ProcessChat: A Business Process Grounded Dialogue Dataset

Anonymous ACL submission

Abstract

Business processes are designed to streamline and optimize work within an organization and are often defined and documented by domain experts or process analysts using formal specifications. However, these specifications may be complex for the users executing the tasks of the process. For example, a recruitment process designed by a domain expert is used by many actors in the organization, who may not be skilled in understanding the formal notations that specify the process. With recent advancements in large language models, there has been increasing interest in enabling users to ask questions in natural language and receive relevant responses that are specific to the user's context and process knowledge. We propose a dialog dataset grounded in domain-specific process knowledge, which it is supposed to follow during the conversation. The dataset consists of 316 dialogs grounded on 73 different process model specifications. We also present a baseline model, which is trained on the proposed dataset. Our experiments find that the model can do zero-shot transfer to unseen processes, and sets a strong baseline for future research.

1 Introduction

A commonly used standard for process modeling is the Business Process Model and Notation (BPMN), overseen by the Object Management Group (OMG) which provides a rich set of notations (OMG, 2011) to represent the process operations. While process models (or specifications) are useful artifacts to represent the operations, one of the challenges in using them is that they are not intuitive to business professionals, who conduct various tasks of the process (van der Aa et al., 2015). For example, a Human Resource (HR) process for a leave application can be defined by a domain expert using BPMN specification. Figure 1 presents a simple leave application process specified using BPMN. The specification comprises: i) the activities (rectangles) performed by different resources or actors (lanes), ii) the sequence flow (directed edges), iii) the gateways that depict parallel, exclusive, and complex forking or joining of paths, and iv) events indicating triggers such as start or end of the process.

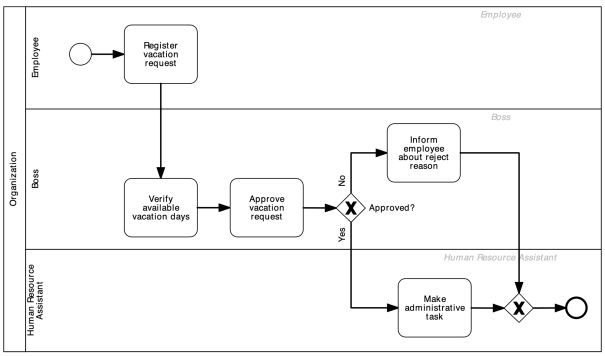

Figure 1: BPMN process specification

A common challenge of using such a specification is the ability to interpret complex specifications, as the process of applying for a vacation will be used by all employees in the organization, who may not have the expertise to interpret the specification. Hence, organizations often provide text-based descriptions that describe the steps of a business process in natural language. Prior work has focused on transforming process models into an intuitive textual description (Leopold et al., 2014), suitable for business process professionals. Yet, there are challenges in navigating and interpreting appropriate textual descriptions through a large repository of process artifacts (van der Aa et al., 2015). One widely plausible approach that organizations have explored to address the challenge of navigating through a large number of documents is to provide chatbots that provide responses based on information present in a knowledge base.

Recent advances have chatbots supported by LLMs thus enabling users ask questions in natural language and receive relevant responses grounded

on domain-specific data [1,2]. While engaging in multi-turn conversations with humans is a fundamental capability of LLMs, there has been no quantitative evaluation of dialogues generated for a multi-turn conversation based on a process model. To the best of our knowledge, there is no proper data set that can be used for this purpose. In this work, we make the following contributions:

- Provide a dialog data set grounded on process descriptions and model specification (ProcessChat Dataset)
- Provide a baseline solution for the problem of end-to-end training of a process model grounded chatbot and evaluate it in a zero-shot setting.

2 Related Work

Quantitative evaluation of chatbots for process management requires datasets depicting process knowledge. There are datasets that are annotated to enable the extraction of business artifacts from natural language descriptions. The most recent is the PET dataset (Bellan et al., 2022), consisting of a corpus of business process descriptions annotated with activities, gateways, actors, and flow information of 45 processes. Another dataset of 17 process descriptions annotated with declarative constraints depicting the relation between various activities has been evaluated and provided by (van der Aa et al., 2019). A large dataset of 73 process model descriptions and their BPMN specifications has been released (Sànchez-Ferreres et al., 2018). The authors provide ground truth references to missing alignments between the process description and the process model. The alignment constraints refer to: i) cardinality: each activity is aligned to exactly one sentence, and each gateway is aligned to exactly one sentence or none at all, ii) ordering: sentences referring to two activities in the textual description follow the same order as the process model. A similar ordering constraint is also defined for parallel and exclusive gateways. We use the dataset of these 73 process descriptions to build the dialog dataset for evaluating chatbots for business processes.

[1]https://www.salesforce.com/news/press-releases/2023/09/12/salesforce-platform-news-dreamforce/

[2]https://blogs.microsoft.com/blog/2023/03/16/introducing-microsoft-365-copilot-your-copilot-for-work/

3 The ProcessChat Dataset

In this section, we describe the proposed dataset, ProcessChat, which includes natural language conversations over BPMN processes. To construct this dataset, we rely on 73 BPMN processes and natural language process descriptions provided by (Sànchez-Ferreres et al., 2018) that have been obtained from 11 different industrial and academic sources[3]. While the original purpose of this repository was to publish a tool to compute an alignment between BPMN process models and natural language descriptions, we simply use the process representations in terms of both BPMN and natural language to build our chat dataset. We start with constructing multiple dialog flows for each process using BPMN. While we focus on BPMN, the approach presented in this paper is independent of the specific notation used to define a process model.

3.1 Dialog grounded on process knowledge

A chatbot, grounded on process knowledge, considers three different kinds of nodes in BPMN (example in Figure 2): i) Events, ii) Activities, and iii) Gateways. Events represent something happening in the process. Common events in a process are the start and end events depicting the start and end of the process. Additionally, messages and timers are depicted as intermediate events in a process. Activities represent steps in the process that are performed. Gateways are used to control the process flows. The two most commonly depicted gateways are: i) an exclusive gateway (Decision) which is used to create alternative paths within a process flow, and an exclusive join gateway which is used to merge alternative paths, ii) a parallel gateway having parallel outgoing process flows, and a parallel join gateway that waits for all incoming flows before triggering the next step in the sequence. Further, the model has pools which are a graphical representation of a participant or an actor responsible for the execution of the activities.

The characteristics of the process-grounded dataset are to provide the following kinds of interactions to the user as detailed by (López et al., 2019):

- Guide the user step-by-step through the process by providing details of the next step to execute having the context of the previous steps the user has been guided through.

[3]https://github.com/setzer22/alignment_model_text

- Ask the user to provide information when the process reaches an exclusive gateway to choose the relevant path and guide the user to the appropriate path
- Inform the user about the actor involved in performing the task
- Provide the user with the list of activities performed by an actor
- Inform the user on reaching the end of the process.

An example of a partial conversation flow based on a Court Hearing process as shown in the BPMN diagram in Figure 2 is given below.

T1 **User**: What is the first step of the process for Court Hearing?
T1 **Assistant**: The Organization will check the files
T2 **User**: What is the next step?
T2 **Assistant**: The Organization wonders if files are missing
T3 **User**: What is the next step?
T3 **Assistant**: What is the outcome of Files missing?
T4 **User**: Yes
T4 **Assistant**: The Organization will initiate a search
T5 **User**: What are the responsibilities of Organization?
T5 **Assistant**: Check files, Files missing?, Initiate search, Track physical files, Hand files to associate, Distribute judge's lawlist, Conduct directions hearings
T7 **User**: What is the next step?
T7 **Assistant**: The Organization will track physical files
T8 **User**: What is the next step after Track physical files
T8 **Assistant**: The Organization will hand files to the associate and The Organization will distribute the judge's lawlist
T9 **User**: What is the next step after Hand files to associate
T9 **Assistant**: The Organization will distribute the judge's lawlist
T10 **User**: What is the next step?
T10 **Assistant**: The Organization will conduct hearings to give directions
T11 **User**: What is the next step?
T11 **Assistant**: You have reached the end of the process

The dialog starts by assuming that a user will initiate the conversation by asking about the first step in a given process. Then the assistant will respond by describing the first step. The user is expected to step through the process by repeatedly asking "What is the next step?" or could ask a specific question asking "What is the next step after A?" (A being an activity in the process), and occasionally asking role-based questions like "Who is responsible for this step?" and "What are the responsibilities of so and so?". When the process is finished, the assistant will inform the user that the end of the process is reached. Each user query and an assistant response is a 'turn'.

In the case of an exclusive gateway, multiple next steps are possible following a given step but are dependent on some conditions. The assistant will then pose a query to the user, asking for the response to the question specified in the exclusive gateway (at T3 in the conversation). Depending on the user's response, the assistant will specify the next step.

In the case of a parallel gateway, multiple next steps are possible following a given step, and the assistant will respond by listing all of them (at T8 in the conversation). It will then step through all the parallel paths based on the user choosing a path and enquiring the next step (at T9 enquiring about the next step after *hand files to associate*). There is another dialog flow created where the user can enquire on the other parallel path ("What is the next step after *distribute judge's lawsuit*?")

Timer or other intermediate events are presented to the user as the next steps. For example, if there is a timer event "design complete", the response of the chatbot to "What is the next step?" will be "The next step is to get the design completed." Hence, the process chat considers events, activities, gateways, and participants (or resources) to construct the dialog.

3.2 Dialog Data construction

We systematically iterate over the paths in the BPMN process model and for each path, construct a dialog flow. Each dialog flow consists of three parts: i) process description in natural language (NL), ii) process description in constrained natural language (CNL), and iii) process path traversal for the multi-turn dialog flow.

Process Description in NL: We envision a scenario where an LLM-driven chatbot (or *assistant*) will be able to answer questions on the process given a textual representation of the process. The textual descriptions in the dataset have been provided by experts covering different styles a variety of styles, The creation of descriptions by experts included three steps (i) Study the process model diagram. (ii) Write the textual description. (iii) Compare the textual description with the process model to and make sure the text accurately describes the process model. This final step aimed to reduce the amount of inconsistencies between texts and models.

For example, the provided NL description for the example process is as follows: *"Each morning, the files which have yet to be processed need to be checked, to make sure they are in order for the*

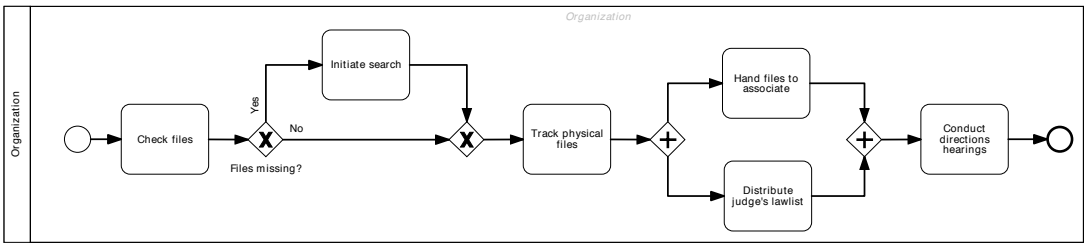

Figure 2: Court Hearing process

court hearing that day. If some files are missing, a search is initiated, otherwise the files can be physically tracked to the intended location. Once all the files are ready, these are handed to the Associate, and meantime the Judges Lawlist is distributed to the relevant people. Afterward, the directions hearings are conducted."

Process Description in CNL: BPMN models can provide precise information as compared to textual descriptions. We would also want to know if an LLM could reason and answer questions with a BPMN model as an input. However, as discussed in prior work (Grohs et al., 2023), as BPMN specification uses the XML format, it can be verbose. The example BPMN of Figure 2, having 6 activities, 2 events and 4 gateways results in 2040 tokens of an open-source LLM such as Mistral (Jiang et al., 2023). The NL description for the equivalent specification results in 94 tokens. Hence, similar to the approach proposed by (Grohs et al., 2023), we synthesize a CNL by parsing the BPMN to represent the process flow. Using a constrained language to represent the flow reduces ambiguity. CNL is NL with restricted grammar and hence simplified and standardized sentence structure. We hypothesize that CNL would be precise as it represents the BPMN model and yet results in improved semantic quality for an LLM to reason. There are constraints the CNL represents: i) an ordering constraint such as *step 1* must happen before *step 2* where *step 1* and *step 2* are consecutive steps in a path, and ii) conditional constraints for an `XOR` gateway such as if the response to *query1* is Yes, then execute *step 3*, where *step 3* is a process step which immediately follows an exclusive gateway in the BPMN represented by *query 1*.

The auto-generated CNL rules for the BPMN model (Figure 2) are:

- if the response to Files missing? is Yes then Initiate search
- Distribute judge's lawlist must happen before Conduct directions hearings
- Initiate search must happen before Track physical files
- Track physical files must happen before Hand files to associate
- Hand files to associate must happen before Distribute judge's lawlist
- Check files must happen before Initiate search

In case of a loopback from a process step B to a previous process step A, while A must always happen before B in any execution path through the process, B only happens before A at the end of the loop. So in such cases, we add the CNL rule "B can happen before A". For the same court hearing process, use of CNL reduces the number of tokens to 142.

Process Path Traversal for Dialog flow: The dialog flow d between a user u and the chatbot (or assistant) a, is represented as a sequence of utterances $d = \{c_1^u, c_1^a, c_2^u, c_2^a \ldots c_m^u, c_m^a\}$, where m denotes the number of exchanges or turns in the dialog. The BPMN can be considered as a set of element nodes N and edges E. As a path in the BPMN is traversed, for each element of type activity, a user question and an assistant response describing the activity are created. For each exclusive gateway, an agent question is associated with the gateway, and the response from the user for the chosen path is created. Hence, an exclusive gateway always results in an assistant asking a question. In case the element type is a parallel gateway, we first create a user query and an agent response that reflects the first activities on each of the parallel paths. We then navigate each valid path with the user specifically choosing the first tasks in each of the parallel branches as a query. We thus create multiple dialog flows for each process having exclusive and parallel gateways. Once all the steps are complete, we create a user query and assistant response depicting the end of the pro-

cess. The objective of using the data is to learn the next response, which takes (i) the dialog history $h = \{c_1^u, c_1^a \ldots, c_u^i\}$, and (ii) a process description (as NL or CNL) and predict the agent response ($y = c_i^a$).

Further, we generate the assistant response for each element in BPMN using an LLM (Mistral (Jiang et al., 2023)). The activity label *Conduct direction hearings*, and the participant name (*Organization*) is used to generate the utterance: *'The Organization will conduct hearings to give directions'*. A title for each process is also generated using an LLM by providing the process description as input. For example, when the user starts a conversation, the user utterance we generate starts as *what is the first step in the process for ⟨process title⟩?* In the future, it is possible to use the ProcessChat data to baseline a retrieval augmented generation framework (Gao et al., 2024) by retrieving relevant process descriptions to generate assistant responses.

3.3 ProcessChat Data

The ProcessChat[4] data is divided into train, test, and validation splits at the process level so that processes are not shared across splits. Each process generates multiple conversation flows depending on the number of exclusive gateways and the paths taken by the user. Again, each conversation flow generates multiple samples where each sample denotes the expected response of the assistant in a single turn given the conversation context. The statistics of the dataset, including processes, conversation flows, samples, and various question types are given in Table 1. Next-step questions include user queries related to the steps of the process (first or next), whereas resource questions consist of role and responsibility-related queries. Assistant questions are generated to identify the outcome of decision points (exclusive gateways), which in turn will be asked by the assistant to decide the next step.

4 Baseline System

We aim to investigate the performance of LLMs on process dialog generation tasks, with a particular focus on decoder-based LLMs. In our baseline experiments, we explore zero-shot prompting, few-shot learning, and fine-tuning. For the fine-tuning process, we employ the LoRA method (a Low-Rank Adapter framework) (Hu et al., 2021). To the best of our knowledge, this is the first instance of LoRA being applied to fine-tuning LLMs for process-grounded dialog tasks.

[4] https://anonymous.4open.science/r/ProcessChat-4FF4/README.md

Data Split	train	val	test
Processes	51	7	15
Conversation Flows	230	24	62
Samples	3092	439	846
Next-step Questions	2081	290	551
Resource Questions	665	107	182
Assistant Questions	355	42	122

Table 1: Dataset statistics

4.1 Prompting

We investigate a varied collection of pre-trained LLMs, all based on the transformer architecture (Vaswani et al., 2017). This collection comprises three distinct LLMs, each trained on instruction data and available in multiple versions with varying parameter sizes. This results in models with parameter sizes ranging from 7 billion to 47 billion.

Mistral-7B-Instruct-v0.2 (Jiang et al., 2023): The model is a 7.3B parameter which is an instruction-tuned version of the base Mistral 7B model. It has outperformed larger models in multiple NLP benchmarks.

granite-13b-chat-v2 (IBM, 2024): The model was initialized from a base model trained on 1.25 trillion tokens and also relies on synthetic data that is designed to improve the model's conversational, safety, and instruction following capabilities. The model is aligned to chat instructions including a conversation history.

Mixtral-8x7B-Instruct-v0.1 (Jiang et al., 2024): This model from Mistral AI is a "mixture of experts" model which is a decoder-only model where the feedforward block picks from a set of 8 distinct groups of parameters allowing it to use only a fraction of the total set of parameters per token. It has 46.7B total parameters but only uses 12.9B parameters per token.

These models were chosen primarily because they support a large (around 8K) input token size. To compare the results from these models with fine-tuning based results we considered the test set only by leaving the training and validation splits. We use few-shot learning, also denoted as K-shot, with

K representing the number of examples provided, where in our case, examples are randomly sampled from the training set. To understand the effect of the number of few-shot examples we experimented with 0, 1, 3, and 5 examples in the prompt. The prompt we used is similar to what is shown in Figure 4 included in Appendix A, where some few-shot examples consisting of conversations represented in the same format are added before the current conversation.

4.2 Fine-tuning

We experimented with fine-tuning two LLMs, IBM's granite-13b-base-v1 (IBM, 2024) and the open-source Mistral-7B-v0.1 (Jiang et al., 2023) from Mistral AI[5]. To conserve GPU memory, fine-tuning is done in a parameter-efficient manner using LoRA (Hu et al., 2021) with a rank of 4. LoRA (short for Low-Rank Adaptation) is a technique that freezes the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the Transformer (Vaswani et al., 2017) architecture, thereby greatly reducing the number of trainable parameters for fine-tuning. We chose this method because fully fine-tuning such LLMs is expensive and can lead to loss of generalizability due to catastrophic forgetting. The models are trained on the samples in the training set and tested on the samples in the test set, while the samples in the validation set are used for evaluating model performance during training. The models are trained for 3 epochs with a batch size of 2 on a single NVIDIA A100 GPU with 80 GB memory. We use the latest checkpoint as the final one for testing the models. We spent around 12 GPU hours for fine-tuning granite-13b-base-v1 and around 8 GPU hours for fine-tuning Mistral-7B-v0.1 (so the total computational budget was around 20 GPU hours).

For each turn of the assistant in the dataset, we obtain a sample data point where the input to the model is a prompt consisting of a system instruction along with the process description and the dialogue history, and the output is the response given by the assistant to the user. For the Court Hearing process described above, a set of example input prompts and output responses are shown in Appendix A.

[5] https://mistral.ai

4.3 Evaluation and Baseline Results

We considered three settings based on the inclusion of process description for evaluating the ability of the models (as assistants) to generate responses:

1. Including Both NL & CNL descriptions
2. Including only NL description
3. Including only CNL description

We compare the assistant response generated by the model with the expected response of the assistant at each turn in the test set using ROUGE-L (Lin, 2004) and BLEU (Papineni et al., 2002) scores. We then compute average ROUGE-L and BLEU scores for all the samples in the test set to get the overall performance of the model. We also separate out samples where the user asks resource (i.e. role/responsibility) based questions from the test set and measure the average ROUGE-L and BLEU scores for only those samples. This gives us an idea of how good the model is at identifying the roles of each resource in the process. Furthermore, we check whether the model is generating questions at the right turn in the dialogue or not. To do this, we compute two additional metrics on the test set: (i) Question Precision (Q-Pr) which measures how often a generated question is expected at that turn, and (ii) Question Recall (Q-Re) which measures how often a question is generated when it is expected at that turn.

4.3.1 Prompt-based results

From Table 2, it is evident that more few-shot examples always help the model to achieve better scores. We see that the Mixtral-8x7B-Instruct-v0.1 model is outperforming the other two models in all metrics. Also, from the results we see that using CNL descriptions, with or without NL descriptions, has a definite advantage over using only NL descriptions in terms of ROUGE-L and BLEU scores. However, it is unclear whether using CNL descriptions in conjunction with NL descriptions is better than only using CNL descriptions or vice versa. We also notice that these models are better at answering resource-related questions in general. While Mixtral-8x7B-Instruct-v0.1 can show good Question Precision, all the prompt-based models show poor Question Recall. This may be because the models are not able to understand the process well enough to know when a question must be asked.

Interestingly, the performance of Mistral-7B-Instruct-v0.2 and Mixtral-8x7B-Instruct-v0.1 in the five-shot setting on the resource-related questions

Expt.	Model	Policy	ROUGE-L (overall)	BLEU (overall)	ROUGE-L (resource)	BLEU (resource)	Q-Pr	Q-Re
LoRA	granite-13b-base-v1	NL+CNL	**0.803**	**0.560**	**0.808**	**0.312**	0.745	0.860
	granite-13b-base-v1	NL	0.654	0.360	0.779	0.244	0.617	0.647
	granite-13b-base-v1	CNL	0.758	0.494	0.799	0.294	**0.781**	**0.877**
	Mistral-7B-v0.1	NL+CNL	0.497	0.221	0.492	0.116	0.172	0.180
	Mistral-7B-v0.1	NL	0.466	0.181	0.495	0.122	0.135	0.106
	Mistral-7B-v0.1	CNL	0.482	0.215	0.492	0.120	0.226	0.238
5-shot Prompt	granite-13b-chat-v2	NL+CNL	0.522	0.237	0.602	0.207	0.200	0.008
	granite-13b-chat-v2	NL	0.457	0.169	0.570	0.157	0.050	0.008
	granite-13b-chat-v2	CNL	0.452	0.207	0.396	0.139	0.133	0.016
	Mistral-7B-Instruct-v0.2	NL+CNL	0.562	0.306	0.544	0.177	0.000	0.000
	Mistral-7B-Instruct-v0.2	NL	0.511	0.222	0.567	0.154	0.333	0.008
	Mistral-7B-Instruct-v0.2	CNL	0.577	0.307	0.645	0.204	0.250	0.008
	Mixtral-8x7B-Instruct-v0.1	NL+CNL	0.653	**0.379**	0.757	0.284	0.823	0.114
	Mixtral-8x7B-Instruct-v0.1	NL	0.577	0.274	0.732	0.218	**1.000**	0.033
	Mixtral-8x7B-Instruct-v0.1	CNL	**0.656**	0.363	**0.809**	**0.286**	0.667	**0.147**
3-shot Prompt	granite-13b-chat-v2	NL+CNL	0.509	0.217	0.592	0.215	0.222	0.049
	granite-13b-chat-v2	NL	0.456	0.163	0.567	0.152	0.204	0.082
	granite-13b-chat-v2	CNL	0.463	0.187	0.554	0.195	0.111	0.016
	Mistral-7B-Instruct-v0.2	NL+CNL	0.549	0.301	0.482	0.160	0.636	0.057
	Mistral-7B-Instruct-v0.2	NL	0.522	0.211	0.642	0.173	1.000	0.008
	Mistral-7B-Instruct-v0.2	CNL	0.578	0.290	0.705	0.226	0.333	0.024
	Mixtral-8x7B-Instruct-v0.1	NL+CNL	0.651	0.355	0.785	0.289	0.692	0.147
	Mixtral-8x7B-Instruct-v0.1	NL	0.573	0.252	0.761	0.223	0.857	0.049
	Mixtral-8x7B-Instruct-v0.1	CNL	0.634	0.335	0.797	0.280	0.695	0.131
1-shot Prompt	granite-13b-chat-v2	NL+CNL	0.458	0.183	0.124	0.497	0.000	0.000
	granite-13b-chat-v2	NL	0.386	0.468	0.113	0.131	0.333	0.016
	granite-13b-chat-v2	CNL	0.457	0.188	0.531	0.151	0.000	0.000
	Mistral-7B-Instruct-v0.2	NL+CNL	0.489	0.238	0.374	0.087	0.000	0.000
	Mistral-7B-Instruct-v0.2	NL	0.478	0.194	0.552	0.142	0.000	0.000
	Mistral-7B-Instruct-v0.2	CNL	0.479	0.231	0.407	0.091	0.000	0.000
	Mixtral-8x7B-Instruct-v0.1	NL+CNL	0.577	0.283	0.600	0.169	0.750	0.024
	Mixtral-8x7B-Instruct-v0.1	NL	0.496	0.205	0.563	0.148	0.000	0.000
	Mixtral-8x7B-Instruct-v0.1	CNL	0.581	0.287	0.620	0.157	0.500	0.016
0-shot Prompt	granite-13b-chat-v2	NL+CNL	0.057	0.012	0.015	0.055	0.000	0.000
	granite-13b-chat-v2	NL	0.071	0.063	0.019	0.017	0.000	0.000
	granite-13b-chat-v2	CNL	0.044	0.010	0.052	0.021	0.000	0.000
	Mistral-7B-Instruct-v0.2	NL+CNL	0.388	0.454	0.102	0.163	0.000	0.000
	Mistral-7B-Instruct-v0.2	NL	0.371	0.123	0.428	0.094	0.000	0.000
	Mistral-7B-Instruct-v0.2	CNL	0.390	0.175	0.458	0.104	0.200	0.008
	Mixtral-8x7B-Instruct-v0.1	NL+CNL	0.192	0.113	0.175	0.052	0.000	0.000
	Mixtral-8x7B-Instruct-v0.1	NL	0.268	0.109	0.236	0.054	0.000	0.000
	Mixtral-8x7B-Instruct-v0.1	CNL	0.068	0.036	0.118	0.043	0.000	0.000

Table 2: Results of various baseline approaches on the test set. The best scores obtained by the fine-tuned and prompt-based models are marked in bold.

using only CNL descriptions is better than the performance using only NL or a combination of NL and CNL descriptions. We surmise that this may be because the model gets a compact description of the process in CNL which helps it to infer the answers to the resource-related questions in a succinct way using the dialog context.

4.3.2 Fine-tuning results

From Table 2 we see that the fine-tuned granite-13b-base-v1 is better than the fine-tuned Mistral-7B-v0.1 by a large margin. In fact, the fine-tuned Mistral-7B-v0.1 seems to be performing worse than the prompt-based Mistral-7B-Instruct-v0.2 model. This may be because of higher quality instruction tuning done by Mistral AI which makes the model very robust to various types of human instructions[6]. However, in terms of ROUGE-L and BLEU scores, the fine-tuned granite-13b-base-v1 is better than all the prompt-based models. We see that using both NL and CNL descriptions is better than only using CNL descriptions, which in turn is better than only using NL descriptions. This may be because fine-tuning allows the model to get useful information from both NL and CNL descriptions which may be complementary to each other. While the fine-tuned granite-13b-base-v1 may have lower Question Precision than some of the prompt-based models, it has the highest Question Recall among all the models. Interestingly, using CNL descriptions alone achieves higher Question Precision and Question Recall than using only NL or both NL and CNL descriptions. This may be because the CNL description indicates the decision points, which may not be indicated in the NL description.

As observed during prompting, the performance of both the models on the resource-related questions using only CNL descriptions is not much worse than the performance using only NL or a combination of NL and CNL descriptions. In the case of granite-13b-base-v1, it is actually better than using only NL descriptions.

Figure 3 shows a graphical representation of the scores attained by the models (considering only fine-tuning and five-shot prompting with both NL and CNL descriptions), to compare average ROUGE-L scores for the samples in the test set for the overall assistant responses to all types of user questions, the responses related to questions on the next step, and the responses related to questions on a resource. The fine-tuned model performed better in all three scenarios, while the prompt-based models did well for only resource-related queries.

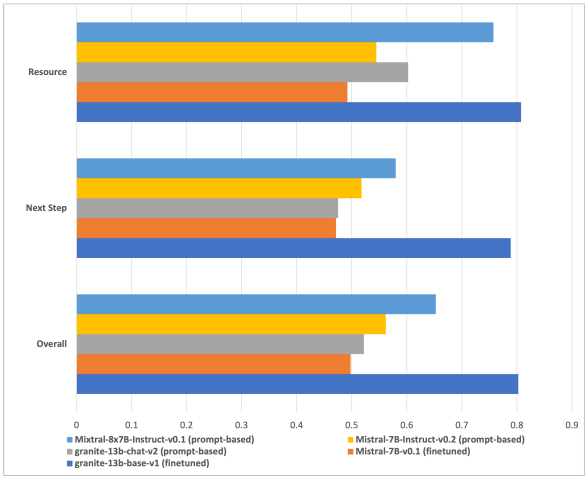

Figure 3: ROUGE-L scores of answers provided by the system for various kinds of user queries (overall, resource-related, and next-step related), considering only fine-tuning and five-shot prompting for the setting where both NL and CNL process descriptions are used.

5 Conclusion and Future Work

In this work, we presented a view of providing chatbot interactions with grounding on process knowledge. We conduct experimental results to verify the suitability of LLMs to infer and provide guidance based on process descriptions in natural language and constrained natural language. We generate and release the ProcessChat dataset, which contains 316 dialog flows grounded on 73 process models. We propose the baseline solutions that evaluate a prompt-based model and a parameter-efficient fine-tuned model using LoRA. Our baseline results show significant improvement using a fine-tuned model with a small training dataset of 230 dialog flows. We foresee multiple directions for future research: i) The ProcessChat dataset is suited for step-by-step guidance. It can be extended to provide an interaction where the user should can ask any specific query regarding the process. This would require generating dialog interactions on any step in the process. ii) The dataset does not consider data flow as the specification had limited information on data artifacts. Incorporating data perspective would ensure a comprehensive experience. iii) Finally, this dataset can be extended for interactions across multiple processes in a repository.

[6]Notably, fine-tuning Mistral-7B-Instruct-v0.2 itself using LoRA yields even poorer results, presumably because the effect of instruction tuning is destroyed.

6 Limitations

In the process notation, splitting parallel gateway multiplies the incoming sequence flow into several outgoing sequence flows that run simultaneously. A joining parallel gateway waits for all incoming sequences to terminate before combining them all in one outgoing flow. This leads to several variations in the flow execution. For example, activities in one parallel path can be interspersed with activities in another parallel path. However, To reduce redundancy, we considered the sequence of activities in one flow intact. Hence, in our dataset, a process with n parallel flows will lead to only n output flows, where a user completes dialog interaction of one parallel path and then proceeds to the next path.

Another limitation in the generation of the dataset is the use of an LLM to form a assistant response or question from the BPMN specification. As the specification only contains activities names and the actors performing it, we generate the assistant response using **Mistral-7B-Instruct-v0.2 (Jiang et al., 2023)**. In the conversation presented for the court hearing process, assistant responses at T1, T2 or the question at T3 was generated by the LLM. Hence, the assistant response or the ground truth response is dependent on the LLM we have used. However, we have manually evaluated these generated responses.

References

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 *BPM 2022 International Workshops, Münster*, volume 460 of *LNBIP*, pages 315–321.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Michael Grohs, Luka Abb, Nourhan Elsayed, and Jana-Rebecca Rehse. 2023. Large language models can accomplish business process management tasks. In *BPM 2023 International Workshops, Utrecht*, volume 492 of *LNBIP*, pages 453–465.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

IBM. 2024. Granite foundation models. https://www.ibm.com/downloads/cas/X9W4O6BM.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Henrik Leopold, Jan Mendling, and Artem Polyvyanyy. 2014. Supporting process model validation through natural language generation. *IEEE Trans. Software Eng.*, 40(8):818–840.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Anselmo López, Josep Sànchez-Ferreres, Josep Carmona, and Lluís Padró. 2019. From process models to chatbots. In *CAiSE 2019, Rome, Proceedings*, volume 11483 of *LNCS*, pages 383–398.

OMG. 2011. Business Process Model and Notation (BPMN), Version 2.0. *Object Management Group*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Josep Sànchez-Ferreres, Han van der Aa, Josep Carmona, and Lluís Padró. 2018. Aligning textual and model-based process descriptions. *Data & Knowledge Engineering*, 118:25–40.

Han van der Aa, Claudio Di Ciccio, Henrik Leopold, and Hajo A. Reijers. 2019. Extracting declarative process models from natural language. In *CAiSE 2019, Rome, 2019, Proceedings*, volume 11483 of *LNCS*, pages 365–382.

Han van der Aa, Henrik Leopold, and Hajo A. Reijers. 2015. Detecting inconsistencies between process models and textual descriptions. In *BPM - 13th International Conference, Austria, 2015, Proceedings*, volume 9253, pages 90–105.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

A Appendix

Set of example input prompts and output responses are given by Figure 4, where each input terminates with the user's utterance and the expected output is given by the assistant's response.

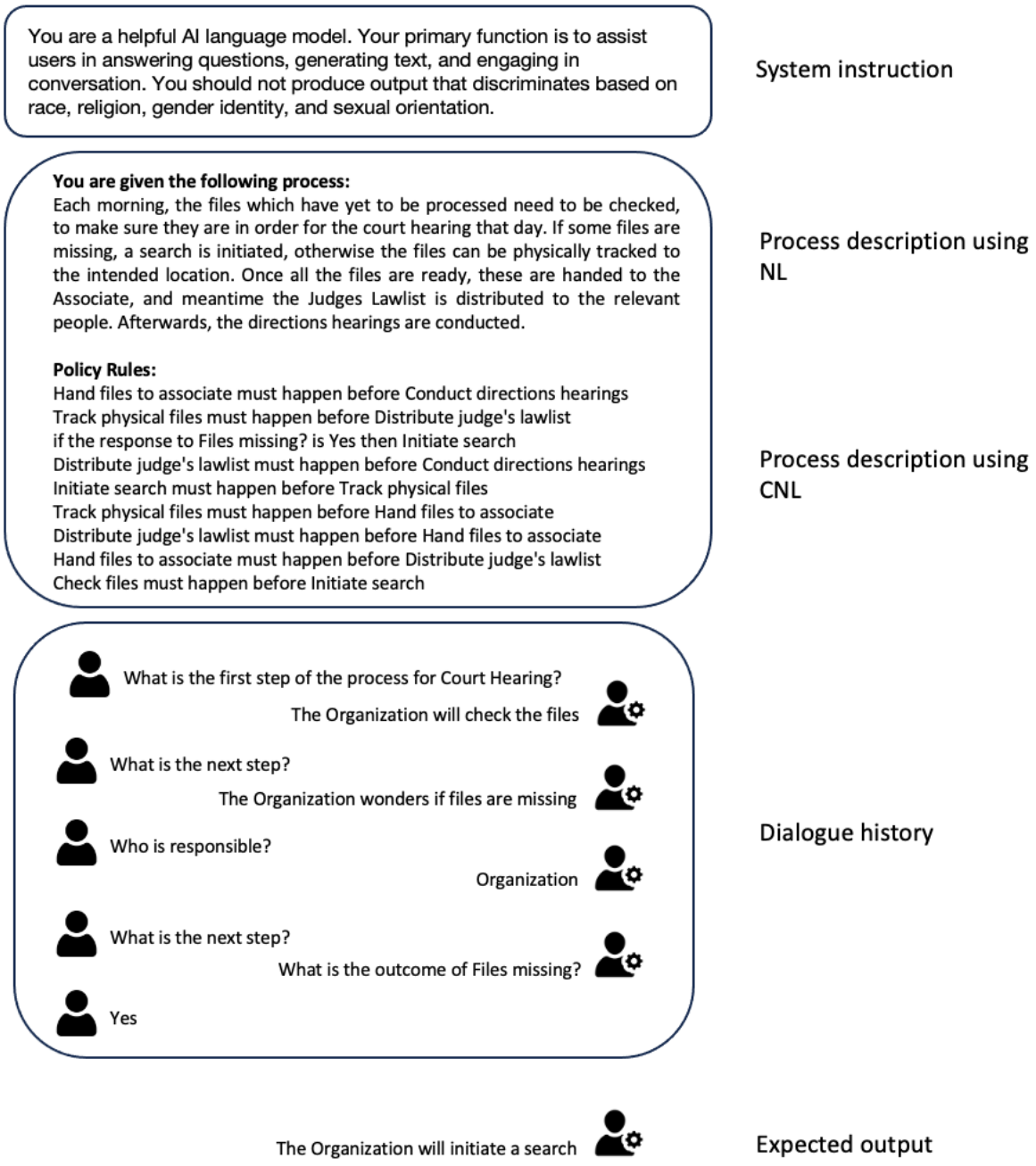

Figure 4: Example input prompt and expected response