An LLM-based Q&A Natural Language Interface to Process Mining

Luciana Barbieri[1][0000−0002−5295−2228], Kleber Stroeh[2], Edmundo R. M. Madeira[1][0000−0002−1180−2637], and Wil M. P. van der Aalst[3][0000−0002−0955−6940]

[1] Institute of Computing, University of Campinas, Campinas, Brazil
{luciana.barbieri,edmundo}@ic.unicamp.br
[2] Pegasystems, São Paulo, Brazil
kleber.stroeh@pega.com
[3] RWTH Aachen University, Aachen, Germany
wvdaalst@pads.rwth-aachen.de

Abstract. Process Mining has come a long way to meet the needs of organizations that must optimize their operations. However, its use is still driven by technical users who can interpret process maps, models, graphs and other types of analyses. Business users, on the other hand, frequently report being intimidated by Process Mining tools' interfaces and not knowing "what to do next". An alternative to address this issue is providing more fluid and friendly interfaces for non-technical users based on natural language querying. Recent advances in Large Language Models (LLMs) have expanded the horizon for such interfaces. In this work we propose a new strategy to combine LLM capabilities with a framework for a natural language question-and-answer interface to Process Mining, which combines the flexibility of the former with the scalability and precision of the latter. We expand upon previous works in the area to research the dimensions of flexibility, generalization, scalability and precision. Finally, we implement such an LLM-enhanced framework and test it against a real-life compilation of questions to compare the performance of LLM-based, non LLM-based and hybrid implementations and point to directions in this field of research.

Keywords: Process Mining · Process Querying · Natural Language Interface · Large Language Models.

1 Introduction

Process Mining has evolved into a mature discipline with deep impact in organizations worldwide. According to Markets and Markets, it is expected to reach a value of USD 12.1 billion by 2028 at a compound annual growth rate (CAGR) of 45.6% [13]. This growth could be further accelerated if business users joined the forces of technical users in leveraging Process Mining technologies in their daily operations. However, they often report difficulties in using the technology, citing challenges in making sense of process maps, dashboards and other representations used by tools.

To address these difficulties, we have previously proposed a framework for a natural language interface to Process Mining tools, such that non-technical users could seize its value through questions and answers [3,2]. While other related work generally mapped Process Mining natural language questions to queries over event log data, our previous method translated these questions to logical queries that ran against existing Process Mining tools, so as to leverage the mature algorithms and techniques they provide. The method, however, applied deterministic approaches for question understanding and mapping, such as rule-based parsing, and failed short in dealing with completely new, unpredicted questions.

The recent advances in Large Language Models (LLMs), such as GPT-4 [14], have uncovered new possibilities to dealing with more open questions and expanded, therefore, the horizons of natural language-based interfaces. Commercial Process Mining tools (e.g., Celonis, SAP Signavio, Microsoft Power Automate, Mindzie, Software AG, Pegasystems, etc.) started to offer co-pilots based on such models, which shows the relevance of the topic.

In this work, we explore the integration of LLMs (GPT-4 in particular) into our previously proposed framework to overcome these generalization limitations by creating an alternative to the rule-based parser in a Process Mining question-and-answer interface.

Our approach seeks to combine the flexibility of LLMs with the power and scalability of existing Process Mining tools. Therefore, we refrain from relying entirely on LLM technologies to answer questions, since such an approach does not take advantage of valuable, dedicated Process Mining algorithms and techniques implemented in existing academic and commercial tools [1]. Furthermore, the use of LLM technologies alone brings severe limitations associated with token limits that would render them useless in bigger data scenarios, commonly found in real life.

More specifically, we (1) build upon our previous work to propose a reviewed architecture to a question-and-answer interface that leverages LLM technology for semantic parsing, (2) use the Process Mining question taxonomy proposed in [2] as the underlying basis to compose a thorough prompt to interact with the LLM, (3) implement and test the proposed architecture against a real-life database of Process Mining-related questions and (4) explore the potential of hybrid approaches that combine rule-based and LLM-based parsers in such an architecture.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the proposed reviewed architecture. Section 4 presents the conducted experiments and results. Section 5 concludes this paper and points out future work and directions.

2 Related Work

This section reviews related work.

2.1 Early Natural Language Interfaces for Process Mining

Earlier works in the use of natural language question answering as an interface for Process Mining have mainly relied on rule-based or machine learning approaches.

Han and other researchers at IBM identified the need for a more friendly interface to query event data about process automation execution [8]. Their work introduces the generation of an ontology on the domain of the problem that is fed into a rule-based Natural Language Interface to Databases (NLIDB) system called ATHENA, so that natural language questions could trigger queries on the process automation data.

Kobeissi et al. also propose a natural language interface for querying event data [12]. They approach it through label property graphs, that explores a graph database. The interpretation of natural language is done through a mix of machine learning and rule based approaches. The method was extended in [11] to support process behavioral queries (e.g. questions related to the sequence of executed activities).

We also approached this problem through deterministic methods previously. We started by introducing a reference architecture for a query interface to Process Mining, as well as an abstract logical representation for Process Mining queries [3]. This combination helped convert natural language questions into executing steps against a generic interface to Process Mining APIs. Later, we extended this work by introducing a taxonomy on Process Mining questions to extend the reach of the queries supported by the interface [2]. In both works, the conversion from natural language to the abstract logical representation was mainly supported by a rule based approach. Real life questions were collected from Process Mining practitioners and tested against the implementation.

All these works share, in some degree, a common limitation, which is the ability to support more generic – almost colloquial – questions, commonly used in real life.

2.2 LLM-based Natural Language Interfaces for Process Mining

Recent developments in Generative Artificial Intelligence (GenAI) and LLMs have sparked new discussions around natural language processing in general, and, more specifically, Process Mining interfaces. Works that explore the use of LLMs to answer Process Mining questions can be categorized under the following general approaches [4].

Direct Answering Works analyzed under this category summarize event data or process mining artifacts into text and feed it to the LLM to answer questions directly.

Pioneering works under this category include Berti et al.'s [5] exploration of GenAI capabilities of prompting direct questions or hypothesis against Process Mining inputs such as discovered Petri Nets, Direct-follows Graph (DFG) abstractions and variant information. In this work, Process Mining data is given as input to the LLM, where the processing/reasoning takes place.

Under a similar approach, Kermani et al. [10] propose an architecture for the integration of Process Mining analyses and LLM technologies, where results from the first (dashboards, process discovery, conformance checking, performance mining, organizational mining) are fed into the second through proper prompt engineering to derive interpretations and recommendations. Special attention is given to providing an optimized prompt structure to accompany the data that is processed by the LLM. Similarly to [5], working on the outcomes of previous Process Mining algorithms (discovery, conformance checking, etc) keeps the number of tokens under control when interacting with the LLM, but limits, however, some of its application in real life scenarios, as the analyses need to take place before the use of LLMs.

Logical Representation Generation Approaches under this category provide metadata (event log metadata, process ontologies, etc.) to an LLM to generate a logical representation, such as a structured query or executable program, corresponding to a question.

This is the case of the method proposed by Jessen et al. [9], which presents an architecture that combines metadata, ontology and LLM capabilities to translate questions into Structured Query Language (SQL) queries on an event log database. They also explore orchestrating calls through Chain-of-Thought, using zero- and few-shot learning and benchmarking results against previous works. It is unclear, however, how the handling of more specific Process Mining functionality such as process discovery or conformance checking is done without further integration into Process Mining tools.

Similarly, our proposed method uses event log metadata fed to an LLM to generate a logical representation for Process Mining questions. Distinctively, however, we propose the use of a multi-shot prompt founded on the Process Mining question taxonomy proposed in [2]. Furthermore, to the best of our knowledge, this is the first method to integrate such an LLM-based natural language interface to existing process mining tools, so that the advanced analyses and algorithms they implement can be leveraged to answer the questions.

3 Proposed Method

Our proposed method extends the architecture presented in [2] and is depicted in Figure 1. Colored blocks correspond to the components of this extended architecture. New and revised components are depicted respectively in green and blue, while external components are shown in gray.

The user inputs a question in natural language (English) through the *Text Interface* component, such as "How many cases have been concluded today?". The question goes through an LLM-based semantic parser (components highlighted in green), which is introduced as an alternative to its rule-based counter part proposed in [2].

The role of the semantic parser is to understand the meaning of the input text (question) and convert it to a logical representation that is machine readable.

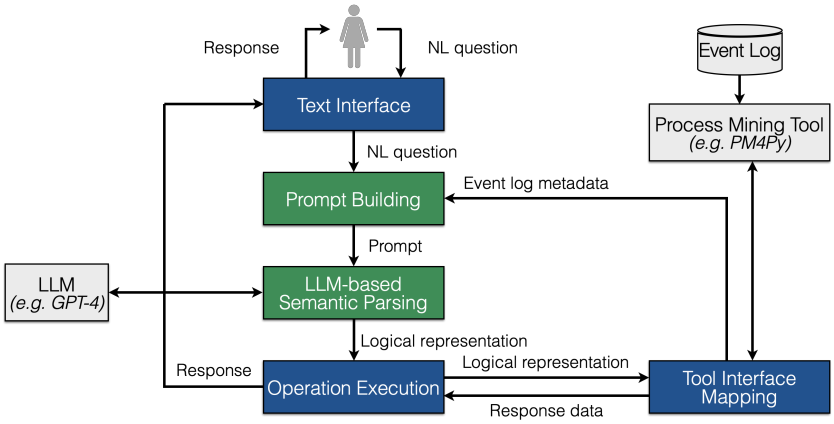

Fig. 1. GPT-based Architecture Overview

This is done by taking the question in natural language, building a prompt to instruct the LLM on how to create this logical representation (which is done by the *Prompt Building* component) and handing it to the model (*LLM-based Semantic Parsing* component). The resulting representation is then mapped into an API call of the underlying Process Mining tool and executed (*Operation Execution* and *Tool Interface Mapping* components). These architecture components are further detailed in the following subsections.

3.1 Logical Representation

The logical representation used to describe questions in [2] is also utilized in this work. It is an extension of the Question Decomposition Meaning Representation (QDMR) proposed in [15]. A QDMR representation comprises a sequence of operations, where each operation is applied to the results of a previous step in the sequence. Operations are inspired by SQL and include `select`, `project`, `aggregate`, `filter` and `group`, among others. The following sequence, for example, represents a question such as "What is the average duration of cases?".

```
select case
project duration #1
aggregate average #2
```

Hash tags refer to the results of a previous operation in the sequence, which may be a set of events, cases or attribute values. In the example above, for instance, `#1` refers to the results of the `select case` operation. For a detailed description of this logical representation, please refer to [2].

3.2 Prompt Building

To enable the LLM to create the proper logical representation for a Process Mining question, a multi-shot prompt [7] was engineered containing the following information:

- A textual description of the logical representation to be used, including each supported operation, its parameters and references
- A small general description of the main Process Mining terms and concepts (event log, event, activity, case, variant, etc.) and how they are represented
- Event log metadata (encoded in JSON), comprising names, types and possible values of attributes contained in the event log, which are obtained from the underlying Process Mining tool
- Metadata for the analyses supported by the underlying Process Mining tool (e.g. conformance checking and rework analysis), also encoded in JSON, including names and returned data types
- Examples of questions and their corresponding logical representations (encoded in JSON)
- Instructions on the expected response contents and format

Listings 1 to 6 depict the actual contents of the prompt. Extensive contents (e.g. metadata and example questions) are abbreviated with "..." for conciseness.

Listing 1. Logical representation description in LLM prompt

```
I'm using a logical representation for natural language questions which is
  similar to SQL. It uses a sequence of operations from the set given below,
  where "reference" is an integer n that refers to the results of the nth
  operation in the sequence:
select concept,
project [distinct] relation of reference,
filter reference where field is [negate] value, ...
```

Listing 2. Event log metadata in LLM prompt

```
Data is organized into 2 tables: case and event, as described in JSON:
{'case': {'case': {'name': 'work_order_id', 'type': 'number'}, 'duration':
  {'name': 'duration', 'type': 'interval'}, ...}, 'event': {'timestamp':
  {'name': 'start_ts', 'type': 'timestamp'}, 'activity': {'name': 'status',
  'type': 'categorical', 'categories': ['open', 'assigned', ...]}, ...}}
```

Listing 3. Process mining terms and concepts in LLM prompt

```
The event table corresponds to the event log of a process execution. Each
  event corresponds to the execution of a single process activity or step
  and is related to a single case or process instance. A case is a temporal
  sequence of events corresponding to a "run" of the process. A trace or
  variant is the sequence of activities executed by a case.
```

Listing 4. Analyses metadata in LLM prompt

```
Predicates apply to specific concepts and may return different data
  depending on the concept they are applied to, as described in JSON:
{'nonconformance': {'trace': {'trace': {'type': 'text'}, 'case_count':
  {'type': 'number', 'sorting': True}}}, 'rework': {'activity': {'activity':
  {'type': 'categorical'}, 'case_count': {'type': 'number', 'sorting':
  True}}, ...}, ...}
```

Listing 5. Example questions in LLM prompt

```
These are examples of questions with their corresponding logical
  representations written in JSON:
How many cases are there in the log?,[{"operator": "select", "concept":
  "case", "ref": []}, {"operator": "aggregate", "aggregate": ["count"],
  "ref": [0]}]
List the non-conformances.,[{"operator": "select", "concept": "case"",
  "ref": []}, {"operator": "predicate", "predicate": "nonconformance",
  "ref": [0]}] ...
```

Listing 6. Response instructions in LLM prompt

```
Respond with a single logical representation in JSON for the given question.
  The JSON representation should not contain c-style comments. If an
  operation uses multiple references, make sure they are given in the
  correct order...
```

The examples of question given in the prompt (Listing 5) are part of a hand-built set of 221 pairs of questions and logical representations. The creation of this set was done as part of this work and guided by the Process Mining question taxonomy proposed in [2]. This taxonomy provides a classification framework for questions, which is organized in seven dimensions:

- *Perspective*: the type of Process Mining (process execution data, process discovery, conformance checking, etc.) the question relates
- *Relativity*: whether the question is absolute or relative to some other data or analysis results
- *Normativity*: indicates if the question requires some normative information/-model to be answered
- *Composition*: whether the given question contains multiple questions inside itself that need separate answers
- *Filtering*: specifies if the question requires a filter to be applied to data before or after computation
- *Ambiguity*: indicates if the question can have multiple interpretations
- *Context*: denotes if the question requires additional information outside its own text to be interpreted (e.g. previous questions and answers made to the natural language interface)

Each dimension is organized hierarchically in a tree structure, so that the classification of a question is done by assigning it a leaf of each tree (dimension). For a detailed description of these dimensions, including their hierarchical breakdown, please refer to [2].

The goal when building this set of example questions was to cover as many taxonomic tree branches as possible along all of these dimensions. Table 1 presents the coverage for each dimension calculated in terms of taxonomic tree leaves represented in the question set for each dimension.

Taxonomic tree branches that are not covered by the example set are related to types of questions that are not handled by this work.

Listing 7 presents a fragment of the 221 pairs of questions and logical representations (encoded in JSON) used to build the prompt. The complete set is available at https://ic.unicamp.br/~luciana.barbieri/promptquestions.csv.

Table 1. Taxonomic coverage of the example question set

Taxonomic Dimension	Coverage
Perspective	52%
Relativity	75%
Normativity	67%
Composition	100%
Filtering	69%
Ambiguity	100%
Context	50%

Listing 7. Samples of questions and logical representations. For the complete set, see https://ic.unicamp.br/~luciana.barbieri/promptquestions.csv.

```
"How many cases are there in the log?",[{"operator": "select", "concept":
  "case", "ref": []}, {"operator": "aggregate", "aggregate": ["count"],
  "ref": [0]}]
"List the non-conformances.",[{"operator": "select", "concept": "case"",
  "ref": []}, {"operator": "predicate", "predicate": "nonconformance",
  "ref": [0]}]
"What are the start activities in the process?",[{"operator": "select",
  "concept": "case", "ref": []}, {"operator": "predicate", "predicate":
  "start", "ref": [0]}]
"How long does my process take, in average?",[{"operator": "select",
  "concept": "case", "ref": []}, {"operator": "project", "relation":
  "duration", "ref": [0]}, {"operator": "aggregate", "aggregate":
  ["average"], "ref": [1]}] ...
```

3.3 LLM-based Parsing

The *LLM-based Parsing* component is responsible for interfacing with the external LLM by feeding it with the prompt built in the previous step (by the *Prompt Building component*) in order to obtain a logical representation for the question. In our experimental implementation, GPT-4 is the LLM of choice, so the *LLM-based Parsing* component plays its role by invoking GPT's Chat Completion API and extracting the logical representation from its response. If a valid logical representation is not present in the response text, a single retry is attempted by re-instructing GPT (in the prompt) that it should answer the question with a valid logical representation.

Other LLMs can alternatively be used for semantic parsing in the future by re-instantiating the *LLM-based Parsing* component alone.

3.4 Hybrid Parsing

As an alternative to the rule-based and pure LLM-based parsers we also propose a hybrid approach, as depicted in Figure 2.

The idea is to initially feed the question to the original rule-based parser proposed in [2] (flow 1 represented with red dashed lines in Figure 2).

The question goes initially through the *Pre-processing and Tagging* component, which is responsible for splitting it into tokens and tagging it with part-of-speech information, (linguistic) dependency relations and recognized entities. It is then semantically parsed by the *Rule-based Semantic Parsing* component

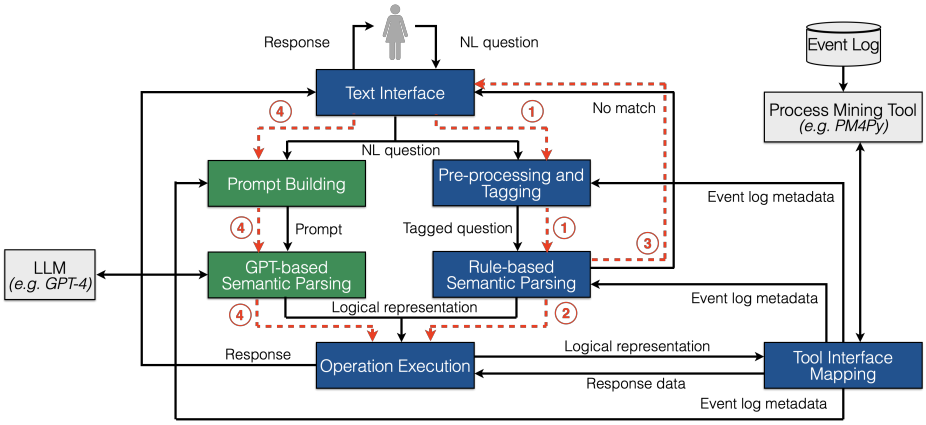

Fig. 2. Hybrid Architecture Overview

using a rule-matching approach (for more information on the *Pre-processing and Tagging* and *Rule-based Semantic Parsing* components, please refer to [2]). If a rule is triggered, the resulting logical representation is used (flow 2). Otherwise, a "no match" indication is returned (flow 3) and the question is then fed to the LLM-based parser (flow 4). The logical representation obtained from the model (Chat Completion API, in the current implementation) is used in this case.

The reasons for choosing to have the question initially parsed by the rule-based parser (and only go through the LLM-based parser when the first is not able to translate it to a logical representation) are its lower cost and response time, alongside its higher predictability.

3.5 Operation Execution and Tool Interface Mapping

After a logical representation is created for the input question, the *Operation Execution* component manages its execution. Each operation contained in the logical representation is carried out by invoking the *Tool Interface Mapping* component, with final results being returned to answer the question.

The *Tool Interface Mapping* component, on its turn, maps the logical operations to real API calls of a Process Mining tool. In our experimental implementation, PM4Py [6] is used as this underlying tool. It is an open-source library written in Python that implements a variety of process mining algorithms.

Using PM4Py's ability to handle event logs as pandas objects allows a straightforward mapping of operations such as `select`, `project`, `filter` and `aggregate` over case and event data. Predicates such as `nonconformance` and `rework`, on their turn, are directly mapped into calls to API methods that implement these analyses.

4 Experimental Results

In order to verify the applicability of the presented methods and compare their results to the existing pure rule-based parser, we have implemented the newly proposed components and integrated them into the existing architecture [2]. The gpt-4-1106-preview model was used for LLM-based parsing, with the Chat Completion API's *seed* parameter set to a fixed value to reinforce predictability.

The question set originally presented in [3] and available at https://ic.unicamp.br/~luciana.barbieri/promptquestions.csv was used to evaluate the implementation. One should notice that this is a set completely independent of the smaller, hand-built set described in Section 3.2 and used to build GPT's prompt. The test set is composed of 794 questions in English. Before this evaluation, each question of this set was manually analyzed to determine if it is possible to create a logical representation that could be mapped to a call and executed by the underlying Process Mining tool. The analysis concluded that, from the original 794 questions, 524 can be fully represented, mapped and executed, while this can be partially done for 96 of them. A question can be partially handled, for example, if it is a composite question for which at least one sub-question can be answered, while others cannot, such as "What are the deviations from the expected model? Why did they happen?". The remaining 174 questions cannot be handled either due to language-related problems (e.g. incomprehensible sentences), Process Mining misconceptions, or because the required functionality is not supported by the underlying Process Mining tool.

The 620 (fully or partially) answerable questions were executed against a real-life Work Force Management-based event log processed by PM4Py. However, any Process Mining event log could be used, as the testing questions are not specifically bound to any particular event log. Questions pass or partially pass a test if the obtained logical representation fully or partially answers them, respectively. If no valid logical representation is obtained or if it does not answer the question correctly, the test fails. Table 2 presents our experimental results.

Table 2. Experimental Results.

Parsing Approach	Passed	Partially Passed	Failed
Rule-based	302 (48.71%)	125 (20.16%)	193 (31.13%)
GPT-based	376 (60.65%)	108 (17.42%)	136 (21.94%)
Hybrid	350 (56.45%)	153 (24.68%)	117 (18.87%)
Ground truth	524 (84.52%)	96 (15.48%)	0 (00.00%)

When analyzing these results, one can initially observe that the GPT-based parser is able to answer more questions with a fully accurate response when compared to rule-based and hybrid approaches. Another important advantage of this parser is that it generalizes well and is able to respond abstract questions such as "What is the most complex case?" with plausible answers it responded this particular question with the case that executed the most distinct activities). Disadvantages, on the other hand, are the higher cost and response time when

compared to the rule-based parser, which can run solely on the user device. It is also worth noticing that, similarly to other AI-based models, the GPT-based parser lacks predictability (even with the use of the *seed* parameter), implying that the same question may be answered differently on separate attempts.

The hybrid approach, on its turn, was able to answer more questions correctly if we consider both full and partial responses, reducing the share of incorrect behavior (failed responses). This can be explained by the fair accuracy of the (deterministic) rule-based parser when dealing with predictable, well-behaved questions [3] being combined with the ability of the GPT-based parser to handle the more open, abstract ones. Additionally, as GPT-based parsing is only used when rule-based parsing does not find a matching rule for the question, cost and response time issues are minimized, as well as unpredictability. The drawback of the approach, however, is that it is not able to detect when the rule-based parser triggered a rule that caused a question to be partially or incorrectly answered (when the GPT-based parser might have been able to provide a fully correct answer).

5 Conclusions and Future Work

In this work, we propose a new architecture for a Q&A natural language interface for Process Mining that combines LLM technologies and Process Mining concepts organized in a taxonomy. This architecture is envisioned to balance the natural language processing capabilities of LLMs with the power of Process Mining tools, such that the first can drive interactions with the latter.

We also experiment around hybrid approaches, where LLM based implementation complements a rule-based one, in search of a better balance between generalization and precision. Finally, we test these architecture variants against real-life questions and event log, to assess their performances. In a nutshell, LLM based methods seem to help reduce failures in answering questions by increasing the number of questions that are fully or partially answered.

Based on our findings, we would like to explore the following directions of future work:

- *Multi-agents*: interface with LLM through a multi-agent framework, such that interactions can be broken down into smaller tasks that LLMs can handle better
- *Fine tuning*: experiment the impact of fine tuning LLMs to better understand Process Mining taxonomy
- *Other LLMs*: compare the performance of alternative LLMs such as Gemini and Claude, among others
- *Conversational*: evolve the question-and-answer interface to a conversational one, where context about previous interactions is used to more naturally speak to human users

Acknowledgments. We would like to thank Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001, for providing the financial support for this work.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. van der Aalst, W.M.P.: Process management after ChatGPT: How generative and predictive AI relate to process mining. http://www.linkedin.com/pulse/process-management-after-chatgpt-how-generative-ai-wil-van-der-aalst-lyyzc/ (2023)
2. Barbieri, L., Madeira, E., Stroeh, K., van der Aalst, W.M.P.: A natural language querying interface for process mining. Journal of Intelligent Information Systems **61**(1), 113–142 (2023)
3. Barbieri, L., Madeira, E.R.M., Stroeh, K., van der Aalst, W.M.P.: Towards a natural language conversational interface for process mining. In: Process Mining Workshops, ICPM 2021. Springer International Publishing, Cham (2021)
4. Berti, A., Kourani, H., Hafke, H., Li, C.Y., Schuster, D.: Evaluating large language models in process mining: Capabilities, benchmarks, evaluation strategies, and future challenges. arXiv preprint arXiv:2403.06749 (2024)
5. Berti, A., Schuster, D., van der Aalst, W.M.P.: Abstractions, scenarios, and prompt definitions for process mining with LLMs: a case study. In: International Conference on Business Process Management. pp. 427–439. Springer (2023)
6. Berti, A., van Zelst, S., Schuster, D.: PM4Py: A process mining library for python. Software Impacts **17**, 100556 (2023)
7. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)
8. Han, X., Hu, L., Sen, J., Dang, Y., Gao, B., Isahagian, V., Lei, C., Efthymiou, V., Özcan, F., Quamar, A., Huang, Z., Muthusamy, V.: Bootstrapping natural language querying on process automation data. In: 2020 IEEE International Conference on Services Computing (SCC). pp. 170–177 (2020)
9. Jessen, U., Sroka, M., Fahland, D.: Chit-chat or deep talk: prompt engineering for process mining. arXiv preprint arXiv:2307.09909 (2023)
10. Kermani, M.A.M.A., Seddighi, H.R., Maghsoudi, M.: Revolutionizing process mining: A novel architecture for ChatGPT integration and enhanced user experience through optimized prompt engineering. arXiv preprint arXiv:2405.10689 (2024)
11. Kobeissi, M., Assy, N., Gaaloul, W., Defude, B., Benatallah, B., Haidar, B.: Natural language querying of process execution data. Information Systems **116**, 102227 (2023)
12. Kobeissi, M., Assy, N., Gaaloul, W., Defude, B., Haidar, B.: An intent-based natural language interface for querying process execution data. In: 3rd International Conference on Process Mining (ICPM). pp. 152–159. IEEE (2021)
13. Markets and Markets: Process Mining Market by Offering - Global Forecast to 2028. https://www.marketsandmarkets.com/Market-Reports/process-mining-market-176608355.html (2023)
14. OpenAI: GPT-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
15. Wolfson, T., Geva, M., Gupta, A., Gardner, M., Goldberg, Y., Deutch, D., Berant, J.: Break it down: A question understanding benchmark. Transactions of the Association for Computational Linguistics **8**, 183–198 (2020)