Process Knowledge Extraction and Knowledge Graph Construction Through Prompting: A Quantitative Analysis*

ABSTRACT

The automated construction of process knowledge graphs from process description documents is a challenging research area. Here, the lack of massive annotated data, as well as raw text repositories describing real-world process documents, makes it extremely difficult to adopt deep learning approaches to perform this transformation. Indeed, the main challenge is to extract conceptual elements representing the actual entities or relations of the process model described within its corresponding natural language document. Large Language Models (LLMs) have shown promising results in supporting the extraction of structured knowledge from unstructured texts. Although several works explored this strategy to build or complete knowledge graphs, the exploitation of LLMs toward domain-specific knowledge base construction from scratch has not yet been investigated deeply. Our aim is to exploit the LLM capabilities to extract process knowledge from unseen natural language descriptions. In this work, we present a prompt-based in-context learning strategy to extract, from process descriptions, conceptual information that can be converted into their equivalent knowledge graphs. Such a strategy is performed in a multi-turn dialog fashion. We validate the accuracy of the proposed approach from a quantitative perspectives. The results highlight the feasibility of the proposed approach within our low-resource scenarios and open interesting perspectives for future activities.

KEYWORDS

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

ACM Reference Format:

. 2024. Process Knowledge Extraction and Knowledge Graph Construction Through Prompting: A Quantitative 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

1 INTRODUCTION

Textual descriptions of business processes, contained for example in Standard Operating Procedure (SOP) documents, are ubiquitous in almost all types of organizations. These documents typically describe how a procedure or a process is performed in a company, e.g., the process to handle a customer claim. While the goal of these descriptions is that of being easy to understand and use, the actual exploitation of the information they contain is often hampered by having to manually analyze unstructured information. The automatic creation of domain-specific knowledge graphs (KG)s from this type of text would facilitate the usage of advanced reasoning and query techniques, thus boosting the actual exploitation of the knowledge they contain. This goal is hampered by several challenges. First, the ambiguous nature of natural language, the multiple possible writing styles, and the great variability of possible domains where business process descriptions exist. Second, the lack of high quantities of carefully annotated data on textual descriptions of business processes, which makes almost impossible the exploitation of modern deep learning natural language processing (NLP) techniques. Third, the multidimensional nature of the entities and relationships involved, which range from temporal elements, like activities, and their ordering relationships, to the data manipulated by the activities, to organizational elements such as actors, or resources, and their relationships with the activities they connect with. Fourth the conceptual nature of these elements, as business process descriptions typically describe a process "model" and not a specific process execution. The relevant entities contained within these texts are common sense terms that can assume relevant conceptual meaning within particular contexts, but not in others, and the construction of KGs by starting from this type of information may be difficult.

Recent advances in NLP, like the availability of *Large Language Models* (LLMs) and the introduction of novel *in-context learning* approaches, enable the possibility to mimic few-shot learning techniques without changing any model parameter [5, 23]. These advances have recently fostered a few attempts to use LLMs for the construction/completion of KGs from text [24].

In this paper, we explore the feasibility of using *in-context learning* to perform knowledge extraction from textual descriptions of a specific type of procedural knowledge, that is, business processes, in a question-and-answer multi-turn dialog fashion, following an approach recently introduced in [3]. An example of the multi-turn dialog approach we aim to implement is shown in Figure 1. In this work, we use the Generative Pre-trained Transformer 3 model (GPT-3) [5] as an artificial agent. It is important to highlight that the aim of this work is not to compare different LLMs, but to explore the suitability of LLMs for extracting domain-specific KGs representing process knowledge, and in particular to explore different settings of the adopted LLM to perform in-context learning, e.g., by providing the problem-context information to prompts and by providing an extremely limited number of examples. The different settings and experimental designs are illustrated in Section 4, while the results are illustrated in Section 5.

The contribution of this paper, therefore, is an in-depth exploration of the capabilities of LLMs to extract domain-specific conceptual knowledge from unseen descriptions of processes, and an in-depth understanding of how their effectiveness on specific tasks may change based on different settings (that is, information) provided as input. To the best of our knowledge, this in-depth exploration is performed for the first time in the literature and can pave

*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

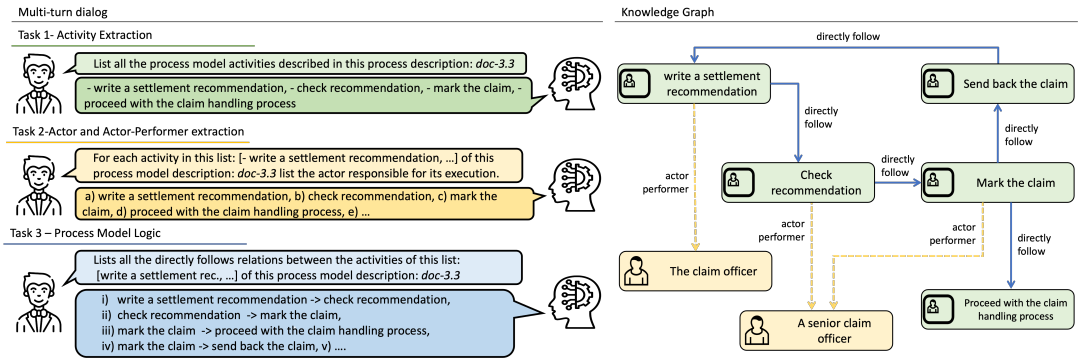

Figure 1: In this figure we show an example of an experimental scenario of our approach. The excerpt is taken from *doc-3.3* of the PET dataset (Section 4.1). In this multi-turn dialog, the artificial agent (acting as domain expert) guides the construction of the process knowledge graph by answering the user. The user poses specific questions to extract the conceptual elements and guide the construction of the corresponding KG incrementally. In the first task, the user asks for the list of activities of a process document to add activity nodes (green squares) in the KG. In the second step, the user asks the domain expert for the list of actors who perform/are responsible for the activities execution to add actor nodes (yellow squares) and actor-performer relation (yellow dashed arrows) to the KG. In the last step, the user asks for the set of temporal relations between the activities and uses the answer to add direct-follow relations (blue arrows) to the KG.

the way to address constructing KGs which incorporate a particularly challenging, and so far, neglected type of knowledge that is one of the processes (or procedures) in organizational settings.

2 RELATED WORK

The information extraction research area has been widely explored in the literature embracing many domains [20]. Specifically on the use of LLMs, several works investigated their use aiming to understand both linguistic and semantic properties of possible word representations and also how LLMs can be exploited within specific knowledge and linguistic tasks. Compared to the aims mentioned above, our approach goes against the trend by trying to exploit LLMs to extract and store factual and common-sense knowledge with the aim of constructing KGs automatically. However, the adoption of LLMs has been investigated from several perspectives. A systematic comparison between neural-based and count-based representations has been performed in [2]. Neural-based representations were demonstrated to be more effective with respect to the count-based ones in most of the tested tasks. Hence, given the neural-based nature of LLMs, they may be considered a suitable starting point for the purposes of this work. Details about the required granularity of these representations have been investigated, instead, in [14].

Large language models started to be exploited also concerning the construction and completion of knowledge bases [16] and provide fresh hope for the synergy of automated approaches and high-precision curated knowledge bases. Such a research field has a considerable history starting from human-curation approaches like the one proposed within the seminal CYC project [18]. Such an approach represented also the backbone of today's most prominent public knowledge bases like Wikidata [30]. Another popular paradigm is the extraction from semi-structured resources, as pursued in Yago [28] and DBpedia [1]. Knowledge-extraction from LLMs provides remarkably straightforward access to very large text corpora [21] and a range of follow-up works focused on investigating entities, improving updates, exploring storage limits, and incorporating unique entity identifiers [6, 26]. However, all the works mentioned above suffer from gaps related to the precision of the knowledge bases generated by LLMs with respect to the actual knowledge expected.

Such a limitation has been tackled by trying to exploit the implicit knowledge contained within LLMs. Indeed, while it is easy to query knowledge in knowledge bases, the implicit knowledge contained in LLMs is difficult to access. Examples of strategies to enhance the retrieval of implicit knowledge from LLMs rely on fine-tuning [8, 27] or prompt tuning [15, 22] with existing knowledge. Together with the success of LLMs, recent work has developed various ways to understand their mechanisms, such as analyzing the internal states of LLMs, and extracting structured linguistic patterns [13, 29]. Factual probing aims to quantify the factual knowledge in LLMs, which is usually implemented by prompting methods and leveraging the masked language model pre-training task. Specifically, the amount of factual knowledge is estimated by a set of human-written close-style prompts, e.g., "Dante was born in ". The accuracy of the model prediction on the blank represents a lower bound of the amount of knowledge in the model. LAMA [21] collects a set of human-written prompts to detect the amount of factual information that a masked language model encodes. LPAQA [15] proposes to use text mining and paraphrasing to find and select prompts to optimize the prediction of a single or a few correct tail entities, instead of extensively predicting all the valid entity pairs like in our framework.

Consistency is a significant challenge in knowledge probing and extraction, which refers to a model that should not have predictions contradicting each other. Basically, models should behave invariant under inputs with different surface forms but the same meaning, e.g., paraphrased sentences or prompts. Several benchmarks are proposed to study consistency in LLMs [10]. Such a work

analyzes the consistency of LLMs with respect to factual knowledge. They show that the consistency of all LLMs is poor in their experiment.

The only work that aims at extracting a KG for process description documents, in an incremental conversational manner, is that of [3]. We differ from this work in a substantial manner. First, we assess our approach on a broad set of documents and use different representative documents as training examples within the prompts. Then, we provide an extensive evaluation of how four different settings can influence the performance of LLMS in a quantitative manner. Therefore the results and lessons learned presented in this paper are likely to pave the way for future efforts, possibly involving different strategies, target entities and relations, and also other LLMs. Our approach is similar to the Decomposing Prompting approach [17] since we decompose the high-level task of extracting conceptual process knowledge information from text into three smaller sub-tasks (**T1**,**T2**,**T3**). The sub-tasks are in essence intermediate states of the problems. There are no intermediate states to reason from and the sub-tasks cannot be decomposed anymore. Our focus targets the evaluation of sub-tasks in isolation to understand the complexity of each sub-task, the limitations, the errors, and the challenges. Moreover, we cannot apply Chain-of-Thought (CoT) [31] or Tree-of-Thoughts (ToT) [32] strategies since we do not perform research (either breadth-first or depth-first) of entities in the text and we do not reason on, i.e., a single question (e.g., a math question) in isolation, neither an answer depends on a previous one. These techniques would have come in handy if our aim was to solve the overall task in one step.

3 PROCESS KNOWLEDGE GRAPH CONSTRUCTION FROM TEXT VIA IN-CONTEXT LEARNING

The task we intend to solve is the generation of process knowledge graph from business process descriptions (see Figure 1). A process knowledge graph is a domain-specific type of KG that stores the basic building blocks information needed to build up the equivalent business process model diagram. This process model diagram is vital in industry since it allows for analyses of a business process's time, costs, and errors. In this Section, we introduce the approach we use to achieve our goal and a description of how we implemented the approach.

3.1 The approach: LLM and In-context learning

The advent of the GPT-3 [5] LLM changed the NLP paradigm about performing fine-tuning on a Pre-Trained Language Model (PLM) for task-specific applications. This model opened the possibility of refining its reasoning ability by providing examples of the task to solve together with the text it has to analyze directly in input, without doing a canonical fine-tuning of the model parameters toward a single specific downstream task. This technique, called **in-context learning**, has been shown to be extremely useful to manage the low-resource issue [25] and have been used to address topics ranging from medical dialogue summarization [7] to hate speech detection [12]. GPT-3 showed the ability to understand utterances and instructions written in natural language and generate task-related answers in a human-like fashion. This LLM is also able to analyze a large portion of text at once, which is usually sufficient to analyze an entire document. Other transformer-like models such as BERT [9] or RoBERTa [19] can not handle large input, such as a process description and, they could not be adapted in real-world scenarios. Since our experimental domain suffers from (*i*) large input, and (*ii*) low-resource issues, we decided to adopt in our investigations the GPT-3 LLM combined with *in-context learning* technique.

1. **Q:** List all the process model activities described in this process description: *doc-1.4 text*
2. ***A:** a) purchase a product or service, b) submit the expense report, ...*
3. **Q:** List all the process model activities described in this process description: *doc-5.4 text*
4. **A:** *a) make the decision to go public, b) select underwriters, ...*
5. **Q:** List all the process model activities described in this process description: *doc-3.3 text*
6. **A:**

Figure 2: The figure shows an example of the *Max* in-context learning prompt for task T1. The blue line marks the in-context learning part of the prompt where we provide the two gold standard examples texts together with task instruction (lines 1 and 3) and the list of gold standard activities of the texts (lines 2 and 4). 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.

3.2 Implementing the approach

The starting point of our approach is the set of conceptual elements and relationships we aim to extract from the textual documents. Since it has been proved that GPT-3 is able to understand task instructions, we formulated a series of incremental questions to pose to the LLM to extract our target entities and relationships. We designed three specific questions (see Figure 3) to enable the extraction of *activities* (Q1), *actors* and their *performs* relations (Q2), and the *directly-follows* relations (Q3). The questions become the specific task instructions we provide to the LLM. The next step is the construction of the input, called *prompt*, to feed the LLM. In Figure 2, we show an example of a prompt template we customized in our experiments to enable in-context learning. Before inputting a prompt to the LLM to generate predictions, we fulfilled a prompt template with the proper set of examples, the specific *task instruction* together with the text to analyze. Finally, prompts are fed into the model to generate the answer. We use the information provided with answers to generate the KG. As a side now, we want to highlight that recently the literature on prompt-based fine-tuning started to grow, e.g. in [11]. Here, we want to remark that a global

Q1: *List all the process model activities described in this process description: [PROCESS DESCRIPTION];*
Q2: *For each activity in this list: [activity-list] of this process model description: [PROCESS DESCRIPTION] list the actor responsible for its execution. If the text does not describe any actor responsible for the execution, answer "NOT DEFINED".*
Q3: *Lists all the directly follows relations between the activities of this list: [activity-list] of this process model description: [PROCESS DESCRIPTION]*

Figure 3: The Task instructions.

investigation about the most efficient prompt is out of the scope of this paper.

4 EMPIRICAL ASSESSMENT

In this work, we explore the suitability of LLMs for extracting KG representing process knowledge, that is knowledge about procedures in organizational settings, from natural language text.

Differently from the literature related to the exploitation of LLMs for extracting factual knowledge directly from the LLMs, in our task, we do not extract the common-sense entities, e.g., named entities that may be linked to a Wikipedia page, and their relations described in a text. Here, our aim is to extract entities and relations having specific conceptual meanings with respect to the scenario described in each document. Examples are verbs representing process activities and names representing the actors performing a specific activity. While, concerning the exploitation of LLMs, in this work, we do not intend to explore the knowledge they already have. Instead, we aim to exploit their capabilities to extract process knowledge from unseen natural language descriptions of procedures. Thus, the final objective of this work is to provide an answer to the meta-research question: *Are LLMs good candidates to support the construction of domain-specific knowledge graphs from texts in a low-resource scenario?*.

Obviously, assessing the quality of LLMs for this specific task is a broad and multi-faceted task that cannot be addressed in a single paper. Here, we may split such a research question into investigating the role of four different types of settings of the adopted LLM, described in Section **??**, to perform the task of our domain in an in-context learning fashion. This gives rise to the following four research questions:

RQ1: Type of prompt template. *Does the quality of the examples provided inside the prompts to enable in-context learning affect the quality of the results?*

RQ2: KG build strategy. *Within our incremental approach, does the LLM generate better KGs 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 problem-context information during the extraction of procedural knowledge?*

RQ4: Evaluation of textual output. *What is the role of different evaluation metrics on the results generated by the LLM?*

4.1 The PET Dataset

We adopted the PET dataset [4] to perform the experiments. Composed of process description documents annotated with process elements and their relations, it is the unique gold-standard dataset specific for process information extraction tasks.

We briefly introduce the process elements and relations of the PET that we use in our experiments, i.e., Activity, Actor, Performer relation, and Directly-Follows relation. An "activity" in BPM represents a single task performed during the execution of a process model. Within the PET dataset, the annotation of an activity follows a pattern where it has been broken down into small components, i.e. the activity verb and the corresponding activity data. For instance, consider the example in Figure 1, the activity *write a settlement recommendation* is annotated in PET as the composition of the activity verb write with the activity data a settlement recommendation by the *uses* relation. A PET "actor" defines a process participant involved somehow in activity and the *perform relation* provides the information about *who is the actor responsible for activity execution* by linking the actor to the corresponding activity (e.g., write a settlement recommendation → the claim officer). Then, a "directly-follows" relation is a process model element representing the temporal relations (i.e., the arrow in BPM) between two process activities (e.g., write a settlement recommendation → check recommendation).

In our experiments, we converted the PET annotations of the documents into their equivalent KGs. Activities and actors performing the activities are described by means of RDF classes while actor-performer and directly-follow relations are described with an object property. This way, we may compare the gold-standard graph manually created and the one built with the knowledge automatically extracted in our experiment from each text.

4.2 The Tasks and the Experimental Setting

Our approach addresses three different tasks (steps in the conversation): (**T1**) the activity extraction; (**T2**) the extraction of who performs an activity given the activity itself, and, (**T3**) the detection of directly-follows relations. For each task, we created a set of prompt templates supporting the knowledge extraction operation from the LLMs. In **T1** we filled the task instruction place-holder in prompt templates with question **Q1** (Figure 3), in **T2** we use **Q2**, and in **T3** we use **Q3**. Finally, we generated the KG from the answers after their validation. Since the answers generated have pretty much the same structure, with very few variations, we created a simple script to extract the KG information from the answers [1].

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. Here, we want to remark that the comparison among different model engines and parameter configurations is out of the scope of this paper and it is planned for a future extension. Differently from other domains, here it is not possible to adopt any standard NLP metrics to judge the similarity between predictions (extracted from the answers) and the gold standard since in some cases, even a little rewording can change dramatically the meaning conveyed in a process model that may give rise to negative consequences in real-scenario. Therefore, to provide a solid and rigorous evaluation, we manually analyzed all the predictions and discussed the extreme cases.

The exploration we performed involved the combination of different parameters defined to enable a deeper understanding of LLM capabilities within this specific task. We defined four parameters: (i) the type of prompt template adapted to query the LLM; (ii) the way with which we build the KG; (iii) the exploitation of contextual-domain information or not; and, (iv) the way we evaluate the textual output produced by the LLM. All these parameters are explained in more detail below.

Parameter 1: Type of prompt template. The first parameter is the type of prompt template we adopted for performing our experiments. We designed in total four prompts: a baseline prompt template (***Raw Prompt***) and three in-context learning prompts enhanced with task examples: ***Min Prompt***; ***Max Prompt***; and ***Cov Prompt***. When using the ***Raw Prompt*** setting, we did not provide any example of the task to solve within the prompts. This way, we

[1] the material can be found at https://anonymous.4open.science/r/Process-Knowledge-Extraction-and-Knowledge-Graph-Construction-Through-Prompting-2055/

were able to test the ability of the bare LLM to solve the tasks and to have a baseline to compare with the other three in-context learning prompts. When using the ***Min Prompt***, we provided as input the two shortest documents of the PET dataset (*doc-8.1* and *doc-10.13*) as task examples. These two examples present a linear structure in their process representation with no split or merging points. Then, we tested the opposite scenario with the ***Max Prompt*** by providing, instead, the two longest documents of the PET dataset (*doc-2.1* and *doc-4.1*) as task examples. These two samples present complex graph structures with split and merging points. Differently from the two previous experimental prompts, where we do not make any constraints about the process model structure, in the ***Cov Prompt*** we instead consider it. We selected the documents *doc-1.4* and *doc-5.4* as task examples for the in-context learning operation. These documents are two longer ones having the maximum coverage of process elements of the PET dataset where at least a split point is present. The first document has a split and a merging point in its process model representation, while the second one presents two open split points. Hence, in summary, our training sets to perform the in-context learning operation are composed of 6 documents used as task examples in the three experimental prompts. Our test set is the set of all the other 39 documents not used for training.

Parameter 2: KG build strategy. The second parameter of the experimental setting regards the knowledge extraction process we adopted to build the KGs. We designed two settings: **Incremental** and **GoldStandard**. The **Incremental** setting corresponds to a knowledge extraction process model in which the KGs are built incrementally. Here, the input of a task *n* is the result of the task *n-1*. In practice, we use the knowledge extracted during the execution of **T1** to fulfill the *list of activity in the prompt*-templates of **T2** and **T3**. One of our aims is to observe how much such an incremental procedure, better reflecting a real-world setting, is effective. Instead, within the **GoldStandard** experimental setting, we want to assess the performance of also **T2** and **T3** without being dependent on the outcome of **T1**. Here, we provided both the gold-standard activity list and the gold-standard examples to prompts. Finally, for both settings, we use the predictions to generate the KG.

Parameter 3: Usage of context. The third parameter of the experimental setting is the context information that may be provided to prompts. We designed two experimental settings: *not context enhanced* and *context enhanced*, to test the hypothesis that *adding in prompts context information about the problem to solve, it helps the LLM to narrow its "reasoning ability" toward the specific problem*, and thus providing better predictions than without this information. In *not context enhanced*, we do not alter the prompts while in *context enhanced* we added the *context information* about the domain of the task to solve, at the beginning of the prompts. In our scenario, we instructed the LLM to consider the context of Business Process Management.

Parameter 4: Evaluation of textual output. The last experimental setting parameter is related to how we validated the information generated by the LLM. We do not want to limit the comparison to pure matching. Instead, we want to provide a larger picture of the actual performance of the proposed approach. For this reason, we relax the comparison by designing four textual output evaluation settings.

Strict. In this setting, we strictly compare predictions with gold-standard data. We do not allow any exceptions.

Relaxed. In this evaluation setting, we allow for small variations in answers that preserve their semantic meaning in the process model described. In **T1**, we considered valid predicted activities that present one or more of the errors described in Figure 4. In **T2** we consider the list of relaxable activities from **T1** while we also allow for little rewording of the actor or, in the case of pronouns, the resolution of the anaphoric reference that has a perform relation with the relaxable activity.

Split Since some activities in the PET dataset present double activity data (e.g., *sends the forms and the documents*) the LLM may generate semantic equivalent predictions of the same activity with the different activity data (e.g., *send the form* and *sends the documents*). In order to deal with this situation, in this setting, we consider them a single activity if all the activities predicted are correct.

Compound. Differently from what happens in the **split** setting, here we judged as a correct activity the conjunction of two activities that, from the procedural perspective, may be compounded. For example, by giving the following two activities within the gold-standard *checks the forms* and *sends the forms*, the compound activity *checks and sends the forms* returned by the LLM has been judged as correct.

In summary, we performed 64 experimental items as the results of the combination of all possible values of the four parameters, i.e., 4 types of prompt, 2 KG building strategies, 2 context enhancement settings, and 4 textual output evaluations.

5 QUANTITATIVE EVALUATION

In this section, we present the outcome of the quantitative evaluations related to the three extraction tasks (i.e., **T1**, **T2**, and **T3**) and of the KG generation task from process descriptions. Through this analysis, we intend to highlight the performance of each prompt and to put the lights on possible limitations and new challenges that will feed future activities. Table 1 provides the results of such analysis. For each evaluation setting, we report the task's scores using the well-known metrics of Precision, Recall, and F1 Score. Then, we wanted to compare the structure of the KGs generated by starting from the process elements extracted from the LLMs. To do that, we applied the Graph Edit Distance (GED) algorithm to measure the similarity between two graphs by calculating the minimum number of operations needed to transform one graph into another. By calculating the GED, it is possible to determine

Added Recipient: The predicted activity contains an actor and the actor is the correct actor recipient.
Added Performer: The predicted activity contains an actor and the actor is the correct actor performer.
Missing part of the Activity Data: The predicted activity contains the "core" part of the PET activity data, but it misses some details.
Added no AD part: The activity data of a predicted activity contains information that is not annotated as such in the PET.
Added Further Spec.: The predicted activity contains the PET Further Specification element.
Rewording: The predicted activity is a rewording of the gold activity.

Figure 4: The list of activity *relaxable errors*

Table 1: Experimental Results. The first row of the headers reports the *KG build strategy* parameter. Next, for each *type of prompt template* parameter, the table reports the *evaluation of textual output* parameter in the first column. Then, the table reports the extraction scores for each task and the Graph Edit Distance (GED) of the predicted graph to the gold-standard one for the *usage of context* parameter. The bold font marks the best score for each evaluation setting in each task; the symbol * marks the best score overall.

	Incremental extraction context										**GoldStandard** extraction context						
	T1			**T2**			**T3**				**T2**			**T3**			
Evaluation	pr.	rc.	F1	pr.	rc.	F1	pr.	rc.	F1	GED	pr.	rc.	F1	pr.	rc.	F1	GED
***not context enhanced* setting**																	
Raw Prompt																	
strict	0.02	0.02	0.02	0.00	0.00	0.00	0.00	0.00	0.00	55.05	0.07	0.07	0.07	0.76	0.71	0.73	24.79
relaxed	0.78	0.70	0.73	0.60	0.54	0.56	0.46	0.40	0.42	25.44	0.26	0.26	0.26	0.76	0.71	0.73	21.26
split	0.79	0.70	0.73	0.60	0.54	0.56	0.47	0.40	0.43	25.23	0.26	0.26	0.26	0.76	0.71	0.73	21.26
compound	0.83	0.75	0.78	0.62	0.55	0.58	0.52	0.45	0.48	23.10	0.26	0.26	0.26	0.76	0.71	0.73	21.26
Min Prompt																	
strict	0.35	0.32	0.33	0.25	0.23	0.24	0.10	0.08	0.09	45.36	0.58	0.59	0.59	0.80	0.73	0.76	14.62
relaxed	**0.86**	**0.79**	**0.82**	0.75	**0.71**	**0.73**	0.57	0.50	0.53	19.62	**0.81***	**0.81***	**0.81***	0.80	0.73	0.76	**9.59***
split	**0.87**	**0.80**	0.83	0.76	**0.71**	0.73	0.59	0.51	0.54	18.90	**0.81***	**0.81***	**0.81***	0.80	0.73	0.76	**9.59***
compound	**0.90***	**0.83***	**0.86***	0.77	**0.72***	0.74	0.62	**0.54**	**0.57**	17.49	**0.81***	**0.81***	**0.81***	0.80	0.73	0.76	**9.59***
Max Prompt																	
strict	**0.48**	**0.46**	**0.47**	**0.33**	**0.32**	**0.32**	0.23	0.22	0.22	**38.38**	0.59	0.58	0.58	0.78	0.74	0.75	13.79
relaxed	0.78	0.76	0.77	0.68	0.66	0.67	0.57	0.53	**0.55**	21.59	0.77	0.75	0.76	0.78	0.74	0.75	10.82
split	0.80	0.77	0.78	0.70	0.67	0.69	0.60	**0.56**	0.57	20.08	0.77	0.75	0.76	0.78	0.74	0.75	10.82
compound	0.79	0.77	0.78	0.71	0.68	0.69	0.62	**0.58***	**0.60***	19.21	0.77	0.75	0.76	0.78	0.74	0.75	10.82
Cov Prompt																	
strict	0.43	0.40	0.41	0.30	0.27	0.28	0.21	0.18	0.19	41.59	**0.62**	**0.61**	**0.62**	**0.85***	0.77	0.80	**12.54**
relaxed	0.81	0.75	0.77	0.67	0.62	0.64	0.55	0.49	0.51	22.87	0.78	0.77	0.77	**0.85***	0.77	0.80	9.92
split	0.82	0.76	0.78	0.69	0.63	0.65	0.56	0.49	0.52	22.18	0.78	0.77	0.77	**0.85***	0.77	0.80	9.92
compound	0.83	0.77	0.80	0.69	0.63	0.66	0.58	0.51	0.53	21.33	0.78	0.77	0.77	**0.85***	0.77	0.80	9.92
***context enhanced* setting**																	
Raw Prompt																	
strict	0.02	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	55.44	0.07	0.07	0.07	0.79	0.74	0.76	24.69
relaxed	0.81	0.71	0.75	0.68	0.62	0.65	0.52	0.45	0.48	22.82	0.25	0.25	0.25	0.79	0.74	0.76	21.00
split	0.81	0.72	0.76	0.69	0.63	0.65	0.53	0.45	0.48	22.95	0.25	0.25	0.25	0.79	0.74	0.76	21.00
compound	0.83	0.74	0.78	0.70	0.63	0.66	0.56	0.48	0.51	21.59	0.25	0.25	0.25	0.79	0.74	0.76	21.00
Min Prompt																	
strict	0.37	0.33	0.34	0.27	0.24	0.25	0.12	0.10	0.11	43.85	0.60	0.60	0.60	0.80	0.72	0.75	14.67
relaxed	**0.86**	0.78	**0.82**	**0.76**	0.70	**0.73**	**0.59**	0.50	0.53	**19.08**	**0.81***	**0.81***	**0.81***	0.80	0.72	0.75	9.69
split	**0.87**	0.79	**0.83**	**0.77**	**0.71**	**0.74**	0.60	0.51	0.54	**18.46**	**0.81***	**0.81***	**0.81***	0.80	0.72	0.75	9.69
compound	**0.90***	**0.83***	**0.86***	**0.79***	**0.72***	**0.75***	**0.63***	0.53	0.57	**17.00***	**0.81***	**0.81***	**0.81***	0.80	0.72	0.75	9.69
Max Prompt																	
strict	0.46	0.45	0.46	0.32	0.31	**0.32**	**0.29**	**0.27**	**0.28**	38.51	0.60	0.58	0.59	0.80	0.73	0.75	13.54
relaxed	0.79	0.77	0.78	0.68	0.66	0.67	0.58	**0.54**	**0.55**	21.44	0.78	0.76	0.77	0.80	0.73	0.75	10.56
split	0.81	0.79	0.79	0.70	0.68	0.69	**0.61**	**0.56**	**0.58**	19.82	0.78	0.76	0.77	0.80	0.73	0.75	10.56
compound	0.82	0.80	0.81	0.71	0.68	0.69	0.62	**0.58***	**0.60***	19.03	0.78	0.76	0.77	0.80	0.73	0.75	10.56
Cov Prompt																	
strict	0.45	0.40	0.42	0.29	0.26	0.27	0.22	0.18	0.20	40.41	0.56	0.55	0.55	**0.85***	**0.78***	**0.81***	13.59
relaxed	0.81	0.73	0.76	0.68	0.61	0.64	0.57	0.48	0.51	22.10	0.77	0.76	0.76	**0.85***	**0.78***	**0.81***	9.90
split	0.82	0.73	0.77	0.69	0.62	0.65	0.58	0.49	0.52	21.64	0.77	0.76	0.76	**0.85***	**0.78***	**0.81***	9.90
compound	0.84	0.75	0.79	0.70	0.62	0.65	0.60	0.50	0.54	20.77	0.77	0.76	0.76	**0.85***	**0.78***	**0.81***	9.90

how different the two KGs are. The lower the value, the higher the similarity between the two graphs.

Given the high number of experimental settings we explored, we split our observations by the type of parameters we adopted.

Parameter 1: Type of prompt template. Among the four different prompt templates we implemented, we may observe that the *Min Prompt* generally outperforms the other three in most of the cases. This point can be appreciated especially for the tasks **T1** and **T2** in both the context-enhanced and not context-enhanced settings. While for task **T3**, it is the *Cov Prompt* that obtained the best results. In this case, our hypothesis is that, since the *Cov Prompt* provides input for the in-context learning step documents having a higher variety of process elements, the LLM is probably able to better learn how *directly-follows* relationships are defined. A further interesting aspect is the performance of the *Raw Prompt* on **T2** within the **GoldStandard** setting. Here, there is the only case in which the **GoldStandard** obtains a lower performance with respect to the **Incremental** one. This aspect is worthy of investigation in the future since the execution of both **T2** and **T3** within the **Incremental** setting suffers from the butterfly effect of the errors committed during the extraction of the process activities. Hence, it was expected that the performance obtained within the **Incremental** was, given each combination of the other parameters, lower with respect to the **GoldStandard** setting. Then, we observed that, independently of the prompt adopted, their enhancement with the context information (*context enhanced*) has a very low effect. Indeed, the differences for all metrics do not go beyond

a value of 0.02 points. Finally, it may be appreciated how the *Raw Prompt* registered a marked decreased score for the *strict evaluation of textual output* with respect to *relaxed*, *split*, and *compound* ones.

Given the obtained results, we can respond positively to **RQ1** since the adoption of different prompts led to different results as well. It demonstrates that such an exploration is useful for understanding how to increase the knowledge extraction performance of LLMs.

Parameter 2: KG build strategy. The comparison of the two possible values for this parameter can be performed only for **T2** and **T3**. Indeed, the difference between the two parameters is related only to the list of the activities used as input, i.e., the list of extracted activities for the **Incremental** setting and the list of the gold-standard activities for the **GoldStandard** setting. By analyzing the results related to **T2**, we may observe how, in general, the improvements between the two settings are always under 0.10 with two exceptions. The first exception is related to the behavior of the *Raw Prompt* that, as already described within the previous paragraph, registered lower scores for the **GoldStandard** setting with respect to the **Incremental** one. The second exception is related to the *strict evaluation of textual output* that for all settings but one (i.e., in *Raw Prompt*) registered an increment of around 100% between the **Incremental** and the **GoldStandard** settings. This aspect demonstrates the impact of the errors done in **T1** and, at the same time, the capability of the LLM to perform **T2**. Indeed, for the other three prompts (i.e., *Min Prompt*, *Max Prompt*, and *Cov Prompt*) the improvement is more limited given the high effectiveness obtained within **T1** thanks to the adoption of the more relaxed evaluations of the textual output.

What happened only for the *Raw Prompt* in **T2**, occurred generally for all prompts in performing the **T3**. Here, the increments of the effectiveness for all metrics range between 0.20 and 0.30. This result is a further demonstration of the impact of **T1**, especially on the capability of the LLM to extract the temporal aspects from the given texts.

This analysis leads to **RQ2** which answer is not totally positive since on the one hand the comparison between the **Incremental** and **GoldStandard** settings is in favor of the second one. However, in most of the cases, the adoption of the **Incremental** settings does not have dramatic detrimental effects on the overall performance of the approach. For this reason, the incremental building of KGs remains an interesting research direction to explore.

Parameter 3: Usage of context. By comparing the *context enhanced* and the *not context enhanced* scenarios, we may observe that there are no significant differences for all the scores reported in Table 1. The addition of the context information has a beneficial effect only for the *Raw Prompt*, while it has almost no-effect on the other ones. This is an important aspect because it puts the light on the facts that in the problem presented in this work, the provision of the basic notions of the BPM domain did not affect the capabilities of the LLM to extract relevant information. Given this result, future activities will focus on totally exploiting the prompt's size to store further samples useful for improving the in-context learning step. Indeed, prompt length turned out to be an important parameter to take under control in research since *text limit has a detrimental effect on the maximum text length that can be analyzed by an LLM using prompting and in-context learning techniques.*

Here, concerning **RQ3**, we may conclude that from the quantitative perspective, the injection of contextual information did not provide any differences in the knowledge extraction performance.

Parameter 4: Evaluation of textual output. The analysis of how the textual output is evaluated shows how the adoption of the *strict* evaluation method always led to worse performance. This is a straightforward consideration since the *strict* textual evaluation must expect a textual output produced by the LLM coinciding with the labels we normalized from the original documents. This is the reason for which we designed three further ways to evaluate the textual output. As expected, by relaxing the evaluation of the textual output, we may observe an important increment of the values on all the metrics for all of the tasks. Indeed, we may appreciate how there is a relevant difference between the performance of the *strict* textual evaluation and the *relaxed* evaluations. Instead, there is only a little performance increase between the *relaxed* evaluation and the *split* and *compound* ones.

Finally, by answering to **RQ4** we may state that the exploration of different evaluation strategies for the generated textual output allowed us to better understand which are the most common language variations that, even if the output does not correspond to the gold standard, did not lead to actual errors. Indeed, moving from the *strict* evaluation to the *relaxed* one performance of the approach relevantly changed.

Finally, an overall comment about the GED scores. Here, we may notice how there is a proportion between the GED scores and the other three metrics. Even if this is a trivial aspect, it is anyway a confirmation about specific criticism associated with particular settings. Obviously, this does not exclude that, at a more general level, a deeper correlation analysis may be performed. We left it as future work. Then, it is also more marked how the usage of the *strict evaluation of textual output* always led to very high GED scores for the **Incremental** setting with respect to the other three methods adopted for evaluating the textual output.

6 CONCLUSION

The quality of the task examples shown to LLMs has a large impact on the quality of the results. While it is sufficient to provide a minimal set of examples to extract conceptual knowledge, providing a high-quality set is useful to extract conceptual temporal relations. We conclude with a positive answer to our first research question. We answer negatively to our second research question since the quality and the quantity of the conceptual information extracted/predicted is strongly connected to the quality of the data we provide. The injection of problem-context information showed an interesting behavior. This information has a large impact on a zero-shot learning setting, no significant effect on in-context learning settings, and a little positive effect on the quality of the conceptual temporal relation predicted. Therefore, we cannot find a final answer to our third research question in this paper. We leave an in-depth understanding of the impact of injecting specific context information into LLMs for future research. Concerning the last research question, we demonstrated that opening up predictions to language variations has a relevant effect. Our contribution provided insights into the understanding of the possibilities and limits of LLMs. We choose the task of generating process knowledge graphs of processes from descriptions of procedures to explore the capabilities of LLM in extracting conceptual knowledge from

texts. The experimental manipulations provided an in-depth understanding of the impact of the information injected while shedding light on new and interesting research directions. As a future direction, we want to adopt the lessons learned in this paper and apply them to extract the control flow structures (decision points) of a process model described in a process described. Finally, we planned to overcome the limit regarding the use of GPT-3 by applying the newest LLMs available today and providing a solid comparison with this work.

REFERENCES

[1] 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.

[2] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, Volume 1: Long Papers*. The Association for Computer Linguistics, 238–247.

[3] 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)*, Vol. 13796. Springer.

[4] 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.

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

[6] Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing Factual Knowledge in Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). Association for Computational Linguistics.

[7] Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2021. Medically Aware GPT-3 as a Data Generator for Medical Dialogue Summarization. In *Proc. of the 6th Machine Learning for Healthcare Conf. (Proc. of Machine Learning Research)*, Vol. 149. PMLR.

[8] Jeff Da, Ronan Le Bras, Ximing Lu, Yejin Choi, and Antoine Bosselut. 2021. Analyzing Commonsense Emergence in Few-shot Knowledge Models. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*, Danqi Chen, Jonathan Berant, Andrew McCallum, and Sameer Singh (Eds.).

[9] 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.

[10] Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard H. Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and Improving Consistency in Pretrained Language Models. *Trans. Assoc. Comput. Linguistics* 9 (2021).

[11] Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. In *Proc. of ACL/IJCNLP 2021*. ACL.

[12] Sachin Gupta. 2022. Hate Speech Detection using OpenAI and GPT-3. *International Journal of Emerging Technology and Advanced Engineering* (2022).

[13] John Hewitt and Christopher D. Manning. 2019. A Structural Probe for Finding Syntax in Word Representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics.

[14] Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation. *Comput. Linguistics* 41, 4 (2015), 665–695.

[15] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How Can We Know What Language Models Know. *Trans. Assoc. Comput. Linguistics* 8 (2020).

[16] 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.

[17] 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

[18] Douglas B. Lenat. 1995. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Commun. ACM* 38, 11 (1995).

[19] 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).

[20] 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.

[21] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language Models as Knowledge Bases?. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics.

[22] Guanghui Qin and Jason Eisner. 2021. Learning How to Ask: Querying LMs with Mixtures of Soft Prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (Eds.). Association for Computational Linguistics.

[23] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.* 21 (2020).

[24] Danilo Neves Ribeiro and Kenneth D. Forbus. 2021. Combining Analogy with Language Models for Knowledge Extraction. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*.

[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] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics.

[27] Jaspreet Singh, Jonas Wallat, and Avishek Anand. 2020. BERTnesia: Investigating the capture and forgetting of knowledge in BERT. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online, November 2020*, Afra Alishahi, Yonatan Belinkov, Grzegorz Chrupala, Dieuwke Hupkes, Yuval Pinter, and Hassan Sajjad (Eds.). Association for Computational Linguistics.

[28] 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.

[29] Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics.

[30] Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014).

[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