A Universal Prompting Strategy for Extracting Process Model Information from Natural Language Text using Large Language Models

Julian Neuberger[1], Lars Ackermann[1], Han van der Aa[2], and Stefan Jablonski[1]

[1] University of Bayreuth, Bayreuth, Germany
firstname.lastname@uni-bayreuth.de
[2] University of Vienna, Vienna, Austria
han.van.der.aa@univie.ac.at

Abstract. Over the past decade, extensive research efforts have been dedicated to the extraction of information from textual process descriptions. Despite the remarkable progress witnessed in natural language processing (NLP), information extraction within the Business Process Management domain remains predominantly reliant on rule-based systems and machine learning methodologies. Data scarcity has so far prevented the successful application of deep learning techniques. However, the rapid progress in generative large language models (LLMs) makes it possible to solve many NLP tasks with very high quality without the need for extensive data. Therefore, we systematically investigate the potential of LLMs for extracting information from textual process descriptions, targeting the detection of process elements such as activities and actors, and relations between them. Using a heuristic algorithm, we demonstrate the suitability of the extracted information for process model generation. Based on a novel prompting strategy, we show that LLMs are able to outperform state-of-the-art machine learning approaches with absolute performance improvements of up to 8% F_1 score across three different datasets. We evaluate our prompting strategy on eight different LLMs, showing it is universally applicable, while also analyzing the impact of certain prompt parts on extraction quality. The number of example texts, the specificity of definitions, and the rigour of format instructions are identified as key for improving the accuracy of extracted information. Our code, prompts, and data are publicly available[3].

Keywords: Process Information Extraction · Large Language Models · AI-assisted Conceptual Modeling · Business Process Modeling

1 Introduction

In the field of Business Process Management (BPM), process models are established tools for designing, implementing, enacting, and analyzing enterprise processes [12]. However, the manual creation of these models is very time-consuming

[3] See https://github.com/JulianNeuberger/llm-process-generation/tree/er2024.

and accounts for around 60% of the total time spent on process management [16]. In order to reduce this effort, the automatic creation of these models based on a variety of information sources is a research focus in the field of BPM [5, 16]. In this respect, the paper at hand contributes to the extraction of process-relevant information from natural language information sources.

Information on organizational processes is frequently contained in a range of textual documents, such as process descriptions, rules and regulations, and work instructions [1, 4]. Recognizing this, a variety of techniques has been developed that aim to automatically extract process information from texts in order to subsequently turn it into process models [2, 8, 16]. This two-step procedure, in which information is extracted first and turned into a process model second, comes with several advantages in comparison to a direct text-to-model transformation approach: (1) The result quality can be evaluated with established means from the information-extraction domain, (2) extracted information can be transformed in more than one target process modeling language[4], and (3) it is possible to use extracted process information for other purposes such as, for instance, compliance checking, formal reasoning [3,26], and process querying [19].

The goal, scope, and challenges of information extraction depend on the input document type and content, as well as the desired output, i.e., the information to be extracted. Still, the extraction of process information from text generally involves: (1) the identification of textual mentions of process entities, such as activities, process participants, and business objects, and (2) relations between these entities, such as sequential dependencies, exclusivity, and assignments (e.g., who performs which step). As an illustration, Figure 1 shows a fragment of a textual description and two instantiations of the extraction task, focused on the information necessary for a model in Business Process Model and Notation (BPMN)[5] model [9] (upper part) and for declarative process modeling [2] (lower part). As shown, they involve different entities and relations, which each need to be inferred from the unstructured textual input.

A key problem is that the extraction of process information is still largely rule-based [23]. However, crafting useful rules is complicated, requires an extensive understanding of the process itself, and the rules are hard to transfer across organizations or text sources. To overcome this, recent work proposed the use of machine learning techniques [23], though these are hampered by data scarcity. Work that strives towards using pre-trained generative LLMs, e.g., GPT-3 [8] aims to circumvent this concern. However, the work in [8] only presents a preliminary study, with limitations in terms of analyzed datasets, extracted information, and result discussion. Therefore, this paper aims to provide deeper insights into the usability of LLMs for process information extraction and specifically includes the following core contributions: **(I)** It presents challenges that make the extraction of process-relevant information in particular a difficult task (Section 2). **(II)** As our main contribution, it proposes a novel, task-specific, and

[4] This is inspired by the paradigm of *interlingua-based machine translation* [27], which reduces the number of translation systems for n languages from n^2 to $2n$.

[5] https://www.omg.org/bpmn/, accessed June 2, 2024.

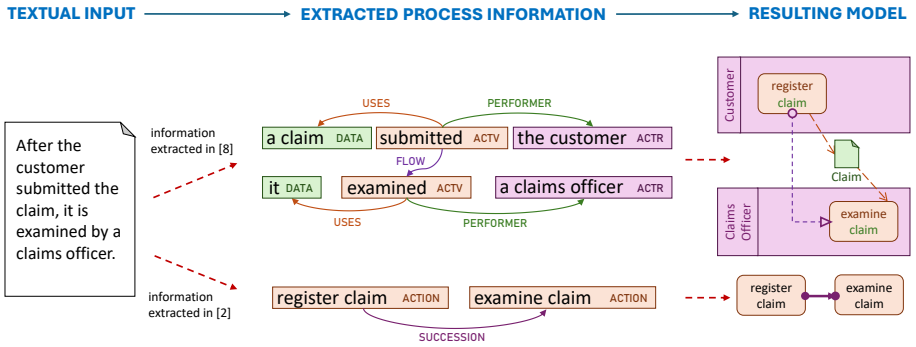

Fig. 1: Fragment of a larger text describing a business process of an insurance company. Different methodologies may extract different process relevant information, depending on the target modelling notation or use case.

rigorously empirically validated prompting strategy for solving the aforementioned information extraction tasks (Section 4). **(III)** It provides the currently most comprehensive study of using LLMs for extracting business process relevant information from natural language text (Sections 5 and 6). To this end we rigorously compare our prompting strategy on multiple datasets with state-of-the-art approaches and achieve up to 7% higher absolute F_1 scores compared to machine learning methods and up to 8% compared to rule-based methods. **(IV)** By testing our prompting strategy with eight state-of-the-art LLMs, we empirically demonstrate the generality of both our results and the applicability of our prompting strategy. **(V)** An ablation study (Section 6.2) shows that common best practices in prompt engineering are only of limited use for process information extraction. Thus, we also define guidelines for using LLMs for process information extraction (Section 6.4).

The rest of this paper is structured as follows. Section 2 describes the information extraction tasks and its challenges in detail. Section 3 summarizes the current state of the art in dealing with these tasks. After that we, describe our prompting strategy, a model generation algorithm, the experiments, and corresponding results (Section 4–6). Section 7 describes limitations and future work.

2 Task Descriptions and Challenges

In this section, we describe the three main (sub)tasks of process information extraction (Section 2.1), before highlighting a range of challenges associated with such extraction and with the use of LLMs for it (Sections 2.2–2.3).

2.1 Task descriptions

Our work focuses on three established subtasks of (process) information extraction from text: Mention Detection (MD), Entity Resolution (ER), and Relation Extraction (RE) [2, 7, 23, 26].

Mention Detection (MD) is concerned with finding and extracting text fragments that contain process relevant information, such as activities (or actions), process-relevant objects or data (i.e., business objects), or involved persons and departments (i.e., actors). For instance, in Figure 1, the upper example shows mentions of *data*, *actions*, and *actor*, whereas the lower one focuses on *activities*. This definition is similar to Named Entity Recognition (NER), though we also extract spans not covered by the traditional definition of NER, e.g., activities.

Entity Resolution (ER) aims to recognize when different mentions refer to the same process entity. For example, in Figure 1, successful ER would identify that the word *it* in "it is examined" corresponds to the *claim* mentioned in the previous phrase. Another common example is using ER to recognize that the same actor (across mentions) performs different steps. ER is a super-set of co-reference resolution and anaphora resolution [29] and is a crucial step when dealing with process-related texts, which frequently involve repeated mentions across sentences or even paragraphs [23].

Relation Extraction (RE) is the task of detecting and classifying relations between mentions. Relations are usually directed and have one (unary relation), or two (binary relations) arguments. For instance, the upper example in Figure 1 shows three kinds of relations: *uses* signals which data objects are used by an activity, *performer* captures which actor performed an activity, and *flow* captures a sequential relation between two activities. RE is crucial when it comes to information extraction in our context, given that processes inherently involve process steps (i.e., activities) that are connected to each other through relations. Note that we regard constraint extraction [2, 26] (CE), which relates to declarative process modeling, as an RE problem: constraints have one or two arguments, are directed, and carry type information (e.g., *Succession*, *Init*).

2.2 Challenges of Process Information Extraction from Text

Information extraction, a common task in natural language processing (NLP), faces general challenges, which are also well-known in BPM literature [1,15], and often central elements of interest in the design of rule-based and learning-based systems alike [2,23]. Simply using LLMs for process information extraction solves some of these challenges and justifies the investigation of their applicability.

In the context of process-related texts ***Linguistic Variance*** means that the same behavior or process characteristics can be described in a variety of ways such as, for instance, active and passive voice. ***Context Cues*** are a challenge in that single words can fundamentally alter the meaning of a process description (e.g., *"first, a claim is created" and inverted semantics in "finally, a claim is created"*. Processes are typically described in sequential form, although they usually contain branches (e.g. XOR decision branches). This results in ***Long-distance***

Relations that existing approaches struggle with [23] or cannot handle [2]. ***Implicit and Ambiguous Information*** such as the "examination target" in *"after registering the file in the database, it needs to be examined"* needs to be interpreted [3,15]. Research indicates a negative correlation between correctness of extracted process information and ***Text Length*** [7]. Finally, the application of deep learning is hindered by the fact that the largest available data set contains only 45 human-annotated process descriptions [9] (***Small Datasets***).

LLMs are able to overcome the above challenges [32], which is why this paper analyzes their suitability for process information extraction, as proposed in previous work [6, 8]. However, LLMs require great care in the formulation of the input (prompts) [8, 22, 31–33]. In [22] authors argue: "a good prompt can be worth hundreds of labeled data points". For this reason, the core of the present work lies in the development (Section 4) and evaluation (Sections 5 and 6.2) of suitable prompts for process information extraction.

2.3 Challenges of Process Information Extraction Using LLMs

Using LLMs for process information extraction from texts helps with linguistic challenges, but adds itself several additional challenges. We discuss these here and reference them later in Section 6.4 to show how we can deal with them.

(*C1*) Limited output control. Input and output consist of plain text. Given that the input for inference is raw text, it inherently suits LLMs for our tasks (cf. Section 2.1). However, as the expected output should adhere to a specific schema, it becomes necessary to instruct the LLM to conform to this schema. Moreover, this principle necessitates a robust output parser, as LLMs tend to exhibit variability in their output, which presently cannot be entirely eradicated. Having only limited control over generated output is especially problematic for the BPM domain, where definitions for relevant information often overlap, e.g., *actions* (just predicate) versus *activities* (predicate and object).

(*C2*) Input presentation dependencies. Although LLMs provide an interface for natural language input, the quantity, form and level of detail must be carefully matched to the task at hand. The LLM faces the challenge of determining the importance of the input components. Furthermore, while LLMs emulate human reasoning, the interpretation of inputs may diverge significantly from that of human beings, thereby rendering prompt optimization a trial-and-error process. This challenge is further aggravated by some elements of process models, that are complex to explain concisely, e.g., parallel and exclusive workflows.

(*C3*) Black-box. Deep learning methods generally suffer from challenges concerning explainability of predictions [34], which is also true for LLMs. Contrary to classical learning methods, such as decision trees, LLMs offer no fail safe mechanism to validate extraction rules. This is problematic for business process information extraction in particular, since recent work focuses on "human-in-the-loop" systems [30], where the human must be able to follow system decisions.

(*C4*) Data unawareness. In contrast to generative AI models trained on task-specific data, an LLM is usually not aware of the particular dataset it is tasked to process. Thus, using LLMs to process a particular dataset requires to

form instructions that precisely describe all relevant details of a dataset. The generalizing capabilities of LLMs can be an additional hurdle in this context, especially, when declarative process models are concerned, where a multitude of constraint types exist. The LLM is likely to know of these through the pre-training process, and therefore may extract irrelevant ones for a given dataset.

(*C5*) Costly experiments. Applying LLMs usually requires usage of commercial APIs (e.g., OpenAI), which come with downsides: (i) a token limit restricting the maximum input and output size and (ii) fees based on the number of tokens processed. In view of the many possible variations in the influencing parameters, conducting experiments can be cost-intensive. This is especially true for the BPM domain, where the density of information in process descriptions is very high, needing many output tokens to extract and encode it.

3 Related Work

Related approaches are divided into rule-based, machine learning (ML)-based, deep learning-based, and LLM-based process information extraction. An approach is considered related if it solves at least one of the tasks in Section 2.1.

Rule-based approaches. Rule-based approaches leverage linguistic features to extract information from natural language process descriptions through explicitly coded mapping rules. For instance, Friedrich et al.'s seminal work [16] employs syntax features and word information from a lexical database to identify patterns at both sentence and document levels for BPMN model creation. Other approaches like those in [26, 28] adopt similar techniques for automatic text annotation, employing regular expressions for syntactic dependency trees, and part-of-speech tags, showcasing superior performance on novel datasets. Additionally, [2] presents a rule-based technique, currently leading in extracting declarative process models from raw text using syntax parsing and word-level features. Similar advancements are seen in [14] and [9], the latter integrating ML-based entity MD with subsequent rule-based RE. Furthermore, [21] focuses on extracting Dynamic Condition Response (DCR) graphs. Recent studies suggest that while rule-based approaches can be tailored to specific tasks and data sets, they can hardly deal with ambiguity and linguistic variance. [8, 23].

ML-based approaches. [23] presents a ML extraction pipeline based on [9] and is used as a baseline for our comparative evaluation (Section 6). The deep-learning approach presented in [25] classifies text fragments analyzing the input text on several levels of granularity. However, extracting these fragments is not part of the approach, which simplifies the task of MD to mention classification, i.e., locating the information to extract is omitted. Though the work presented in [6] overcomes this limitation and outperforms the approach, it does not support RE. In general, techniques of this paradigm either struggle to deal with linguistic variability and ambiguity, or they require vast amounts of training data, making them particularly unfeasible for small datasets (see Section 2.2).

LLM-based approaches. Bellan et al. [8] utilize pre-trained LLMs to cope with data scarcity, yet their approach exhibits three primary weaknesses: *(i)* It is

restricted to a subset of entity types, namely *activities*, *participants*, a *performs* relation, and a *direct-consequences* relation, *(ii)* it lacks strict output formatting, hindering automated result processing, unlike our prompting strategy, and *(iii)* its evaluation is limited to 7 of the 45 process descriptions from the PET dataset, whereas we evaluate our modular prompt on the entire dataset, plus two validation datasets. Thus, direct comparison between [8] and our work is not feasible. Nonetheless, as in [8], our modular prompt also descriptive instructions with input examples accompanied by their expected outputs. Although process models are generated using LLMs in [18], the work is not comparable to ours, as [18] requires human involvement and only supports the extraction of activities and their arrangement in a directed graph (e.g., actors and data are missing).

4 LLM-based Process Model Extraction from Text

Extracting process information with LLMs requires a *prompt design* that addresses the challenges mentioned in Section 2.3. Thus, a prompt structure consisting of the three modules *Context*, *Task Description* and *Restrictions* is described below at the example of the Mention Detection task. However, the prompting strategies for the remaining tasks (Section 2.1) are analogous.

High-Level Prompt Structure. LLMs take freely formulated text as input, which is called the prompt. To this end we base our prompts on an ablation study (Section 6.2), which is used to identify beneficial and detrimental prompt components. To do this, we first need a modular prompt design so that we can specifically remove individual components in the study to examine their benefits and ultimately to only keep the advantageous components. Adhering to the best practices outlined in [31], our initial prompt design is structured into three modules (see Figure 2): *(A)* a context description framing the process information extraction task on a high level, *(B)* a detailed task description, and *(C)* constraints that further restrict the context and the output format, and contains disambiguation hints. To design potentially relevant components for all three modules we rely on general design patterns [11, 22, 31, 33]. Therefore, in the next subsection, the three modules are specified and discussed in terms of how they address the LLM-specific challenges outlined in Section 2.3.

Context and Task Description. In module *(A)* we use the *persona* design pattern [33] to control the language style of generation results. We assign it the role of a process modeling expert. This is followed by the *context manager* design pattern [33], which includes a general description of the information extraction task (i.e., objectives and a description of the input specifics). This limits the information basis the LLM may use, mitigating the risk of hallucination.

Module *(B)* is mainly concerned with defining the specifics of the process information extraction task. Its backbone is the *creation of a meta language* [33], which defines the types of elements to extract from the input text. Figure 2

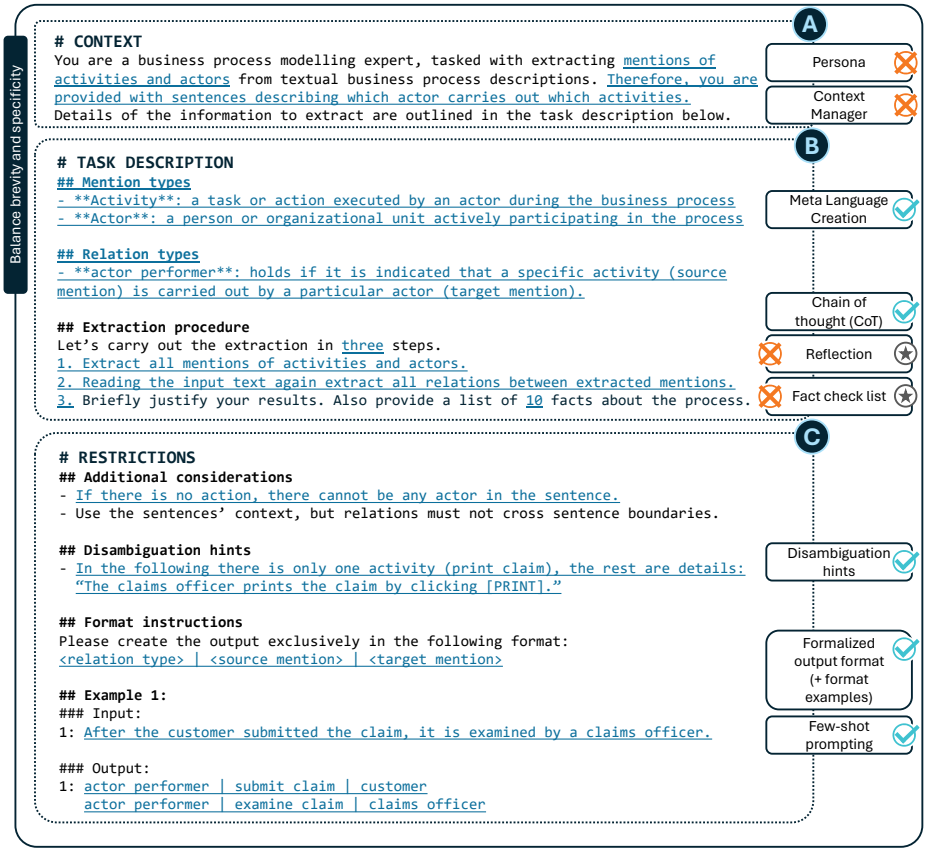

Fig. 2: Modular prompt structure (underlined = task-specific content, boxes = design pattern, ☑ = useful, ☒ = non-useful, ⍟ = use in prompt engineering).

provides an example that defines activities and actors as mention types and a relation called *actor performer* that associates actions with their performers, following our running example from Figure 1. Another widespread best practice is known as *chain of thought (CoT)* [31, 32], whereby the actual task is broken down into individual steps. Thus, our prompt divides the relation extraction task into two steps that separate the extraction of mentions from the prediction of their relations and a third step, which combines two more best practices, i.e., generating a *list of facts* about the process and *reflection* about the results [33]. These cause the LLM to elaborate both on the input and on its own output, which allows experts to validate the extracted information and has also been shown to have a positive impact on extraction performance [31, 32].

Restrictions. The last prompt module *(C)* defines expectations towards the LLM's output. *Additional considerations* include rules for the extraction task,

such as that an actor can only be described as such if the action it performs is also named (compare Figure 2). *Disambiguation hints* are particularly useful for information types that are hard to distinguish from other information types or irrelevant information. In Figure 2 it is intended to guide the LLM what makes up an *Activity*, if the input gives additional, irrelevant specifics. Prompts further include a schematic definition of a *formal output format* [33], which for the exemplary prompt is a tuple of a relation type, a source mention and a target mention, each separated by a pipe symbol. The definition is complemented by an (out-of-domain) example. Finally, *few-shot prompting* [22, 31] dynamically adds examples for input and corresponding output. For the current paper few-shot samples are pairs of raw textual process descriptions and a task-dependent set of process-relevant information (e.g., actor mentions). This design pattern and best practice is known to alleviate the issue of *data unawareness* of LLMs [17, 20].

5 Experiment Setup

In this section we define the experiments we run to evaluate the usefulness of LLMs for process information extraction [6]. It covers an overview of the datasets we use, including the respective baselines, and a definition of metrics we apply.

We use three well-known datasets for evaluating our prompts. One of these (PET) represents the current state of the art, both in terms of size, as well as the process information techniques developed for it. The other datasets feature different characteristics, making them relevant for validation experiments. This lets us gain insights into the robustness of an LLM as a process information extractor, as well as how it behaves when applied to other process modeling languages. We call the best approaches for extracting the information from these datasets *baselines*, and use them in Section 6 as comparisons with various LLMs.

PET [9]: This is the largest dataset currently available. It contains 45 documents with annotations for information especially useful for creating process models in BPMN. These include 7 types of mentions such as activities, actors, data objects, but also 6 relation types. These cover the behavioural process perspective (*Flow*), data perspective (*uses*, and organizational perspective (*actor, performer*). Additionally, this dataset features relations that span multiple sentences. It therefore tests the ability of approaches to reason across wider spans of text. We use an extended version of this dataset, which includes data for the ER task, as presented in [23]. The currently best approach for extracting information is using conditional random fields for MD, a pre-trained neural coreference resolver for ER, and a decision tree ensemble for RE [23]. We use scores as reported in [23], which have corresponding publicly available code and can be reproduced with it.

DECON [2]: This collection of 17 textual process descriptions is annotated with a set of 5 Declare [24] Constraint types between business process relevant activities. Additionally, constraints may be negated, as well as unary, i.e., constraining

[6] Code at https://github.com/JulianNeuberger/llm-process-generation/tree/er2024 .

a single action. Annotations are given on a sentence level, and only sentences that describe at least one constraint are contained in the dataset. The expected (ground-truth) activities are already transformed into Declare-conform phrases, i.e., the activity description "*The claim is registered*" should be extracted as "*register claim*". It does not contain an approach for MD in isolation, and only contains mentions of type action. The authors of [2] propose a rule-based approach combining multiple NLP techniques, e.g., *typed dependency relations*.
ATDP [26]: This dataset uses 18 textual descriptions, that largely overlap with the ones from [2], but also contains sentences, that describe no constraint. As such, this dataset tests approaches for their ability to judge whether or not sentences contain relevant information, before extracting constraints. Furthermore, the set of constraints was expanded to eight types. Additionally, this dataset also provides annotations of actions, conditions, entities, and events, which we used in an MD setting. Quishpi et al. proposed a rule-based ensemble of patterns for MD and CE on typed dependency structures [26].

We use the well established metrics *Precision* P and *Recall* R for our experiments. P is a measure of how well an extraction approach is able to avoid false positives, i.e., assigning the wrong type to mentions and relations, or extracting them, where they are not expected. R on the other hand measures how much of the expected information (true positives) is found. The two metrics are typically aggregated via their harmonic mean $F_1 = 2\frac{P \cdot R}{P+R}$. Following [26], we use $P = \frac{\#correct}{\#pred}$ and $R = \frac{\#correct}{\#gold}$, with $\#correct$ as the number of correct predictions, $\#pred$ the number of total predictions, and $\#gold$ as the number of expected mentions or relations. For a fair assessment, we count predictions as correct in exactly the way described by the work we compare the LLM to.

6 Results

Table 1 shows the results we observed when running the experiments as described in Section 5 with an optimized prompt, that follows the recommendations we found in our study of best practices (Section 6.2). All results use GPT-4o, the latest version of OpenAI's GPT with $temperature = 0$. top_p is unchanged, as per OpenAI's recommendation, when using temperature based sampling. [7]

For the reference dataset PET, our experiments show that GPT-4o is capable of an absolute F_1 score improvement of 5% for MD, 22% for ER, and 17% for RE. Remarkably, for RE, GPT-4o is able to match and outperform the machine learnt baseline, which was trained on 36 manually annotated documents [23], without any labeled data (zero-shot). For MD it reaches similar scores, even when not given any examples, compared to the machine learnt baseline, which was trained using 36 manually annotated documents [9]. For real-world application this means that LLMs can be used in business process information extraction scenarios, even if the organization has not a single manually annotated training example. This is an exciting find, as it promises significant speed-up of model creating tasks of practitioners across business domains.

[7] see OpenAI's source code, accessed June 3, 2024

	Dataset	DECON			ATDP			PET		
	Metric	P	R	F_1	P	R	F_1	P	R	F_1
Mention Detection	Baseline	*no baseline*			0.62	**0.82**	0.71	**0.73**	0.64	0.69
	Zero-shot	0.72	0.75	0.73	0.58	0.77	0.66	0.65	0.71	0.68
	1-shot	0.87	**0.80**	0.83	0.63	0.77	0.69	0.72	0.75	0.73
	3-shot	**0.88**	0.79	**0.83**	**0.68**	0.79	**0.73**	0.72	**0.77**	**0.74**
Entity Resolution	Baseline	*no data*			*no data*			0.55	**0.51**	0.52
	Zero-shot							0.67	0.55	0.60
	1-shot							0.76	**0.70**	0.73
	3-shot							**0.79**	**0.70**	**0.74**
Relation Extraction	Baseline	0.77	0.72	0.74	**0.58**	0.64	0.61	0.79	0.66	0.72
	Zero-shot	0.66	0.75	0.70	0.49	0.66	0.57	0.88	0.85	0.86
	1-shot	0.76	0.82	0.79	**0.58**	**0.73**	**0.64**	0.90	0.89	0.89
	3-shot	**0.79**	**0.85**	**0.82**	**0.58**	0.72	**0.64**	**0.90**	**0.89**	**0.89**

Table 1: Results for each dataset and the different extraction stages, compared to baseline results using GPT-4o.

When evaluating on the validation datasets (cf. Section 5), we found that GPT-4o is able to match and out-perform the rule-based systems in all cases, most notably improving F_1 scores for RE on dataset DECON by an absolute 8%. Our result for MD on dataset DECON has no corresponding baseline, as the authors of [2] did not report values for MD in isolation. Errors and ambiguities are common in dataset ATDP hindering machine learning methods in learning valid extraction rules. This also adversely affects the extraction accuracy of GPT-4, when extracting the same types of constraints in the ATDP dataset compared to the DECON dataset. We discuss this further in Section 6.4. Since the importance of ER only recently gained attention [23,26], the reference dataset PET is currently the only dataset providing data for evaluation of this task.

6.1 Model Comparison

We originally developed our prompts for GPT-4 version *GPT-4-0125-preview*, to assess how well our prompting strategy generalizes to other models we prompted a total of eight models for the MD and RE tasks on PET. We selected models following AlpacaEval [8], which is designed for testing the instruction following capabilities of LMMs [13]. At the time of writing model *YI Large Preview* was not publicly accessible and could not be considered in our comparison, even though it ranked third on AlpacaEval.

Results for the comparison can be found in Table 2. We set the temperature for all models to 0. All GPT models perform on similar levels, with the exception of GPT3.5, which is significantly smaller compared to GPT-4 models. For the zero-shot RE task GPT3.5 even failed to produce responses for most documents, leading to very low recall. Claude3 Opus seems to be as capable as GPT-4, its

[8] See https://tatsu-lab.github.io/alpaca_eval/, last accessed May 30, 2024.

Task	**PET** MD (Zero-shot)			**PET** MD (3-shot)			**PET** RE (Zero-shot)			**PET** RE (3-shot)		
Model	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
GPT-4o	0.58	0.69	0.63	0.68	0.77	0.72	**0.88**	**0.85**	**0.86**	0.90	0.89	0.89
GPT-4-2024-04-09	0.63	0.67	0.65	**0.73**	0.76	**0.74**	0.87	0.79	0.83	0.89	0.88	0.88
GPT-4-0125-preview	**0.65**	0.71	**0.68**	0.72	0.77	**0.74**	0.87	**0.85**	**0.86**	0.89	0.87	0.88
GPT-3.5-0125	0.35	0.50	0.42	0.51	0.70	0.59	0.51	0.06	0.11	0.74	0.64	0.69
Claude 3 Opus	0.55	**0.72**	0.63	0.66	**0.80**	0.73	0.86	**0.85**	**0.86**	**0.92**	**0.91**	**0.91**
Claude 3 Sonnet	0.46	0.65	0.54	0.63	0.78	0.70	0.78	0.67	0.72	0.91	0.87	0.89
Llama 3 70B Instruct	0.56	0.64	0.59	0.62	0.71	0.67	0.76	0.66	0.70	0.88	0.81	0.84
Qwen1.5 72B Chat	0.32	0.33	0.33	0.53	0.65	0.59	0.61	0.65	0.63	0.74	0.77	0.75

Table 2: Comparison of our prompts across different models. Best results per task and metric are set **bold**.

smaller variant Sonnet performs significantly worse on zero-shot tasks, but is able to produce comparable results given three examples. Llama 3 70B Instruct is an open-weight model and could be run locally, i.e., it is useful for using our prompting strategy in scenarios where sending data to an API is not possible. Llama 3 70B Instruct seems to be nearly as capable as the closed-weights Claude 3 Sonnet and is therefore viable in a few-shot setting.

6.2 Ablation Study

We conduct an ablation study to assess the usefulness of the best practices presented in Section 4 and to measure the impact of the prompt's main components. This study is run on the reference dataset (PET), as it is the largest one and used by recent publications [8, 9, 23]. To obtain a baseline for the tasks of MD and RE, we use a prompt that implements the best practices as shown in Figure 2 and run it on the *GPT-4-0125-preview* model. We then purposefully remove specific components from this prompt, namely the *format examples*, the *context manager*, the *persona*, the definition of mention and relation types (*meta language*), the instruction to think in several steps (*chain of thought*), any *disambiguation* hints, and the instruction to generate explanations (*reflection*) and a *fact check list* about the process. Additionally, we also use a prompt with very short descriptions of relations and types (*balancing brevity and specificity*). We run each prompt in the zero-shot setting and record the observed F_1 score, as well as the parsing errors that occurred.

Table 3 provides detailed results. Removing examples has a significant negative effect (-0.22 for MD and -0.07 for RE), mainly rooted in the number of parsing errors that are made (919 for MD), as well as directionality of relations for the RE task (confusing source and target mentions). Removing the context manager and persona only has minor effects (± 0.01 per task), suggesting lower relevance for process information extraction compared to other NLP settings.

In addition to removing prompt components, we also tested using GPT in an older, less capable, but much cheaper version, GPT-3.5. Running the baseline

	Mention Detection (MD)			Relation Extraction (RE)			
Experiment	*Relative* F_1	*Absolute* F_1	*Parsing Errors*	*Relative* F_1	*Absolute* F_1	*Parsing Errors*	*Useful*
Baseline	–	0.59	0	–	0.77	0	
No Format Examples	−0.22	0.37	919	−0.07	0.70	1	✓
No Context Manager	+0.01	0.60	0	−0.01	0.76	0	
No Persona	+0.01	0.59	1	+0.01	0.78	0	
No Meta Language	−0.09	0.49	2	−0.05	0.72	0	✓
No Chain of Thought	−0.01	0.57	1	−0.02	0.75	0	✓
No Disambiguation	−0.03	0.55	0	−0.01	0.76	1	✓
No Reflection	+0.04	0.63	0	+0.02	0.79	0	★
No Fact Check List	+0.03	0.62	1	−0.02	0.75	0	★
Very Short Prompt	−0.04	0.54	1	−0.03	0.74	0	✓

Table 3: Changes in F_1 score of GPT-4, without specific prompt components given in Figure 2. Column *relative* F_1 shows difference to the baseline prompt, *Useful* shows a ✓, if we recommend this component in prompts for process information extraction and ★ for prompt engineering and data curating only.

prompt, results in a significant drop in extraction quality, with $F_1 = 0.27$ for MD and $F_1 = 0.56$ for RE. Splitting the baseline prompt into multiple prompts, each focusing on only one mention type, lets us prompt GPT-4 repeatedly for the same document. These highly specialized prompts are called "agents", which pass information between each other. For example, we instruct the first agent to extract *Actions*, which are passed to other agents extracting *Actors* and *Business Objects* respectively. This lets us exploit the inherent dependency between these elements. If no Action is detected in a sentence, then there is likely no relevant Actor or Business Object, even if there are nouns that would qualify from a linguistic standpoint. This way of prompting leads to an absolute improvement of +0.08 in F_1 score for both the MD and RE tasks.

6.3 Stability of Results

LLMs are notorious for their non-deterministic output [10], which often puts the validity and stability of results into question. To assess the severity of these problems with our prompting strategy, we repeated the extraction of mentions (MD) and relations (RE) on all documents of PET five times. In each iteration we used gpt-4o-2024-05-13 as a model in a 1-shot setting and recorded the micro F_1 score. We then calculated mean (0.70 for MD, 0.89 for RE), standard deviation (0.003 for MD, 0.002 for RE), minimum (0.69 for MD, 0.89 for RE), and maximum (0.70 for MD, 0.89 for RE). While there are fluctuations in results, they are so minor that they do not call the validity of our results into question.

6.4 Lessons Learned

Using LLMs for extracting process relevant information brings with it a category of challenges, which we already discussed in Section 2.3. Solving these is paramount for successful application of LLMs. In this section we discuss how we approached these challenges and what lessons we learned.

(*C1*) Limited output control. The expected output format, as well as the form of extracted information, can mainly be influenced by the prompt components *Meta Language* and *Format Examples*. Adding these results in significantly improved F_1 scores, (+0.22 and +0.05 respectively for MD on PET). These improvements are explained by less parsing errors (919 less for MD on PET), and better recall and precision in detecting mentions. LLMs also run the risk of being "stochastic parrots", simply synthesizing linguistically correct phrases, based on their training data [10]. In our experiments we observed changes in F_1 of maximally -0.02 for rephrased prompts. This indicates robustness of our prompts and suitability of LLMs as an tool for business process information extraction.

(*C2*) Black-box. A valuable advantage of utilizing LLMs is their ability to reflect, thereby providing justification for their generated results. Figure 3 shows three examples of justifications for extraction results in the ATDP dataset (see Section 2.1). Case I shows the ideal outcome where prediction and expected constraint are identical. Note, that the justification even refers to the meta-language provided in the prompt (compare Section 4). In case II, the prediction and the gold standard constraint do not match, because of an error in the gold standard data, following [2], which defines completing a process as a *meta action* that can not be part of any constraint. The dataset creators are alerted of this issue by the LLM, since it plausibly justifies why *send out report* marks the end of a process instance. Finally, in case III the extraction result is controversial, since the sentence is ambiguous. If we consider the term *immediately* to encompass both actions, they are constrained by an *existence* constraint. Alternatively, viewing *check quantity* as a subtask of *process part list* suggests only one action needs modeling. Using such reflective explanations make LLMs useful for "human-in-the-loop" systems, which are already applied in fields like process mining [30].

(*C3*) Input presentation dependencies. Adding more text to prompts sometimes has an adverse effect, reducing extraction quality (Section 2.3). This makes optimizing prompts difficult, since it is not clear, if adding additional disambiguation hints or longer definitions would improve the result. Using partial extraction prompts helps with this issue, as the sections regarding *Meta Language Creation* can be focused on a few types. Depending on the task, there may even be interdependency between information, that can be efficiently exploited in this way.

(*C4*) Data unawareness. This issue arises, when LLMs are used in a zero-shot setting. There, the components *Meta Language*, and *Format Examples* are the only ways to "teach" the model how to perform the task. Applying the pattern of *few-shot prompting*, i.e., using labeled data in a few-shot setting was beneficial. This makes the use of an LLM more akin to training a machine learning model, but with significantly lower data requirements. In our experiments, three exam-

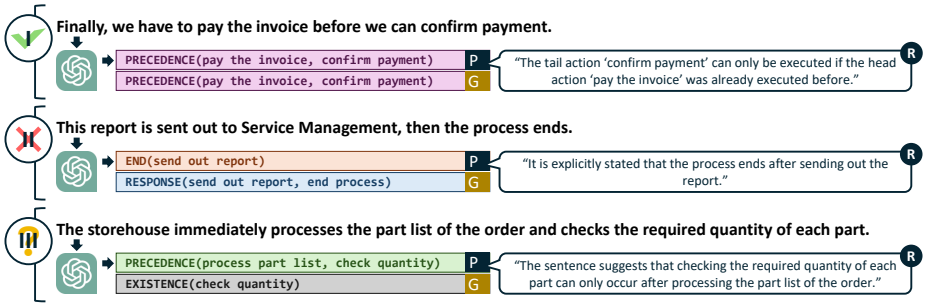

Fig. 3: Reflection example with Predicted and Gold standard constraints: (I) perfect match, (II) gold standard error, (III) ambiguous case

ples were sufficient to achieve better extraction results than those of machine learning models trained with more than ten times of the data.

(*C5*) Costly experiments. This is a major drawback of LLM based process information extraction. The most capable LLMs are hosted as cloud-based solutions and are priced per token. We found that limiting the number of examples to 1 resulted in the best cost-value ratio. Additionally, our experiments showed that leaving out the prompt components *Context Manager*, *Persona*, and *Disambiguation* is a valid way to limit the number of tokens sent per request, albeit with potential minor decreases in extraction accuracy. Prompting LLMs without the request for a *Fact List*, nor *Explanations* for extracted information greatly reduces the amount of tokens as well, especially useful after prompt engineering or data curating (during "*inference*"). Alternatively one can switch to cheaper models, i.e., *LLama 3 72B*, if the drop in performance is acceptable.

7 Conclusion

Summary. This paper presents an extensive study on the usefulness of LLMs for the extraction of process information from natural language text. We collected linguistic challenges and discuss how LLMs are uniquely fit for solving them. We also discussed challenges that arise through the use of LLMs and show how other communities propose to deal with these (or similar) concerns through prompt engineering. We present experimental results on three process information extraction datasets, which at least match the current state of the art on these datasets and in most cases improve it by as much as 8% in the F_1 metric. This shows the suitability of LLMs as a method for extracting business process relevant information from natural language process descriptions. To flesh out this notion, we analyze how well our prompting strategy can be applied to different LLMs without changing them, showing their universal nature. We expect LLMs to be a benchmark in the process information extraction domain for the foreseeable future, as limitations in dataset quality and quantity, combined with the need for complex reasoning make it very hard to train large extraction

approaches from scratch. We make all our code, prompts, and LLM answers available, to support further research.

Limitations. A limitation of our work is that the list of prompt components we present may not be exhaustive, and they may have interactions that our ablation study does not capture. Additionally, some models suffer from hallucinations, especially Qwen1.5 and Llama 3, which hallucinate non-existent entity and relation types – 20 and 37 instances in the worst cases respectively. However, the severity of this problem diminishes in the few-shot setting (0 and 4 instances respectively in the worst case). We plan on analyzing how our prompts could be enhanced to improve instruction following for these models. Lastly, the current pricing models prohibit large-scale application of the most capable model to process information extraction. Alternatives (e.g., Llama) avoid this issue, but require more labeled examples to reach comparable levels of performance. This limitation may change in the near future, as more cost-efficient models and specialized hardware reduce costs to acceptable levels. Alternatively, very capable — and therefore expensive — LLMs could be used to create and curate training data for smaller local models, leveraging the reasoning capabilities of LLMs indirectly.

Future Work. In future work we aim to use LLMs as tools to support labeling of new data. Current datasets are limited in origin, i.e., they usually describe processes from municipalities or small service providers. We plan to analyze the ability of LLMs to generalize beyond the domains with available labeled data and highlight the promising flexibility observed in our current experiments. Additionally, using GPT-4o's image processing and generation capabilities could be a promising line of research for direct text to model transformation. Finally, our results show very limited improvement in extraction quality, when the prompt includes a role the LLM is restricted to (persona). A slight variation of this idea is to describe the target audience of extraction results in the prompt, to further improve the quality of extracted process information.

References

1. Van der Aa, H., Carmona Vargas, J., Leopold, H., Mendling, J., Padró, L.: Challenges and opportunities of applying natural language processing in business process management. In: COLING (2018)
2. van der Aa, H., Di Ciccio, C., Leopold, H., Reijers, H.A.: Extracting declarative process models from natural language. In: CAiSE (2019)
3. Van der Aa, H., Leopold, H., Reijers, H.A.: Checking process compliance against natural language specifications using behavioral spaces. IS (2018)
4. van der Aa, H., Leopold, H., van de Weerd, I., Reijers, H.A.: Causes and consequences of fragmented process information: Insights from a case study. In: AMCIS (2017)
5. van der Aalst, W.M.P.: Process Mining - Data Science in Action, Second Edition. Springer (2016). https://doi.org/10.1007/978-3-662-49851-4, https://doi.org/10.1007/978-3-662-49851-4
6. Ackermann, L., Neuberger, J., Jablonski, S.: Data-driven annotation of textual process descriptions based on formal meaning representations. In: CAiSE (2021)

7. Ackermann, L., Neuberger, J., Käppel, M., Jablonski, S.: Bridging research fields: An empirical study on joint, neural relation extraction techniques. In: CAiSE (2023)
8. Bellan, P., Dragoni, M., Ghidini, C.: Extracting business process entities and relations from text using pre-trained language models and in-context learning. In: EDOC (2022)
9. Bellan, P., Ghidini, C., Dragoni, M., Ponzetto, S.P., van der Aa, H.: Process extraction from natural language text: the PET dataset and annotation guidelines. In: NL4AI (2022)
10. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: Can language models be too big? In: ACM FAccT (2021)
11. Cui, L., Wu, Y., Liu, J., Yang, S., Zhang, Y.: Template-based named entity recognition using bart. arXiv preprint arXiv:2106.01760 (2021)
12. Davies, I., Green, P., Rosemann, M., Indulska, M., Gallo, S.: How do practitioners use conceptual modeling in practice? Data & Knowledge Engineering **58**(3), 358–380 (2006)
13. Dubois, Y., Li, C.X., Taori, R., Zhang, T., Gulrajani, I., Ba, J., Guestrin, C., Liang, P.S., Hashimoto, T.B.: Alpacafarm: A simulation framework for methods that learn from human feedback. Advances in Neural Information Processing Systems **36** (2024)
14. Ferreira., R.C.B., Thom., L.H., Fantinato., M.: A semi-automatic approach to identify business process elements in natural language texts. In: ICEIS (2017)
15. Franceschetti, M., Seiger, R., López, H.A., Burattin, A., García-Bañuelos, L., Weber, B.: A characterisation of ambiguity in bpm. In: International Conference on Conceptual Modeling. pp. 277–295. Springer (2023)
16. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: CAiSE (2011)
17. Kaddour, J., Harris, J., Mozes, M., Bradley, H., Raileanu, R., McHardy, R.: Challenges and applications of large language models. arXiv preprint (2023)
18. Kourani, H., Berti, A., Schuster, D., van der Aalst, W.M.: Process modeling with large language models. arXiv preprint arXiv:2403.07541 (2024)
19. Leopold, H., van der Aa, H., Pittke, F., Raffel, M., Mendling, J., Reijers, H.A.: Searching textual and model-based process descriptions based on a unified data format. SoSym **18**, 1179–1194 (2019)
20. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., et al.: Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems **33**, 9459–9474 (2020)
21. López-Acosta, H.A., Hildebrandt, T., Debois, S., Marquard, M.: The process highlighter: From texts to declarative processes and back. In: CEUR Workshop Proceedings. pp. 66–70. CEUR Workshop Proceedings (2018)
22. Min, B., Ross, H., Sulem, E., Veyseh, A.P.B., Nguyen, T.H., Sainz, O., Agirre, E., Heintz, I., Roth, D.: Recent advances in natural language processing via large pre-trained language models: A survey. ACM Computing Surveys **56**(2), 1–40 (2023)
23. Neuberger, J., Ackermann, L., Jablonski, S.: Beyond rule-based named entity recognition and relation extraction for process model generation from natural language text. In: CoopIS (2023)
24. Pesic, M., Schonenberg, H., Van der Aalst, W.M.: Declare: Full support for loosely-structured processes. In: 11th IEEE international enterprise distributed object computing conference (EDOC 2007). pp. 287–287. IEEE (2007)

25. Qian, C., Wen, L., Kumar, A., Lin, L., Lin, L., Zong, Z., Li, S., Wang, J.: An approach for process model extraction by multi-grained text classification. In: CAiSE (2020)
26. Quishpi, L., Carmona, J., Padró, L.: Extracting annotations from textual descriptions of processes. In: BPM 2020 (2020)
27. Richens, R.H.: Interlingual machine translation. The Computer Journal **1**(3), 144–147 (1958)
28. Sànchez-Ferreres, J., Burattin, A., Carmona, J., Montali, M., Padró, L., Quishpi, L.: Unleashing textual descriptions of business processes. SoSyM (2021)
29. Sukthanker, R., Poria, S., Cambria, E., Thirunavukarasu, R.: Anaphora and coreference resolution: A review. Information Fusion (2020)
30. Ter Hofstede, A.H., Koschmider, A., Marrella, A., Andrews, R., Fischer, D.A., Sadeghianasl, S., Wynn, M.T., Comuzzi, M., De Weerdt, J., Goel, K., et al.: Process-data quality: The true frontier of process mining. ACM JDIQ (2023)
31. Törnberg, P.: Best practices for text annotation with large language models. arXiv preprint arXiv:2402.05129 (2024)
32. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al.: Chain-of-thought prompting elicits reasoning in large language models. NIPS (2022)
33. White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., Schmidt, D.C.: A prompt pattern catalog to enhance prompt engineering with chatgpt. arXiv preprint arXiv:2302.11382 (2023)
34. Xu, F., Uszkoreit, H., Du, Y., Fan, W., Zhao, D., Zhu, J.: Explainable ai: A brief survey on history, research areas, approaches and challenges. In: NLPCC (2019)