Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

An Embedding-assisted Prompt-engineering Approach for Translating Natural Language Text to Cypher Query Language using a Small-sized Large Language Model

Bruno Esteves Dalla Costa Filho

Monografia apresentada como requisito parcial
para conclusão do Bacharelado em Ciência da Computação

Orientador
Thiago de Paulo Faleiros

Brasília
2024

An Embedding-assisted Prompt-engineering Approach for Translating Natural Language Text to Cypher Query Language using a Small-sized Large Language Model

Bruno Dalla Costa
Departament of Computer Science
University of Brasília
Brasília, Brazil
brunoesteves.dcf@outlook.com

Andrei Lima Queiroz
Departament of Computer Science
University of Brasília
Brasília, Brazil
andreiqueiroz@unb.br

Thiago de Paulo Faleiros
Departament of Computer Science
University of Brasília
Brasília, Brazil
thiagodepaulo@unb.br

***Abstract*—This article explores the domain of natural language translation to Cypher Query Language within the context of graph databases, focusing on the task of Text-to-CQL conversion. Leveraging a small 7-billion-parameter language model, we investigate the impact of prompt-engineering techniques on language model performance. Our experiments reveal insights into the effectiveness of different approaches, including Execution-Based Self-Consistency and Embedding-Assisted Few-Shot prompting. Additionally, we introduce a small annotated dataset constructed from official publications of a Brazilian government gazette. The results demonstrate that the Embedding-Assisted approach significantly enhances the accuracy of the small model, achieving an average result of 55.36%. When combined with Execution-Based Self-Consistency, the approach showcases consistent improvements, leading to an average result of 58.69%. Moreover, a comparative analysis with a larger 70-billion-parameter model achieved a result of 48.88%, emphasizing the efficiency gains achievable with the smaller model. The findings underscore the significance of prompt engineering in enhancing language generation capabilities for Text-to-CQL tasks, providing valuable insights for natural language interactions with graph databases. The study contributes to the evolving field of natural language translation to query languages and guides prompt-engineering techniques for efficient and accurate interactions with graph databases.**

***Index Terms*—NLP, Text-to-CQL, Prompt Engineering, LLM**

I. Introduction

In order to retrieve stored information, many information systems and software applications interact with database management systems (DBMS) through a set of specific data query language (DQL) commands. For the majority of relational-based DBMS, the Structured Query Language (SQL) has been the dominant standard so far, although each DBMS provider may offer its type of DQL to the clients; for example, in Neo4j, a graph-based database employs its language called the Cypher Query Language (CQL). CQL language syntax differs from the traditional SQL standard and is used to query information related to nodes, properties, and relationships between nodes.

Recently, there has been an important use of Large Language Models (LLM) such as OpenAI GPT [1], Google BARD [2], and GitHub CoPilot [1] to assist in writing code snippets without prior knowledge in programming language. These models accept natural language inputs such as "How to write a recursive function in Python," or "How to write the bubble sort algorithm in Java" and output the respective code. Similarly, in the area of DBMS, a user wants to have a query that answers a specific question; for instance, he/she wants to know how to write a SQL query that brings in its results the answer to the question "What is the number of employees in company XPTO," so he/she would provide the following natural language input to the LLM: "Given the following database schema [SCHEMA], What is the SQL query to obtain the number of employees working at XPTO company?". Then the output should be similar to: "SELECT count(*) FROM employees WHERE company = 'XPTO'". However, these LLMs might produce suboptimal outputs without an ideal designed prompt [1].

Following this premise and also knowing that the freely available model called Llamma can achieve comparable results with the state-of-the-arte models in inference tasks [3], this work presents a study on several prompting engineering techniques to leverage the power of Llamma to produce CQL queries based on natural language input of a user, in other words, perform a Text-to-CQL.

To perform this experiment, we used a database constituted by publications of the Official Gazette of the Federal District and constructed by the DODFMiner project [4], [5], in which the data is stored in a graph DBMS and is accessible through CQL query language. With this data, we create a dataset that contains CQL queries that answer the most common question of the users of this system. Thus, we could evaluate the accuracy of an LLM in combination with prompt engineering techniques to perform translation of the natural language question to CQL query.

The main contributions of this work are:

- A small annotated dataset containing natural language questions and respective CQL queries.

- A Few-Shot approach for LLMs in-context learning on Text-to-CQL.
- Exploration of Execution-Based Self-Consistency on Text-to-CQL alongside a proposed Embedding-Assisted Few-Shot method.

This paper is structured as follows. Section II details the related works in this line of study. Section III describes the proposed method and its constituting steps. Section IV relates the results obtained. Sections V, VI, and VII conclude this paper and discuss possibilities for future work.

II. Related work

While exploring related works in natural language conversion to query languages has emphasized the crucial role of LLMs adaptation, mainly through techniques like Few-Shot prompting [1] and fine-tuning, we identified a gap in the literature. This gap results from the absence of extensive annotated datasets, impeding the execution of experiments and fine-tuning processes tailored explicitly for Text-to-CQL tasks. Despite this gap, the author of [6] presents a significant contribution in this context. They introduced SpCQL, a dataset featuring 10,000 manually annotated natural language questions and corresponding Cypher queries, and proposed a Seq2Seq framework baseline on this newly curated corpus. Notably, it is essential to mention that the dataset is predominantly in Chinese. Although their results may not have been satisfactory, achieving only 2.6% as the best result, their work shed light and inspired research opportunities for the Text-to-CQL task.

Given the relatively unexplored Text-to-CQL as a research area, this study briefly introduces the Text-to-SQL task. We explore some works that implemented Fine-Tuning and Few-Shot approaches using extensive datasets. WikiSQL [7] and Spider [8] are the most well-known datasets on which state-of-the-art models base their experiments. A significant distinction between the two is that WikiSQL is a dataset designed for single-table queries, whereas Spider encompasses queries involving multiple tables using Joins. However, excluding SpCQL [6], we did not find any dataset tailored explicitly for the Cypher language of Neo4j.

Experiments aim for more realistic scenarios focusing on the Spider dataset for their Seq2Seq model training. One of the experiments achieving high results on the dataset is DIN-SQL [9], which employed a four-step approach to tackle the query translation problem. These steps include Schema linking, Query classification and Decomposition, SQL Generation, and Self Correction. Query Classification and Decomposition are stages in which prompt-engineering techniques are applied to guide the LLM to follow a logical analysis and decomposition of the natural language question into segments of the SQL query. The self-correction mechanism represents an additional step where the generated query is fed back into an LLM to verify potential errors.

Following a similar line of study, [10] proposed Few-Shot SQL-PaLM and Fine-Tuned SQL-PaLM, a prompt design and a fine-tuning approach that leverages the PaLM-2 model [11] as its foundation. This work pushed the state-of-the-art in both Few-Shot an Fine-Tuning approaches and represents a milestone in the field, achieving an impressive 77.3% accuracy on the Spider test suite with Few-Shot SQL-PaLM, significantly surpassing the prior state-of-the-art in fine-tuning settings by 4%. Furthermore, the results demonstrate that the fine-tuned SQL-PaLM outperforms this benchmark by an additional 1%, underscoring its effectiveness in natural language to SQL conversion tasks.

Furthermore, building upon the advancements in Text-to-SQL parsing, the Graphix-T5 [12] model emerges as the current state-of-the-art, marking a significant stride in the domain. It extends the capabilities of the T5 [13] pretrained text-to-text transformer and proposed the GRAPHIX-T5 architecture, integrating specially designed graph-aware layers with the standard pre-trained transformer model.

While the Graphix-T5 model stands as the current pinnacle in Text-to-SQL parsing, it is essential to acknowledge the trade-offs associated with its achievement. Graphix-T5 leverages the strategy of fine-tuning, a process known for its remarkable effectiveness in tailoring pre-trained models to specific tasks. However, fine-tuning comes with inherent challenges, notably its high computational cost and susceptibility to overfitting the training data. The extensive customization involved in fine-tuning makes it less adaptable to new data and poses challenges in scenarios where computational resources are constrained.

In contrast to fine-tuning, there is the Few-Shot prompt engineering approach, such as in DIN SQL [9] and Few-Shot SQL-PaLM [10]. This approach, compared to fine-tuning [14] [15], eliminates the need for a dedicated training phase, resulting in significantly lower computational requirements. This characteristic is precious in Text-to-SQL parsing, where SQL encompasses diverse dialects. Additionally, DIN SQL and Few-Shot SQL-PaLM's adaptability to new data make them good choices for the Text-to-SQL task, as they offer ease of integration with the evolving language patterns, syntax, and variations in SQL. The absence of a dedicated training phase makes these approaches highly efficient, underscoring their suitability for scenarios where the computational resource is not abundant.

The Few-Shot prompting in LLMs, known as in-context learning, was initially identified in [1]. By incorporating a limited number of demonstrations, accompanied by instructions, within the prompt text to establish a contextual basis, LLMs have demonstrated the ability to generalize to novel examples and tasks in the same format without necessitating model adaptation. [16] reveals that the effectiveness of Few-Shot prompting is more pronounced in LLMs exceeding a certain parameter size threshold. The notable success of in-context learning has sparked the development of advanced prompting strategies, such as Chain-of-Thought prompting (CoT) [17]. Unlike Few-Shot prompting, CoT involves constructing a logical sequence of thoughts within the model's prompt. This approach aims to guide the model in generating responses with a coherent and deeper reasoning structure. CoT enables the model to maintain logical consistency throughout its gener-

ated output by chaining concepts together strategically. This technique mainly benefits tasks requiring intricate inferential reasoning and a more profound understanding of contextual information.

In reviewing the related works, many strategies and methods discussed have primarily been tested in the context of Text-to-SQL tasks. The existing articles have demonstrated progress in enhancing natural language understanding and conversion to SQL queries. Therefore, based on the mentioned works, this article extends the current understanding by investigating the effectiveness of the established strategies in Text-to-SQL when applied to the task of Text-to-CQL.

III. Methodology

This section outlines the experimental methodology. Subsection III-A details the various approaches employed to address the Text-to-Cypher (Text-to-CQL) task, introducing the Embedding-Assisted Few-Shot prompt engineering method and evaluating its effectiveness in the conducted experiments, revealing the gains achieved in the task. Subsection III-B describes the dataset utilized for the experiments and outlines its creation process. Subsection III-C discusses the chosen language model.

A. Experimental Approaches

We adopted a combination of prompt-engineering techniques to address the Text-to-cipher task, as summarized in Table I. Notably, the Few-Shot method was the foundation for all experiments, leveraging a validated approach from SQL PaLM [10]. This method incorporated database information, including Nodes, Properties, and Relationships pertinent to the Neo4j database. A set of five demonstrations, consisting of pairs of questions in natural language and their corresponding queries, was included to provide context. The format of the input context in the Few-Shot method is expressed as follows:

$$\text{SCHEMA} + \text{LIST OF DEMONSTRATIONS}_{[5]}. \quad (1)$$

TABLE I: Method Combinations Employed in Text-to-CQL

	Few-Shot	**Execution-Based Self-Consistency**	**Embedding-Assisted Few-Shot**
FS Baseline	X		
FS + EBSC	X	X	
FS + EA	X		X
FS + EA + EBSC	X	X	X

The temperature configuration in LLMs applied to tasks involving code or data is typically set to low values to produce less creative and imaginative text [18], [19]. We systematically varied hyperparameter values for temperature during text generation for each experiment, ranging from 0.1 to 1.5. This exploration aimed to assess the model's sensitivity to different levels of randomness with each prompt-engineering technique, providing insights into how temperature changes influence the quality and variability of the model's output in Text-to-CQL.

1) Execution-Based Self-Consistency (EBSC) Few-Shot: This approach focused on leveraging EBSC [10] to refine the model's output. The Execution-Based [20], [21] and Self-Consistency [22] methods, collectively known as Execution-Based Self-Consistency, were employed. This approach, illustrated in Fig. 1, aims to make the model generate its answer multiple times, enhancing diversity. Outputs with failed execution are discarded, and those with the lowest occurrence are excluded, assuming the highest occurrence corresponds to the correct answer.

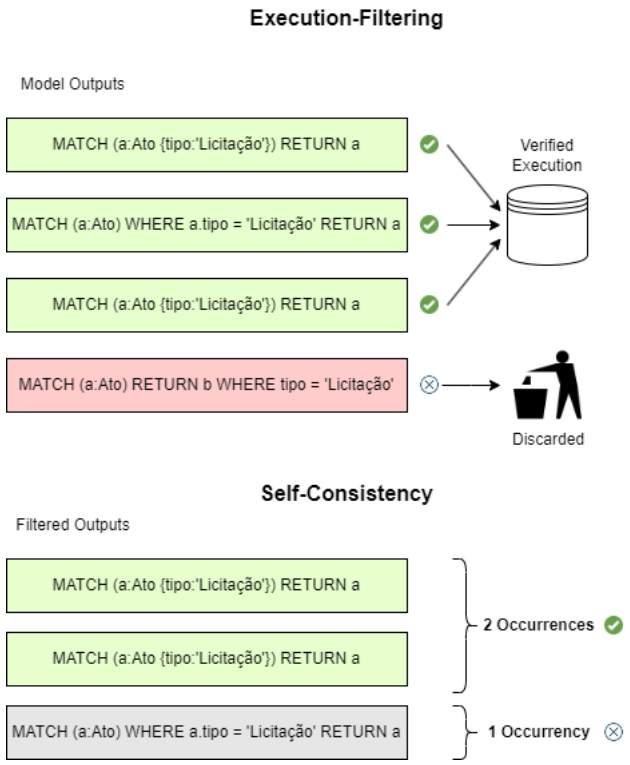

Fig. 1: Execution-Based Self-Consistency

2) Embedding-Assisted (EA) Few-Shot: In the realm of NLP, semantic similarity [23] is a crucial task that evaluates the association between texts or documents through a specified metric. We proposed the EA prompt engineering method, using the Multilingual MPNET Base embeddings [24] alongside the application of cosine similarity [25]. This approach creates a set of contextual examples from other questions closely aligned with the meaning of the target question, enhancing the model's ability to generalize.

3) Embedding-Assisted Execution-Based Self-Consistency (EA+EBSC) Few-Shot: This experiment integrated Embedding-Assisted Few-Shot and Execution-Based Self-Consistency, aiming to combine the strengths of each method for improved language generation and assessing their performance together.

B. Dataset

Government gazettes are official documents to disseminate authoritative announcements, decisions, and activities to the public. These publications offer a wealth of information, encompassing various topics such as actions about civil servants (appointments, dismissals, replacements, etc.), bidding processes, contract statements, and other pertinent subjects involving the utilization of public resources [26]. The Federal Government and all Brazilian federative units consistently publish gazettes daily in Brazil. One of these federative units, the Official Gazette of the Federal District (DODF - Diário Oficial do Distrito Federal, in Portuguese), has been in circulation since 1960. The editions of DODF from October 1967 to April 2020 exist solely in PDF format, while those published since May 2020 are also accessible in JSON format. Notably, the DODF is organized into segments known as acts, which are further categorized into three sections. This study specifically concentrates on the third section of the document, which comprises information about contracts and public bidding process.

To address questions concerning the third section of the Official Gazette, we curated a small dataset comprising 60 questions with their respective CQL queries. These questions cover users' main inquiries regarding acts of contracts, bidding processes, and their respective life-cycle [5]. The queries were generated in an equal ratio based on the number of relationships used, as illustrated in Table II, comprising 20 items with zero relationships, 20 items with one relationship, and 20 items with two relationships. This variation in complexity within the Few-Shot context aims to accommodate the increasing complexity associated with more relationships in queries.

Additionally, it is important to grasp the distinction between queries involving zero, one, or two relationships in Cypher. When a query is constructed without the inclusion of relationships (*zero relationships*), it focuses on specific node properties without considering connections to other elements in the graph. For instance, a question regarding the count of bidding acts can be mapped to a query without relationships, as illustrated below:

```
Question: Quantos atos são de licitação?

MATCH (a:Ato)
WHERE a.tipo = 'Licitação'
RETURN COUNT(a)
```

On the other hand, queries incorporating one relationship (*one relationship*) are designed to explore direct connections between entities in the graph. Taking an example of a question about acts belonging to a specific organization, it generates a one-relationship query:

```
Question: Quais atos pertencem ao órgão 'Casa Civil'?

MATCH (a:Ato)-[:PERTENCE]->(o:Orgao)
WHERE o.orgao = 'Casa Civil'
RETURN a
```

Finally, queries involving two relationships (*two relationships*) further enhance complexity, allowing for deeper investigations into patterns of connections. For instance, when inquiring about the number of processes linked to a specific organization, the following two-relationship query can be formulated:

```
Question: Quantos processos estão vinculados ao órgão
          'Tribunal de Justiça'?

MATCH (a:Ato)-[:POSSUI]->(p:Processo),
      (a)-[:PERTENCE]->(o:Orgao)
WHERE o.orgao = 'Tribunal de Justiça'
RETURN COUNT(p)
```

TABLE II: Dataset Description Grouped by Number of Relationships in the Cypher Queries (Zero, One, Two)

Relationships	Zero	One	Two	Total
Instances	20	20	20	60

C. Models

Due to limited access to computational resources necessary for large-scale models with tens of billions of parameters, we opted for a smaller model. This decision introduced an additional challenge to the experiment, as model performance is observed to be proportional to the number of parameters [16]. Specifically, we utilized the Llama 2 7b model from the META organization [3], consisting of 7 billion parameters. Additionally, we compared results with the 70 billion parameters Llama 2 available in the Petals [27] collaborative system for inference and fine-tuning of large models.

TABLE III: Comparison of Llama 2 7b and Llama 2 70b Models

Model Specification	Llama 2 7b	Llama 2 70b
Number of Parameters	7×10^9	70×10^9
Context Length	4k tokens	4k tokens
Training Tokens	2 trillion	2 trillion
Learning Rate	3.0×10^{-4}	1.5×10^{-4}
Grouped-Query Attention (GQA)	No	Yes

IV. Results

The results section addresses the evaluation and analysis of the outcomes obtained in our study. The Subsection IV-A highlights the chosen evaluation metric for measuring the model's performance. In Subsection IV-B, we delve into result analysis, interpreting and contextualizing the findings to unveil insights into the results.

A. Evaluation

This subsection presents the methodology for assessing the model's performance in each experiment. The chosen performance indicator is the Execution Accuracy, which compares the results obtained by executing the query generated by the model with those derived from the execution of the "Ground Truth" query. The formula for Execution Accuracy (ACC) is defined as:

$$ACC = \frac{\text{Number of Execution Results Identical to Ground Truth}}{\text{Total Items in the Dataset}} \quad (2)$$

This formula quantifies the model's accuracy by determining the ratio of execution results that match the Ground Truth to the total number of items in the dataset.

B. Result Analysis

In this subsection, we investigate the impact of the Baseline, EBSC, EA, and EA+EBSC methods on the Llama 2 7b model. Using the ACC metric, we assess the performance of each method and evaluate how temperature variations can influence their results. Table IV presented the model performance for each approach in a 0.1, 0.3, and 0.5 temperature range.

The baseline Few-Shot (FS) approach with the Llama 2 7b model achieved the lowest average ACC result at 22.21%, and exhibited varying performance at different temperatures. The highest result was achieved at a temperature of 0.1 with 24.44%. The performance declined with higher temperatures, indicating sensitivity to the temperature parameter.

Utilizing FS+EBSC improved 1.11% in the average ACC performance. At a temperature of 0.3, it achieved its highest result with 25.55%. This method alone proved ineffective in refining the model's output, contributing to an Average TS result of 23.32%.

The EA method significantly enhanced the baseline performance at 33.15%. At a temperature of 0.1, the ACC result reached 57.22%, showcasing the effectiveness of leveraging embeddings and cosine similarity to generate contextually relevant examples. The average ACC result across all temperatures was 55.36%.

TABLE IV: ACC Results of Different Prompt-Engineering Methods in the Llama 2 7b Model

Method	Temperature 0.1	Temperature 0.3	Temperature 0.5	Avg ACC
FS Baseline	24.44%	23.88%	18.33%	22.21%
FS + EBSC	19.99%	25.55%	24.44%	23.32%
FS + EA	57.22%	54.99%	53.88%	55.36%
FS + EA + EBSC	58.88%	61.66%	55.55%	58.69%

Combining EA + EBSC yielded further improvements in the Llama 2 7b model's performance by 36.48% compared to the baseline, indicating a successful combined effect between the two methods. Notably, at a temperature of 0.3, the ACC reached 61.66%. The average ACC across all temperatures was 58.69%, the highest across all the previous. These results indicate the effectiveness of prompt-engineering techniques, notably EA + EBSC, in enhancing the Llama 2 7b model's language generation capabilities for the Text-to-Cypher task.

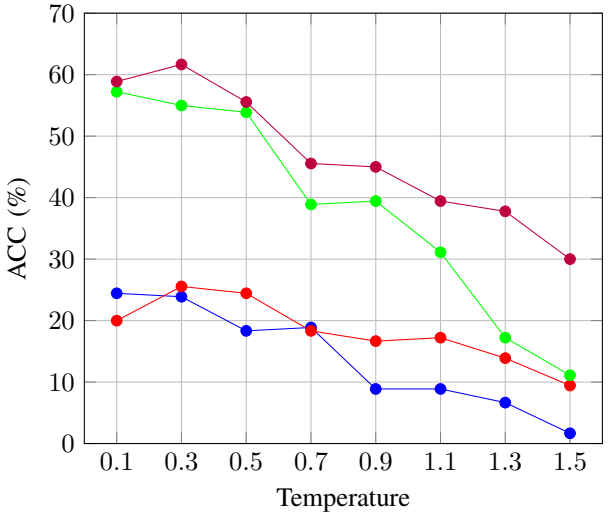

Fig. 2: TS Results with Temperature Variation.
Blue: FS, Red: EBSC, Green: EA, Purple: EA+EBSC

C. Temperature Range Analysis

Moreover, Figure 2 illustrated that all experiments had their worst results at 1.5 temperature. This result supports the choice in this and other studies that opted for low temperatures for tasks involving code generation [18], [19]. Fig. 3 demonstrates how the ACC values dropped when comparing temperatures of 0.1 and 1.5. With a temperature of 1.5, the baseline FS achieved 1.67% ACC, the lowest value. All other methods were also heavily impacted at 1.5 temperature, with EBSC at 9.44%, EA at 11.11%, and EA+EBSC at 30%.

The Baseline FS had the most significant reduction at 1.5 temperature, reaching approximately 6% of its value at 0.1 temperature. The EA method also experienced a significant decline in ACC, reaching only 19% at 1.5 temperature compared to its result at 0.1. Methods incorporating EBSC showed more resilience to the temperature increase to 1.5, with EBSC reaching approximately 47% and EA+EBSC reaching 51% of their results at 0.1 temperature. This demonstrates that even with a more random model at a higher temperature, EBSC can maintain a sure consistency in outputs, tending towards more desirable result choices.

D. Larger Model Comparison

A comparative analysis was conducted by employing the baseline method with a larger Llama 2 model containing 70 billion parameters. The outcomes were compared to those of the 7-billion-parameter model, providing insights into the effectiveness of prompt-engineering techniques across models of different scales as demonstrated at 4. EA + EBSC improved by 9.81% over the Llama 2 70b Few-Shot method, underscoring the importance of considering computational resources when choosing the model size, as the smaller model variants can still achieve competitive performance with appropriate prompt engineering.

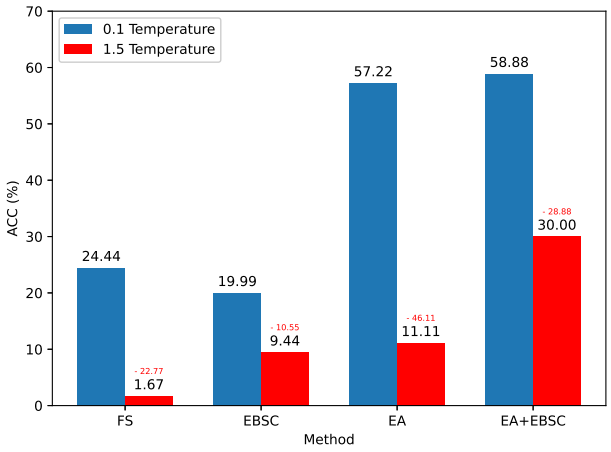

Fig. 3: ACC Comparison in 0.1 and 1.5 Temperatures

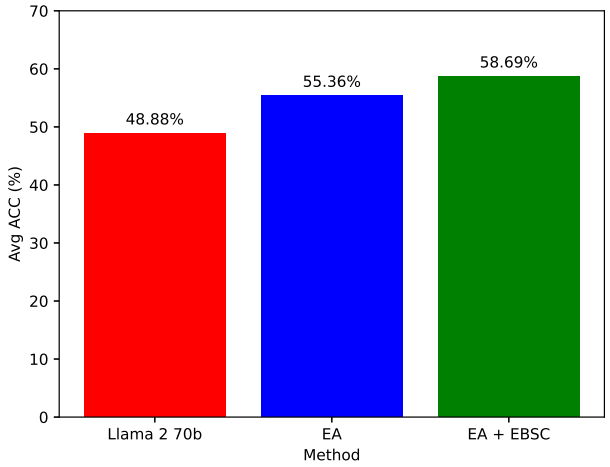

Fig. 4: Llama2 Few-Shot comparison with Embedding-Assisted Few-Shot based methods

V. Conclusion

This work emphasized the need for improved database accessibility and eliminating the requirement for users to master query language syntax. The study focused on overcoming these challenges through innovative approaches, primarily emphasizing the Embedding-Assisted Few-Shot method.

The Embedding-Assisted Few-Shot approach emerged as a key contribution, demonstrating its effectiveness. This technique leverages embeddings and cosine similarity to generate contextually relevant examples, resulting in a remarkable performance boost. The experiments validated the use of the Embedding-Assisted Few-Shot method and underscored its potential to enhance language model capabilities in Text-to-Cypher tasks.

Furthermore, the comparative analysis with the Llama 2 70b model revealed that competitive results could be achieved through appropriate prompt engineering even with less powerful hardware. This finding emphasizes the importance of considering computational resources when selecting model sizes and highlights the efficiency gains achievable with the proper techniques.

Notably, the study demonstrated the practical applicability of Text-to-Cypher scenarios in a real-world context—specifically. The potential use cases include applications such as chatbots or question-answering systems, facilitating user access to database information without requiring intricate query syntax.

In summary, this work not only tackled the initial challenges posed in the introduction but also provided potential solutions and valuable insights. The Embedding-Assisted Execution-Based Self-Consistency Few-Shot method emerged as a powerful combination for achieving effective and efficient language generation in Text-to-Cypher tasks. The demonstrated applicability in a real-world scenario further reinforces the significance and relevance of this study in advancing the field.

VI. Limitations

Datasets for Text-to-CQL are scarce, posing a challenge for experiments in this research domain. Also, the model selection introduces another challenge, given that the dataset questions are in Portuguese. Pre-trained Large Language Models (LLMs) are primarily trained on English data, resulting in significantly better performance on tasks specified in English. For instance, the Llama 2 model [3] was trained with 89.70% of the data in English, with only 0.09% of the data in Portuguese.

Furthermore, it is essential to acknowledge that the size of our dataset, while representative of real user queries, may be considered small. This limitation could impact the model's ability to generalize across a broader range of scenarios.

VII. Future Work

For future research stemming from this study, we can derive inspiration to curate a more expansive dataset. This augmented dataset aims to cover a spectrum of diverse databases and include variations in language, thereby fostering a more comprehensive and multilingual environment.

We can adopt a fine-tuning methodology by utilizing this enriched dataset and harnessing ample computational resources. The goal of this approach is to refine a Text-to-CQL domain-specific LLM. The fine-tuning process ensures a targeted adaptation, optimizing the LLM's performance and enhancing its proficiency in handling the complexities inherent in CQL queries.

Furthermore, we can explore these approaches using larger models like GPT-4 [28] and BARD [2]. Assessing these advanced models helps understand their scalability and provides insights into their effectiveness in handling complex language nuances in the context of Text-to-Cypher queries. This comprehensive exploration contributes to a better understanding of their practical applicability and potential trade-offs in real-world scenarios.

REFERENCES

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[2] R. Thoppilan, D. D. Freitas, J. Hall, N. Shazeer, A. Kulshreshtha, H.-T. Cheng, A. Jin, T. Bos, L. Baker, Y. Du, Y. Li, H. Lee, H. S. Zheng, A. Ghafouri, M. Menegali, Y. Huang, M. Krikun, D. Lepikhin, J. Qin, D. Chen, Y. Xu, Z. Chen, A. Roberts, M. Bosma, V. Zhao, Y. Zhou, C.-C. Chang, I. Krivokon, W. Rusch, M. Pickett, P. Srinivasan, L. Man, K. Meier-Hellstern, M. R. Morris, T. Doshi, R. D. Santos, T. Duke, J. Soraker, B. Zevenbergen, V. Prabhakaran, M. Diaz, B. Hutchinson, K. Olson, A. Molina, E. Hoffman-John, J. Lee, L. Aroyo, R. Rajakumar, A. Butryna, M. Lamm, V. Kuzmina, J. Fenton, A. Cohen, R. Bernstein, R. Kurzweil, B. Aguera-Arcas, C. Cui, M. Croak, E. Chi, and Q. Le, "Lamda: Language models for dialog applications," 2022.

[3] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, "Llama 2: Open foundation and fine-tuned chat models," 2023.

[4] G. M. Guimarães, F. X. da Silva, A. L. Queiroz, R. M. Marcacini, T. P. Faleiros, V. R. Borges, and L. P. Garcia, "Dodfminer: An automated tool for named entity recognition from official gazettes," *Neurocomputing*, vol. 568, p. 127064, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231223011876

[5] I. Ferreira, L. Lopes, T. de Paulo Faleiros, L. Garcia, V. Borges, A. L. Queiroz, and L. Mota, "A tool for retrieving the life-cycle of public procurement processes," in *Proceedings of the 18th Iberian Conference on Information Systems and Technologies - CISTI 2023*, 2023.

[6] A. Guo, X. Li, G. Xiao, Z. Tan, and X. Zhao, "Spcql: A semantic parsing dataset for converting natural language into cypher," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, ser. CIKM '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 3973–3977. [Online]. Available: https://doi.org/10.1145/3511808.3557703

[7] V. Zhong, C. Xiong, and R. Socher, "Seq2sql: Generating structured queries from natural language using reinforcement learning," 2017.

[8] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. Radev, "Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task," 2019.

[9] M. Pourreza and D. Rafiei, "Din-sql: Decomposed in-context learning of text-to-sql with self-correction," 2023.

[10] R. Sun, S. O. Arik, H. Nakhost, H. Dai, R. Sinha, P. Yin, and T. Pfister, "Sql-palm: Improved large language model adaptation for text-to-sql," 2023.

[11] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Abrego, J. Ahn, J. Austin, P. Barham, J. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni, A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. C. Ros, A. Roy, B. Saeta, R. Samuel, R. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, and Y. Wu, "Palm 2 technical report," 2023.

[12] J. Li, B. Hui, R. Cheng, B. Qin, C. Ma, N. Huo, F. Huang, W. Du, L. Si, and Y. Li, "Graphix-t5: Mixing pre-trained transformers with graph-aware layers for text-to-sql parsing," 2023.

[13] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," 2023.

[14] T. Scholak, N. Schucher, and D. Bahdanau, "Picard: Parsing incrementally for constrained auto-regressive decoding from language models," 2021.

[15] J. Qi, J. Tang, Z. He, X. Wan, Y. Cheng, C. Zhou, X. Wang, Q. Zhang, and Z. Lin, "Rasat: Integrating relational structures into pretrained seq2seq model for text-to-sql," 2022.

[16] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent abilities of large language models," 2022.

[17] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023.

[18] J. Li, Y. Zhao, Y. Li, G. Li, and Z. Jin, "Acecoder: Utilizing existing code to enhance code generation," 2023.

[19] S. Fakhoury, S. Chakraborty, M. Musuvathi, and S. K. Lahiri, "Towards generating functionally correct code edits from natural language issue descriptions," 2023.

[20] X. Chen, M. Lin, N. Schärli, and D. Zhou, "Teaching large language models to self-debug," 2023.

[21] F. Shi, D. Fried, M. Ghazvininejad, L. Zettlemoyer, and S. I. Wang, "Natural language to code translation with execution," 2022.

[22] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," 2023.

[23] S. Harispe, S. Ranwez, S. Janaqi, and J. Montmain, *Semantic Similarity from Natural Language and Ontology Analysis*. Springer International Publishing, 2015. [Online]. Available: http://dx.doi.org/10.1007/978-3-031-02156-5

[24] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019.

[25] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," 2014.

[26] E. Passos, "Doing legal research in brazil," 2002. [Online]. Available: https://www.nyulawglobal.org/globalex/Brazil.html

[27] A. Borzunov, D. Baranchuk, T. Dettmers, M. Ryabinin, Y. Belkada, A. Chumachenko, P. Samygin, and C. Raffel, "Petals: Collaborative inference and fine-tuning of large models," 2023.

[28] OpenAI, "Gpt-4 technical report," 2023.

VIII. Supplementary Material

A. *Few-Shot Prompt Example*

Esse é um schema de um banco Neo4j orientado à grafos de uma base de dados do Diário Oficial do Distrito Federal. O schema possui Nós, Relacionamentos.

SCHEMA

Nós:

```
Ato {secao: STRING, texto: STRING,
tipo: STRING, titulo: STRING}
Orgao {orgao: STRING}
Processo {processo: STRING}
Data {data: STRING}
```

Relacionamentos:

```
(Ato)-[:PERTENCE]->(Orgao)
(Ato)-[:POSSUI]->(Processo)
(Ato)-[:PUBLICADO]->(Data)
```

FIM SCHEMA

O objetivo da tarefa é traduzir as perguntas de um usuário sobre a base de dados para uma query Cypher que vai consultar o banco Neo4j.

Observações:
- As datas são strings, então para fazer ordenação ou comparação precisamos usar funções da biblioteca APOC do Neo4j.
- A timeline de um processo é o conjunto de atos que estão ligados por um mesmo processo ordenados pela data de publicação.

Aqui vai um exemplo de como traduzir uma pergunta para uma query:

EXEMPLO

Quais datas distintas são de 2023?

QUERY:

```
MATCH
(d:Data)
WHERE d.data CONTAINS '2023'
RETURN DISTINCT d
```

FIM QUERY

FIM EXEMPLO

Aqui vai um exemplo de como traduzir uma pergunta para uma query:

EXEMPLO

Quais processos ligam pelo menos 30 atos?

QUERY:

```
MATCH
(a:Ato)-[:POSSUI]->(p:Processo)
WITH COUNT(a) as quantidade, p
WHERE quantidade >= 30
RETURN p
```

FIM QUERY

FIM EXEMPLO

Aqui vai um exemplo de como traduzir uma pergunta para uma query:

EXEMPLO

Quais atos de 'Decisão' estão vinculados ao órgão 'Tribunal de Justiça'?

QUERY:

```
MATCH
(a:Ato)-[:PERTENCE]->(o:Orgao)
WHERE a.tipo = 'Decisão' AND o.orgao = 'Tribunal de Justiça'
RETURN a
```

FIM QUERY

FIM EXEMPLO

Aqui vai um exemplo de como traduzir uma pergunta para uma query:

EXEMPLO

Que atos da 'Seção iii' do tipo 'Aviso' possuem o termo 'inexigibilidade' em seu título?

QUERY:

```
MATCH
(a:Ato)
WHERE a.secao = 'Seção iii' AND a.tipo = 'Aviso' AND a.titulo
CONTAINS 'inexigibilidade'
RETURN a
```

FIM QUERY

FIM EXEMPLO

Aqui vai um exemplo de como traduzir uma pergunta para uma query:

EXEMPLO

Que atos são de Portaria?

QUERY:

```
MATCH
(a:Ato)
WHERE a.tipo = 'Portaria'
RETURN a
```

FIM QUERY

FIM EXEMPLO

Agora vou te dar uma pergunta e você vai me gerar a query Cypher que traduz essa pergunta, levando em consideração o Schema que foi passado e as observações feitas sobre a base de dados.

PERGUNTA

Quantos atos são de licitação?

QUERY: