Process Model Generation Through Prompting: a Process Model Structure Analysis*

ABSTRACT

The automated construction of business process models from textual documents is a challenging research area that still lacks the ability to scale to a variety of real-world scenarios. Here, the lack of massive annotated data makes deep learning approaches to perform the transformation of text into process model graphs nearly infeasible. Furthermore, a process model is composed of non-trivial conceptual entities and relations we need to capture in texts. To solve these challenges, we adopt the GPT-3 Large Language Model with a prompt-based in-context learning strategy in a multi-turn dialog fashion to generate a process model out of process description.

Previous literature on this topic mainly focuses on the analysis of the quantity of process model information extracted. However, those results do not provide any insight into the overall process model structure generated. Here, false positive and false negative relations may change completely the process model graph structure by changing its semantics and leading to possible catastrophic consequences in real scenarios.

In this paper, we focus on this aspect. We propose an analysis of the process model generated, in the form of Directly Follows Graph (DFG), starting from the information contained in the textual description of process models and procedures. Our contribution provides further insights into the understanding of the possibilities and highlights the limits and challenges of adopting LLMs to generate business process models out of textual documents.

CCS CONCEPTS

• **Information systems → Business intelligence**; **Process control systems**; **Decision support systems**; **Document structure**; **Novelty in information retrieval**; **Retrieval models and ranking**;

KEYWORDS

Process Extraction from Text, In-context learning, Process Model, Large Language Model, Business Process Management

ACM Reference Format:
. 2024. Process Model Generation Through Prompting: a Process Model Structure Analysis. In *Proceedings of ACM SAC Conference (SAC'24).* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/xx.xxx/xxx_x

*Produces the permission block, and copyright information

SAC'24, April 8 âĂŞApril 12, 2024, Avila, Spain

ACM ISBN 979-8-4007-0243-3/24/04…$15.00
https://doi.org/xx.xxx/xxx_x

1 INTRODUCTION

The recent increase in computational power and the advent of new Large Language Models (LLMs) are greatly changing the information and communication technologies (ICT) systems. Manual tasks that require a huge human effort to be completed, nowadays, start to be automatized by artificial intelligence (AI) models. Intelligent systems, co-pilots, and chat-bots are playing a crucial role, especially in industry, opening up new possibilities to support businesses and helping companies to survive and compete in fast business revolutions. These Intelligent systems can now analyze a great amount of data in an efficient way. They provide valuable support by reducing time, errors, and costs. Thus, researchers from both academia and industry are interested in finding an automatic way to analyze the massive amount of textual documents a company has to generate structured representations with different expressivities (e.g., process models) of the information these textual resources contain. Obtaining process models of processes is vital to performing analyses and increasing the efficiency of the processes and procedures. However, the high cost of manually analyzing and generating such representations from documents has impeded the generation of such models, hampering process analyses and thus lowering process performance.

A process model is a formal representation of a process performed in a company. These models decompose the overall process in single steps (called tasks) and provide the real picture of a process to business analysts. Process models are vital to identify bottlenecks and problems and then make the necessary process improvements to reduce resources, errors, times, and costs. Textual descriptions of business processes (e.g., Standard Operating Procedure) are textual documents describing how a procedure or a process is performed in a company, e.g., the process of handling a customer claim. These descriptions that typically describe a procedure "model" and not a specific process execution, should be easy to understand by all the parties involved in the process. However, the automatic extraction of the information they contain is hampered by several challenges. The information they store does not follow a common guideline. The format, writing style, and structure of the document change from company to company. This goal is also impacted negatively by the lack of massive annotated data on textual descriptions of business processes, which makes classical deep-learning approaches almost infeasible. Furthermore, the multidimensional nature of the process model entities and relations (e.g., ranging from temporal elements, like activities, and their temporal order, to the resources manipulated by the activities, to the actors involved somehow in activity execution) makes the generation of a process model from documents a challenging research area.

Nowadays, *Large Language Models* (LLMs) are shifting the NLP paradigm, opening the possibility to understand, analyze, and perform complex reasoning tasks over long and complicated textual

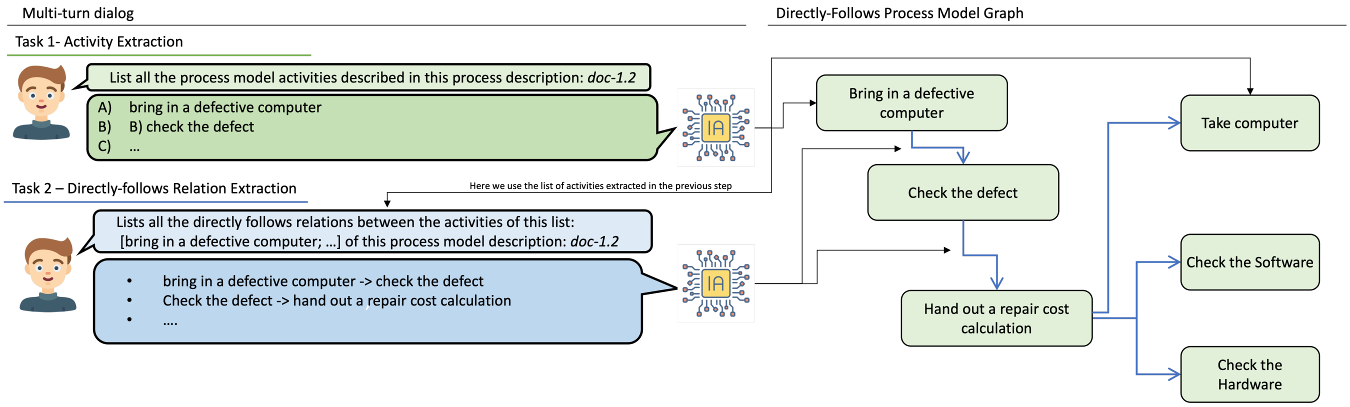

Figure 1: In this figure, we show an example of the approach. The excerpt is taken from *doc-1.2* of the PET dataset (see Section 4.1). In this multi-turn dialog, the artificial agent acting as domain expert guides the construction of the process model, in the form of Directly Follows Graph (DFG), by answering the user. The user poses specific questions to extract the process model elements and guide the construction incrementally. In the first step, the user asks for the list of activities of a process document to add activity nodes (green squares). Then, the user asks for the set of temporal relations between the activities and uses the answer to add directly-follows relations (blue arrows) to the process model graph.

resources (such as a process model description) without being constrained to train a language model with a great amount of carefully annotated data specific to the particular task one may need to solve. These advances have recently stimulated a growing body of literature claiming for a possible integration at different stages of LLM in the context of Business Process Management (BPM) [4, 5, 30]. In this paper, we *in-context learning* and the GPT-3 LLM to perform the text-to-process model transformation.

Different from the literature contributions on this topic that mainly focus on quantifying the extraction of process elements, the contribution of this paper, therefore, is an in-depth exploration of the structure of the process model to provide an in-depth understanding of their effectiveness beyond the pure quantitative measure of the quantity of process model information extracted. Since, for example, the quantity of false positive and false negative relations do not provide any insight into the overall process model structure generated. The presence of these false relations may change the structure and consequently, the semantics of the process model with unpredictable negative consequences in real scenarios. It is important to highlight that the aim of this work is not on the approach proposed to generate the process model. It is on the analysis of the structure of the process model generated out of process documents. To the best of our knowledge, this in-depth analysis is performed for the first time in the literature and can pave the way for future research by shedding light on new challenges in this topic.

2 RELATED WORK

Our work aims at building a process model from scratch by starting from a natural language description of a business process. Our approach relies on the use of LLM and in-context learning techniques. For these reasons, our work can be found at the intersection between BPM and NLP. In this section, we provide the literature contributions related to these two research fields. We begin with the analysis of the contributions concerning process model extraction from text in the BPM field. Next, we analyze the contributions concerning the use of LLM to extract information from text and generate the structured representation, e.g., in the form of knowledge base.

Process extraction from text task [3] aims to find an algorithmic transformation method to extract meaningful process model information from process descriptions and generate the underlined process model in a specific formalism (e.g., BPMN). Several factors, such as the ambiguous nature of natural language, the multiple possible writing styles, and the great variability of possible domains of application make this task extremely challenging. Indeed, as recent papers on this topic highlight [17, 28], this task is still in an early stage of development and its adoption in real-world scenarios seems to be far away. The literature contributions propose two distinct solutions to the process extraction from text task (see Figure 2). The first approach is a direct transformation of the input text into a specific process model formalism. The mapping is performed via a single function f (graphically depicted in the top part of Figure) that analyses the text and generates the process model diagram. This approach is typically implemented via a *complex and*

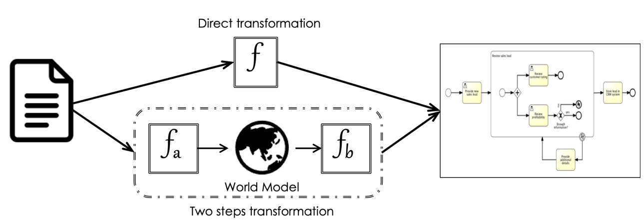

Figure 2: The figure, taken from [3], shows the two approaches proposed in the literature to perform process extraction from text.

ad-hoc tailored pipeline, as proposed in [29]. However, the advantage of defining a tailored transformation can become a drawback when the algorithmic function f is applied to different contexts. A further approach towards the implementation of a direct mapping f is the exploitation of Artificial Neural Networks. However, nowadays, there does not exist any gold standard dataset that can be adopted to train deep learning solutions to perform such transformation. The second approach is a two-step transformation approach with intermediate representation. This approach follows the principle of divide-and-conquest. Firstly, an analysis of the text is performed and meaningful process model elements are extracted (f_a). These elements are memorized into a structured intermediate representation (a world model). Then, the second component of the pipeline generates the process model in a specific formalism, starting from the information contained in the world model (f_b). This direction has been partially explored in [1, 9–11, 13, 21, 24, 29]. These contributions rely on template and rule-based approaches, which often lack the flexibility needed to fully cover the great variability of text one may encounter in this domain. Recent contributions [12, 20] try to leverage deep learning models to provide more flexibility to their approaches. But they (somehow ironically) targeted highly structured text [12] or sequential lists of tasks [20], thus avoiding the real-world problems just highlighted. The problems of leveraging the potential of deep learning solutions to solve this task is the **lack of the high quantities of carefully annotated data on textual descriptions** [3] needed to make these techniques work. The only annotated datasets providing gold standard annotation, the PET dataset [6] is too small in size to address this data limitation issue. Creating annotation campaigns to collect gold standard data specific to this task is difficult to organize given the multi-perspective nature of process elements (activities, data objects, process participant (actors), resources, flow objects, and their mutual relations, among others), which require an articulated set of annotation labels and laborious planning. Therefore, the adoption of an efficient way of deep learning techniques in this topic seems to be a not-practicable way to follow.

The information extraction from textual resources is a foundational NLP research area widely explored in the literature [18]. Specifically on the use of LLMs, several works investigated the use of LLMs to understand both linguistic and semantic properties of possible word representations and also how LLMs can be exploited within specific knowledge and linguistic tasks. Concerning the capability of LLMs to perform natural language inference, the works proposed in [19] show how both BERT and ELMo-based models are able to infer syntactic relationships from natural language texts. Then, in [27], the authors investigate to what extent language models encode sentence structure for different syntactic and semantic phenomena and find that they excel for the former but only provide small improvements for tasks that fall into the latter category. While this provides insights into the linguistic knowledge of language models, it does not provide insights into their factual and commonsense knowledge. In [22] the authors introduced a language model (LM) based on transformers which they called generative pre-training (GPT-1). This work evolved in two further versions: GPT-2 [23] and GPT-3 [7]. These LMs demonstrated their suitability to work within zero-shot environments in several tasks and their ability to store factual knowledge. Moreover, the authors

task instr. 1 *List all the process model activities described in this process description: [PROCESS DESCRIPTION].*
task instr. 2 *Lists all the directly-follows relations between the activities of this list: [activity-list] of this process model description: [PROCESS DESCRIPTION]*

Figure 3: The list of task instructions used in prompts.

of GPT-3 demonstrated how it is possible to perform fine-tuning operations on the PLM in order to enhance its effectiveness within specific tasks or domains. The authors of [14] exploited LLMs for the construction and completion of knowledge bases and opened interesting directions towards automation and high-precision curated knowledge base extraction. LLMs have been demonstrated to be efficient language models to adopt to perform the extraction of information from semi-structured resources, as pursued in Yago [26] and DBpedia [2]. Regarding the prompting techniques, we can say that our approach is somehow similar to the Decomposing Prompting approach [15] since we decompose the high-level task into two smaller sub-tasks **T1** and **T2**. These sub-tasks can be seen as a type of intermediate states of the problems. Since there are no more intermediate states to reason from the sub-tasks we cannot be decomposed anymore. In our task we do not perform any type of research (either breadth-first or depth-first) of process model entities in the text and we do not need to reason on them, e.g., as it is required to answer a single math question in isolation. Also, since the focus of this paper is on the evaluation and not on the approach, we do not apply Chain-of-Thought (CoT) [31] or Tree-of-Thoughts (ToT) [32] strategies in our settings. These family of techniques would have been a positive impact only if our aim was to solve the overall task in one step. Finally, only the work proposed in [4] proposed a new approach that aims at extracting a KG of a process from process description documents, in an incremental and conversational manner, exploiting Large language models and in-context learning. The proposed approach extracts conceptual information by posing specific questions to the LLM. The answers are then used to instantiate elements and relations in a domain-specific knowledge graph.

3 PROCESS MODEL GRAPH GENERATION THROUGH PROMPTING AND IN-CONTEXT LEARNING

The advent of the GPT-3 [7] LLM greatly changed the NLP paradigm on how a language model can be "fine-tuned" for task-specific applications. A growing body of literature has demonstrated the ability of the GPT-3 model to understand texts and solve tasks on the texts in a human-like fashion. Indeed, it is able to solve complex NLP tasks and generate good-quality task-related answers by understanding the text and the task instructions provided (see Fig. 4). Furthermore, it can analyze and understand a large portion of text at once, pushing far away the text limit of other transformer-like language models such as BERT [8] or RoBERTa [16]. In addition, it is possible to use in-context learning techniques with this LLM. That means providing some examples of the task to solve together with the text to analyze and the task instructions directly in the input (called prompt), without doing a canonical fine-tuning of the model parameters toward a single specific downstream task. The coupling of the in-context learning technique and the GPT-3 LLM has been shown to be extremely useful to solve complex tasks that suffer from low-resource issue [4, 25].

Our experimental domain is impacted negatively by two challenges: (*i*) large input, and (*ii*) low-resource issues. Thus, we decided to adopt in our investigations the GPT-3 LLM combined with *in-context learning* technique. The GPT-3 model solves the first problem since it can analyze a large portion of text at once. The *in-context learning* technique solves the second problem since it is possible to instruct an LLM to solve a particular task using very few examples of the task to solve. Hence, since our experimental domain suffers from low-resource issues as well, we decided to adopt in our investigations the GPT-3 LLM combined with *in-context learning* technique.

3.1 Implementing the approach

The incremental approach we implement (see Fig. 1) is similar to the one proposed in [4]. In This work, an LLM is used to generate a Knowledge Graph out of procedural description documents. We differ from this work in a substantial manner since we use different task instructions, we assess our approach on a broad set of documents, and we do not focus on the performance of extraction and generation. Therefore the results and lessons learned presented in this paper are likely to pave the way for future efforts, possibly involving different strategies, and maybe also other LLMs. Again, we want to remark to the reader that the goal of this paper is not on the approach. The focus is on the analysis of the process model graph in the form of directly-follows graph, generated from textual resources.

Similarly to [4], we designed two specific task instructions (see Figure 3) to enable the extraction of *activities* (**task instr. 1**) and their *directly-follows* (temporal) relations (**task instr. 2**). These task

in-context learning

1. **Q:** Lists all the directly-follows relations between activities of this list of activities: {list gold activities doc-1.4} of the process model described in this process description: doc-1.4
2. **A:** a) make the decision to go public -> select underwriters,
   b) Select underwriters -> prepare a registration
   c) Prepare a registration… -> check. ….
3. **Q** Lists all the directly-follows relations between activities of this list of activities: {list gold activities doc-5.4} of the process model described in this process description: doc-5.4
4. **A:** a) Purchase a product or a service -> submit an expense report
   b) Submit an expense report -> review the expense report …
5. **Q:** Lists all the directly-follows relations between activities of this list of activities: {list activities doc-1.2} of the process model described in this process description: doc-1.2
6. **A:**

Figure 4: The figure shows an example of the *Cov Prompt* for task T2. The blue line marks the in-context learning part of the prompt where we provide the two gold standard examples texts and their list of gold standard activities, together with task instructions (lines 1 and 3). We instruct the language model on how to solve the task by providing the gold standards Directly Follows Graph model (lines 2 and 4) for each text. The yellow line marks the raw part of the prompt where we provide only the text to analyze together with the task instructions (line 5) and we wait for the answer (line 6). Intuitively, a *Raw prompt* is composed of the yellow part only. The task instructions are the same in the in-context learning and raw parts of a prompt.

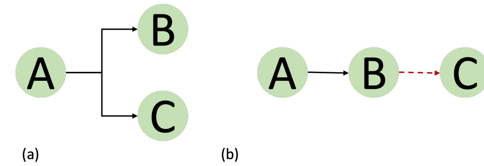

Figure 5: The figure shows an example of a gold standard graph (a) and the predicted one (b). Circles represent activities, black arrows represent a true positive directly-follows relation, dashed red arrows represent a false positive directly-follows relation, and dotted orange arrows represent a false negative relation.

instructions, performed in an incremental manner (see Figure 1), mimic the follow-up questions we may pose to a domain expert to build our final conceptual models (process model graph).

The next step is the construction of the input, called *prompt*, to feed the LLM. We create a set of prompt templates with the proper task instructions to simulate each step in the conversation. Before feeding the prompt to the language model, we fulfill the in-context learning place-holders with gold standard examples of the task and we provide task-related information, i.e., a list of activities, to the prompt. Figure 4 shows an example of a prompt template we customized in our experiments to enable in-context learning. Finally, prompts are fed into the model to generate the answer and we use the answers to generate the process model graph of a document.

4 EMPIRICAL ASSESSMENT

In this research, we adopt an LLM to generate the process model from descriptions of process models. The focus of this research is neither on the approach used to generate a process model graph nor on the integration of prompting and in-context learning to solve a complex task such as the extraction of conceptual entities from text [5]. Different from the literature contribution in this topic that mainly concentrates on the exploitation of LLM to perform the extraction of process model information from text, in this paper we want to assess the quality of the process model graph generated from the answers. Since the solely quantitative analysis of the conceptual information extracted from the text may hide the real quality of the process model graph generated. For example, false positive and false negative relations may change completely the overall graph structure and create a false positive split point in the process model or may create the opposite situation and generate a linear structure instead of a more complex one (see Fig 5).

Our final objective is to answer the meta-research: *Are LLMs good candidates to support the construction of domain-specific knowledge graphs from texts in a low-resource scenario?.* We decided to decline this research question since it is too broad to be investigated in a single article. In this work, we want to provide a partial answer by answering to three more specific questions.

RQ1 **Type of training examples.** *Does the quality of the examples provided inside the prompts to enable in-context learning have a great impact on the quality of the structure of the process model generated?*

RQ2 **Building strategy.** *Within our incremental approach, does the LLM generate better process model graphs starting from the data it extracts as responses to the previous questions or starting from gold-standard data?*

RQ3 **Usage of context.** *What is the impact of providing problem-context information during prompting on the structure of the process model generated? Does it help the LLM to generate better results?*

4.1 The PET dataset Dataset

In this Section, we briefly introduce the PET dataset dataset [6], a gold-standard dataset specific for process information extraction tasks. The dataset is a collection of process model descriptions annotated at the textual level. It proposes a general process annotation schema not related to any particular process model diagram formalism. The annotations provide information about the overall process model graph described. Our focus of interest in this research targets the activities and their relations. An 'activity" in a process model represents a single task performed during the execution of a process instance. That is the single task a user performs or he is responsible for its execution. Activities are annotated in PET dataset at their atomic level by differentiating among, i.e. the activity verb, e.g., *check*, the object it operates, called activity data *the defect*, and other details. In our experiments, we consider as activity their combination (e.g., *check the defect*). Since we are interested in the generation of a process model, we also consider the temporal execution order between the activities that are annotated in PET dataset as *flow* relation (e.g., *bring in a defective computer→check the defect*). In our experiments, we create the gold standard process model, in the form of *directly-follows graph* (DFG), starting from the annotation of activities and their (directly follows) flow relation as annotated in PET.

4.2 The Tasks

The overall task we are assessing is the generation of a process model out of a process model document starting from the predictions of the LLM. We designed a multi-turn pipeline approach in which each interaction extracts process model information and we use it to generate the process model incrementally (see Figure 1). Our dialog is composed of two steps. Each step corresponds to a single task, designed to target the extraction of a particular element or relation.

T1 In the first step, we aim to extract the set of activities described. To solve this task, we customize the prompt with the task instruction: **task instr. 1**.

T2 In the third step, we target the extraction of the temporal relation (directly follows relation) between the activities starting from a process description and its list of activities. In this case, we customize the prompt template with the task instruction **task instr. 2**.

For each task, we created a set of prompt templates supporting the extraction of process model information using the LLMs. We filled the templates with the specific experimental settings, the proper set of training examples, and the text we want to generate the process model graph from (see Figure 4).

4.3 Experimental Setting

To provide solid answers to our research questions, we exploited an experimental parameter to investigate each of our research questions. We use the parameters the customize the prompts.

Parameter 1 **Quality of the examples provided in the in-context learning component of prompts.** The first experimental parameter we want to manipulate regards the choice of the "quality" of the examples to enable in-context learning. We want to test the hypothesis that good-quality examples that cover somehow all the possible cases are more effective for solving complex tasks, such as the one in our domain. This parameter is explored to answer our **RQ1**.

Parameter 2 **Building strategy.** The second parameter we are manipulating wants to address our **RQ2**. Here, we want to assess the quality of the generation of a process model in real-world scenarios where we start to build the model from previously extracted data without any "gold standard" entities to rely on.

Parameter 3 **Exploitation of contextual-domain information.** The third parameter we are manipulating wants to assess is the impact of providing problem-context information to answer to **RQ3**. The hypothesis is that by providing problem-context information about the topic of the task to solve, the structure of the process model generated should be more similar to the gold standard one compared to the process model generated without this information in prompt.

We explain the experimental parameters more in detail.

***Parameter 1**: Quality of the examples provided in the in-context learning component of prompts.* We designed in total four prompts to test our **RQ1**: (1) *Raw Prompt* (2) *Min Prompt* (3) *Max Prompt* (4) *Cov Prompt*

The *Raw Prompt* is a prompt without in-context learning (the yellow component of Fig.4) since we do not provide any example of the task to solve within the prompts. We use this prompt to test the tasks on the bare LLM and to have a baseline to compare with. In the *Min Prompt*, we provide a minimal set of examples in the in-context learning component of the prompt. Here, we provide two documents (docs 8.1 and 10.13) that present a process model with a linear structure. This set of examples is used to understand if the LLM can generate complex structures even if it has never seen a complex one before (in the in-context learning examples). In other words, we want to assess if the LLM can generate process models with split and merging points without being exposed to them. We tested the opposite scenario with the ***Max Prompt***. Here, we provide two documents (docs 2.1 and 4.1) that both present a process model with a complex structure composed of split and merging points in the same models. This set of examples is used to test the opposite situation than the *Min Prompt*. Providing the maximum quality of information that covers "all the possible cases" (split and merging points) should generate process models with complex structures, that should be more similar to the gold standard ones. Finally, in the *Cov Prompt*, we want to provide a balance between the *Min Prompt* and the *Max Prompt*. We selected these two documents (docs 1.4 and 5.4) by constraining them to present the maximum coverage of process model elements of our interest. The first document has a split and a merging point while the second in-context learning example presents two open split points only but no merging point. We use this training set to assess if a balanced set of examples could generate better results. This prompt does not exacerbate the ratio between the quantity of the information (in terms of process model structure) and the text example length of the examples to be null (*Raw Prompt*), too stringent (*Min Prompt*), or maximum (*Max Prompt*).

Parameter 2*: Building strategy.* The second parameter we manipulate assesses the quality of the generation of a process model in a real extraction scenario. Here, we are interested in comparing the quality of the process model generated starting from information previously extracted against the process model generated starting from gold standard data. We designed two settings: **Incremental** and **GoldStandard**. Under the **Incremental** setting, we construct the process model starting from the data extracted in task **T1**. Practically, we use the *list of activity* extracted to fulfill prompt templates in task **T2**. While in the **GoldStandard** setting, we construct the process model graph starting from gold standard data, by fulfilling the prompt templates with the list of gold standard activities.

Parameter 3 *Exploitation of contextual-domain information.* The third parameter of the experimental setting target **RQ3** is to assess if problem-context information that may be provided to prompts may help the model to generate better process models. We designed two experimental settings: *not context enhanced* and *context enhanced* to address this research question. In *not context enhanced*, we do not provide any extra information about the topic of the problem to solve in prompts. In *context enhanced*, we added the contextual information about the domain of the task to solve, at the beginning of the task instructions in prompts. Specifically, the problem-context information we are injecting instructs the LLM to consider the context of Business Process Management.

In our research, even if conversationally use the GPT-3 model, we do not aim to evaluate the quantity of correct information about a process model graph extracted from text. Instead, we aim to evaluate the quality of the process model graph generated by comparing it with the gold standard to provide insights for future research. Thus, in our experiments, we use the *text-davinci-003* engine of the GPT-3 model and we set all the model's parameters (e.g., sampling temperature) to 0.0 to preserve the reproducibility of all results. In summary, our training sets are composed of 6 documents in total, two for each in-context learning training set. We investigated our research questions on a test set composed of the remaining 39 documents of the PET dataset. We performed 16 experimental items as the results of the combination of all possible values of the three parameters, i.e., 4 types of prompt, 2 building strategies, and 2 context enhancement settings for each test item of our test set [1].

5 QUALITATIVE ANALYSIS

We reported in Table 1 the analysis of the structure of predicted process model graphs (DFG models). We compare the results against the gold-standard statistics (first row of the table). We split our observations by the type of parameters we adopted to provide a clear picture of the problem we are investigating. In this section, we provide an analysis of how the structure of the generated process model changes.

Parameter 1 *Quality of the examples provided in the in-context learning component of prompts.* Comparing the prompts against the gold standard, the generated DFG model representation has, in general, fewer nodes and edges with fewer variabilities given by the lower standard deviations. The *Max Prompt* is an exception to this trend since it can generate only several nodes resembling the gold data. Looking at the process model structure generated, we discover that all the prompts generate a linear structure [2]. Only the *Max Prompt* can generate more complex DFGs. However, *Max Prompt* suffers from *text limit problem.* Indeed, analyzing its statistics in the table, the minimum number of edges found among the predicted DFG models is 0. This means that it was unable to provide an answer about the *directly-follows* relations of a text. Here, text limit causes the generation of some partial answers in tasks when an LLM has to work with a long text. Hence, it occurred that in the DFG model predicted by performing in-context learning with the longest texts we were able to find only nodes, and consequently, the presence of the largest number of isolated nodes and isolated components. This explains the lower scores obtained in the three tasks by *text limit problem.* Only the *Cov Prompt* generated DFG model structures without isolated nodes and components. This could mean that by using this prompt, the LLM can correctly connect all the information extracted previously. Also from the qualitative perspective, we can positively answer to **RQ1** since the usage of different quality of information to enable in-context learning in the prompts impacts differently on the structure of the process model generated, especially concerning the node isolation aspect. In the end, we can say that using a set of documents that present a high ratio between the coverage of process elements and the text length appears to be a winning strategy.

Parameter 2 *Building strategy.* Through the comparison of the two possible values for this parameter, we note a substantial difference between them in terms of edges and DFG model structures. Since in the **GoldStandard** extraction context we start from the list of gold-standard activities the number of nodes and the standard deviation remain almost the same among the four experimental prompts (*Raw Prompt*, *Min Prompt*, *Max Prompt*, *Cov Prompt*). Providing gold-standard data to prompts, the LLM generates many more relations and generates more complex DFG model structures, i.e., the number of linear structures decreases considerably. The partial answer problem, highlighted above, is still present in the **GoldStandard** setting, causing the generation of many isolated nodes and components in the *Max Prompt*. The *Cov Prompt* generated a large number of edges concerning the others. Instead, by using the **Incremental** setting, we may observe how the number of linear models generated dramatically increases. On the contrary, we may observe how the number of edges decreases by around 30% on average. This aspect leads to the state that from a qualitative

[1]All the material associated with this research is available at https://anonymous.4open.science/r/Process-Model-Generation-Through-Prompting-a-Process-Model-Structure-Analysis-4638

[2]linear since the process model graph's nodes have at most one incoming edge and one outgoing edge.

Table 1: Graph summary. The table reports the summary of the information about the structure of the *directly-follows* graphs generated from predictions. We report the average size (avg), standard deviation (std), minimum number (min), and maximum number (max) of nodes and edges, and the graph topology information. We report in column l the total number of graphs with a linear structure; in column s the total number of graphs that present only split points; in column m the total number of graphs that present only merging points; in column b the total number of graphs that presents both split and merging points; in column i.n. the absolute number of isolated nodes; and in column i.c.. the absolute number of isolated components.

	Incremental extraction context														**GoldStandard** extraction context													
	node				edge				graph structure						node				edge				graph structure					
prompt	avg	std	min	max	avg	std	min	max	l	s	m	b	i.n.	i.c.	avg	std	min	max	avg	std	min	max	l	s	m	b	i.n.	i.c.
Gold-standard																												
	9.85	*5.6*	*4*	*32*	*10.03*	*6.5*	*3*	*35*	*6*	*8*	*0*	*25*																
not context enhanced																												
Raw	8.67	4.03	4	26	7.54	4.02	3	25	30	0	5	4	7	15	9.67	5.31	4	30	9.21	5.4	3	30	22	1	3	13		
Min	8.92	4.08	4	23	7.95	4.01	3	22	34	0	1	4	1	4	9.72	5.33	4	30	8.90	5.44	3	30	24	2	7	6	2	5
Max	9.49	4.27	4	24	8.13	3.52	0	17	18	5	0	16	35	39	9.69	5.31	4	30	8.05	3.52	0	17	21	1	1	16	54	56
Cov	8.74	3.45	4	23	8.03	3.54	3	22	19	10	1	9			9.67	5.31	4	30	18,00	6,00	0	15	18	6	0	15		4
context enhanced																												
Raw	9.05	5.79	4	39	7.56	3.6	3	21	30	3	2	4		27	9.67	5.31	4	30	9.15	5.41	3	30	18	6	3	12	2	5
Min	8.72	3.83	4	23	7.72	3.7	3	22	35	0	0	4	1	5	9.72	5.33	4	30	8.92	5.46	3	30	24	2	6	7	2	5
Max	9.54	4.22	4	25	8.03	3.29	0	17	18	6	0	15	37	41	9.67	5.31	4	30	7.82	3.54	0	16	18	5	0	16	56	60
Cov	8.33	3.16	4	20	7.59	3.22	3	19	18	10	1	10		2	9.67	5.31	4	30	9.00	5.44	3	30	17	7	0	15	1	5

perspective, the **RQ2** is not satisfied since the overall quality of the DFG model generated with the **Incremental** setting is lower concerning adopting the **GoldStandard** setting.

Parameter 3 *Exploitation of contextual-domain information.* The usage of the problem-context information in prompts has a direct positive effect on the DFG model structure predicted. It seems to have the effect of making the overall DFG model structure more robust. Indeed, by looking at the node statistics, we may see how the number of nodes increases, while the standard deviation decreases. A different trend is found in the edge statistics that show a reduction in both the number of edges and the standard deviation. By analyzing the type of DFG model's structure, the two extraction context settings show different trends. In the incremental extraction context, only the *Raw Prompt* benefits from the problem-context information with a reduction of the number of isolated nodes. In the **GoldStandard** extraction context, the addition of contextual information increases the generation of more complex DFG model structures and, at the same time, reduces the number of linear DFG models. Finally, concerning **RQ3**, we may conclude that from the qualitative perspective, the injection of contextual information provides some light differences within the structure of the DFG model generated. Hence, the injection of contextual information may still be an interesting research direction to explore.

6 CONCLUSION

In this paper, we provide a complementary perspective on the generation of a process model out of process description by analyzing in a qualitative manner, the nodes, the edges, and the overall structure of the DFG models generated from the predictions performed within tasks **T1** and **T2**. This analysis would help to understand the strengths and limitations of our approach. We look at these aspects since they represent the basic building blocks to start to build up process models, i.e., *a directly-follows graph*, out of the LLM predictions.

Our analysis highlights that the quality of the task examples shown to LLMs greatly impacts the quality of the results. Providing minimal examples to extract process models is not an effective strategy. Providing a high-quality set of examples that try to cover somehow all the possible cases turned out to be an efficient strategy, especially for what concerns the extraction of temporal relations between activities. The quality of the predicted conceptual model (DFG model) is strongly connected to the quality of the data we provide to the language model. The injection of problem-context information has a large impact on a zero-shot learning setting, no significant effect on in-context learning settings, but it has a little positive effect on the quality of the conceptual temporal relation predicted. We cannot find the final answer to this point, and we leave an in-depth understanding of the impact of injecting specific context information into LLMs for future research.

Overall, the experimental manipulations provided an in-depth understanding of the impact of the information injected while shedding light on limitations and new challenges, opening up interesting research directions to explore. We noted a text limit problem that hampered the results. However, the advent of the new version of the GPT model, should solve this problem once and for all and it is expected to generate better process models.

REFERENCES

[1] Lars Ackermann and Bernhard Volz. 2013. model[NL]generation: natural language model extraction. In *Proc. of 2013 ACM workshop DSM@SPLASH 2013*. ACM, 45–50.

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007 (Lecture Notes in Computer Science)*, Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux (Eds.), Vol. 4825. Springer.

[3] Patrizio Bellan, Mauro Dragoni, and Chiara Ghidini. 2020. A Qualitative Analysis of the State of the Art in Process Extraction from Text. In *Proceedings of the AIxIA 2020 Discussion Papers Workshop co-located with the the 19th International Conference of the Italian Association for Artificial Intelligence (AIxIA2020), Anywhere, November 27th, 2020 (CEUR Workshop Proceedings)*, Vol. 2776. CEUR-WS.org.

[4] Patrizio Bellan, Mauro Dragoni, and Chiara Ghidini. 2022. Assisted Process Knowledge Graph Building Using Pre-trained Language Models. In *AIxIA 2022 - Advances in Artificial Intelligence - XXIst International Conference of the Italian Association for Artificial Intelligence, AIxIA 2022, Udine, Italy, November 28 - December 2, 2022, Proceedings (Lecture Notes in Computer Science)*, Agostino Dovier, Angelo Montanari, and Andrea Orlandini (Eds.), Vol. 13796. Springer, 60–74. https://doi.org/10.1007/978-3-031-27181-6_5

[5] Patrizio Bellan, Mauro Dragoni, and Chiara Ghidini. 2022. Extracting Business Process Entities and Relations from Text Using Pre-trained Language Models and In-Context Learning. In *Enterprise Design, Operations, and Computing - 26th*

International Conference, EDOC 2022, Bozen-Bolzano, Italy, October 3-7, 2022, Proceedings (Lecture Notes in Computer Science), Vol. 13585. Springer.

[6] Patrizio Bellan, Han van der Aa, Mauro Dragoni, Chiara Ghidini, and Simone Paolo Ponzetto. 2022. PET: An Annotated Dataset for Process Extraction from Natural Language Text Tasks. In *Business Process Management Workshops - BPM 2022 International Workshops, Münster, Germany, September 11-16, 2022, Revised Selected Papers (Lecture Notes in Business Information Processing)*, Vol. 460. Springer.

[7] Tom B. Brown and et al. 2020. Language Models are Few-Shot Learners. In *Annual Conf. on Neural Information Processing Systems 2020, NeurIPS 2020.*

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of NAACL-HLT 2019, Volume 1*. ACL.

[9] Elena Viorica Epure, Patricia Martín-Rodilla, Charlotte Hug, Rébecca Deneckère, and Camille Salinesi. 2015. Automatic process model discovery from textual methodologies. In *9th IEEE International Conference on Research Challenges in Information Science, RCIS 2015*. IEEE, 19–30.

[10] Renato César Borges Ferreira, Lucinéia Heloisa Thom, and Marcelo Fantinato. 2017. A Semi-automatic Approach to Identify Business Process Elements in Natural Language Texts. In *ICEIS 2017 - Proc. of the 19th Int. Conf. on Enterprise Information Systems, Volume 3*. SciTePress, 250–261.

[11] Fabian Friedrich, Jan Mendling, and Frank Puhlmann. 2011. Process Model Generation from Natural Language Text. In *Advanced Information Systems Engineering - 23rd International Conference, CAiSE 2011. Proceedings (LNCS)*, Vol. 6741. Springer, 482–496.

[12] Xue Han, Lianxue Hu, Lijun Mei, Yabin Dang, Shivali Agarwal, Xin Zhou, and Pengwei Hu. 2020. A-BPS: Automatic Business Process Discovery Service using Ordered Neurons LSTM. In *2020 IEEE International Conference on Web Services, ICWS 2020*. IEEE, 428–432.

[13] Krzysztof Honkisz, Krzysztof Kluza, and Piotr Wisniewski. 2018. A Concept for Generating Business Process Models from Natural Language Description. In *Knowledge Science, Engineering and Management - 11th INt. Conf., KSEM 2018, Proc., Part I (LNCS)*, Vol. 11061. Springer, 91–103.

[14] Jinhao Ju, Deqing Yang, and Jingping Liu. 2022. Commonsense Knowledge Base Completion with Relational Graph Attention Network and Pre-trained Language Model. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, Mohammad Al Hasan and Li Xiong (Eds.). ACM.

[15] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed Prompting: A Modular Approach for Solving Complex Tasks. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net. https://openreview.net/pdf?id=_nGgzQjzaRy

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv* abs/1907.11692 (2019).

[17] Bilal Maqbool, Farooque Azam, Muhammad Waseem Anwar, Wasi Haider Butt, Jahan Zeb, Iqra Zafar, Aiman Khan Nazir, and Zuneera Umair. 2018. A Comprehensive Investigation of BPMN Models Generation from Textual Requirements - Techniques, Tools and Trends. In *Information Science and Applications - ICISA 2018 (LNEE)*, Vol. 514. Springer, 543–557.

[18] José-Lázaro Martínez-Rodríguez, Aidan Hogan, and Ivan López-Arévalo. 2020. Information extraction meets the Semantic Web: A survey. *Semantic Web* 11, 2 (2020), 255–335.

[19] Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proc. of the 57th Conf. of the Association for Computational Linguistics, ACL 2019, Volume 1: Long Papers*. ACL, 3428–3448.

[20] Chen Qian, Lijie Wen, Akhil Kumar, Leilei Lin, Li Lin, Zan Zong, Shuang Li, and Jianmin Wang. 2020. An Approach for Process Model Extraction by Multi-grained Text Classification. In *32nd Int. Conf., CAiSE 2020, Proc. (LNCS)*, Vol. 12127. Springer, 268–282.

[21] Luis Quishpi, Josep Carmona, and Lluís Padró. 2020. Extracting Annotations from Textual Descriptions of Processes. In *Business Process Management - 18th Int. Conf., BPM 2020, Proc. (LNCS)*, Vol. 12168. Springer, 184–201.

[22] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. OpenAI Blog. (2018).

[23] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[24] Kiran Prakash Sawant, Suman Roy, Srivibha Sripathi, François Plesse, and A. S. M. Sajeev. 2014. Deriving requirements model from textual use cases. In *36th Int. Conf. on Software Engineering, ICSE '14, Proc.* ACM, 235–244.

[25] Teven Le Scao and Alexander M. Rush. 2021. How many data points is a prompt worth?. In *Proc. of NAACL-HLT 2021*. ACL.

[26] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*. ACM.

[27] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? Probing for sentence structure in contextualized word representations. In *7th Int. Conf. on Learning Representations, ICLR 2019*. OpenReview.net.

[28] Han van der Aa, Josep Carmona, Henrik Leopold, Jan Mendling, and Lluís Padró. 2018. Challenges and Opportunities of Applying Natural Language Processing in Business Process Management. In *Proc. of 27th Int. Conf. on Computational Linguistics, COLING 2018*. ACL, 2791–2801.

[29] Han van der Aa, Claudio Di Ciccio, Henrik Leopold, and Hajo A. Reijers. 2019. Extracting Declarative Process Models from Natural Language. In *Advanced Information Systems Engineering - 31st Int. Conf., CAiSE 2019, Proceedings (LNCS)*, Vol. 11483. Springer, 365–382.

[30] Maxim Vidgof, Stefan Bachhofner, and Jan Mendling. 2023. Large Language Models for Business Process Management: Opportunities and Challenges. In *Business Process Management Forum - BPM 2023 Forum, Utrecht, The Netherlands, September 11-15, 2023, Proceedings (Lecture Notes in Business Information Processing)*, Chiara Di Francescomarino, Andrea Burattin, Christian Janiesch, and Shazia W. Sadiq (Eds.), Vol. 490. Springer, 107–123. https://doi.org/10.1007/978-3-031-41623-1_7

[31] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*. http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html

[32] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. *CoRR* abs/2305.10601 (2023). https://doi.org/10.48550/arXiv.2305.10601 arXiv:2305.10601