXiv.2005.11401.

[6] B. Zhang, I. Reklos, N. Jain, A. M. Peñuela, E. Simperl, Using large language models for knowledge engineering (llmke): A case study on wikidata, 2023. URL: https://arxiv.org/abs/2309.08491. arXiv:2309.08491.

[7] X. Chen, J. Zhu, Enhancing context coverage for question answering with multiple knowledge sources, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, 2020, pp. 2340–2350. doi:10.18653/v1/2020.emnlp-main.182.

[8] H. Zhao, M. Eskenazi, Understanding and mitigating the impact of data source limitations on llm performance, in: Proceedings of the 2021 Conference on Neural Information Processing Systems (NeurIPS), 2021. doi:10.48550/arXiv.2110.11398, to appear.

[9] T. Kwiatkowski, J. Palomaki, M. Redfield, M. Edward, M. Collins, et al., Natural questions: a benchmark for question answering research, Transactions of the Association for Computational Linguistics 7 (2019) 453–466. doi:10.1162/tacl_a_00276.

[10] D. Khashabi, H. Hajishirzi, Learning to denoise knowledge in text-based question answering, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, 2021, pp. 4142–4154. doi:10.18653/v1/2021.emnlp-main.327.

[11] Y. Jernite, J. M., Analyzing the effects of noise on neural network performance in natural language processing, in: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, 2017, pp. 2892–2900. doi:10.18653/v1/d17-1291.