



Ambiguity Detection in Business Process Descriptions: An Evidence and an Automated Approach

Hugo A. López¹(✉) , BingKun Feng¹, Jonas Lindner¹,
Marco Franceschetti² , and Amine Abbad-Andaloussi²



¹ Department of Mathematics and Computer Science,
Technical University of Denmark, Kgs. Lyngby, Denmark
hulo@dtu.dk

² Institute of Computer Science, University of St. Gallen,
St. Gallen, Switzerland
{marco.franceschetti,
amine.abbad-andaloussi}@unisg.ch

Abstract. This paper examines the role of ambiguity in process descriptions and how feasible it is to identify ambiguities using automated techniques. Process descriptions are natural language specifications of a business process. An ambiguity occurs whenever part of the specification can be understood in multiple ways. Ambiguities harm model quality, as well as comprehension and business process execution. This paper provides two contributions. First, it examines the prevalence of ambiguities in process descriptions. Second, it explores how the creation of computational artifacts could support the identification of ambiguities at design time. We constructed a dataset of process descriptions considering multiple authors, writing styles, and possible uses in academia and industry. In a dataset of $N = 71$ process descriptions, we show that ambiguities are a prevalent phenomenon, where 94,3% of the descriptions exhibit at least one type of ambiguity. Concerning identification, we can see that standard classifier techniques fall short in identifying ambiguities. At the same time, the application of pre-trained and fine-tuned LLMs improves the identification of ambiguity significantly. By demonstrating the presence of ambiguities in process descriptions and identifying them, our work aims to influence research on the application of Natural Language Processing (NLP) in Business Process Management (BPM) by refining pipelines to consider multiple interpretations. Moreover, it advocates for the transformative role of NLP as a tool that can promote clearer and more reliable process definitions.

Keywords: Ambiguity Detection · Natural Language Processing · Process Model Extraction

1 Introduction

This paper concerns the design of correct process models. The design is crucial in the BPM lifecycle as it ensures that the downstream analysis and execution

phases are done correctly. However, the modeling discipline needs to integrate different sources of information, many of them unstructured. We focus on business process descriptions as a consolidated view of a process written in a natural language. One of the major challenges in the extraction of formal specifications (i.e., process models) from process descriptions is the existence of ambiguities. Ambiguity in process descriptions can stem from unclear relationships between process elements, vague references, underspecified requirements, and inconsistent information [13]. Ambiguities in descriptions can lead to multiple interpretations, causing inconsistencies in process implementation and execution [2, 28]. Resolving ambiguities wrongly can result in operational inefficiencies, increased costs, and failure to meet business objectives [28]. Addressing ambiguities in process descriptions is essential to ensure that designed processes are accurately understood and implemented. By reducing ambiguities, organizations can achieve better alignment between their business objectives and process outcomes, enhance process performance, and improve overall operational efficiency [13].

In general, there are two primary methods for handling ambiguity: preventing it or detecting and resolving it. While a preventive approach would be ideal, it would require that requirements are formally stated using precise language with unique meanings in controlled languages, context-free grammar, or sentence templates [28]. The formalization requirement is difficult to achieve in real life: process information is not written by computer scientists but by domain specialists such as legal practitioners or clinicians. In these cases, formalization of their existing documentation is simply not feasible [16, 21]. In this paper, we will focus on the detection part, as it is a topical aspect for new pipelines of process extraction using Large Language Models (LLMs).

Text-to-Process (T2P) pipelines [1, 7, 10, 19, 22, 25] include the set of techniques designed to extract a formal specification (i.e., a process model) from a natural language description. Traditionally, T2P pipelines break down the process generation task into well-defined NLP tasks, such as Named Entity Recognition and Relationship extraction. Recently, the emergence of LLMs has facilitated the simplification and enhancement of pipeline accuracy using prompts that incorporate multiple dimensions, leveraging their large context to achieve generalization [7, 19, 25]. However, most T2P pipelines assume that their process descriptions are unambiguous, overlooking the impact of ambiguities on their output. An exception is the work by Nour et al. [25], which addresses potential errors through their tool. Yet, ambiguity in process descriptions is not inherently a correctness issue. As shown by Franceschetti et al. [13], multiple *correct interpretations* can arise from an ambiguity, and sometimes the ambiguities are established *by design* as intentional elements given to allow multiple interpretations of a text, and resolving them automatically may bias the output.

This paper makes two contributions. The first is an investigation into the prevalence of ambiguities within business process descriptions. The second examines how machine learning and deep learning techniques can be used to detect these ambiguities. By doing so, we aim to highlight their frequent occurrence in virtually all process descriptions we examined and advocate for their consider-

ation as a task within T2P pipelines. As a byproduct, this research presents a new process description dataset, including 71 texts sourced as explanations of both imperative and declarative process descriptions, including industrial and academic processes.

Structure of this Paper. Section 2 introduces the theoretical framework of ambiguities from [13], instantiated to ambiguities in natural language. Section 3 documents the analysis of ambiguities in our process description dataset. Section 4 presents the results on ambiguity detection using deep learning models. Section 5 discusses related work. Section 6 describes the reproducibility of the experiments and code. Section 7 points towards the implications of ambiguity presence in NLP+BPM tasks, and Sect. 8 concludes.

2 Background: Ambiguity in Process Descriptions

This section recalls the definitions of ambiguity in Business Process Management (BPM) and instantiates it in business process descriptions. Ambiguity can arise in various artifacts, including textual specifications, formal representations of process models (such as BPMN), and event logs from the monitoring phase. This study focuses on the first type of artifact: specifications in natural language.

Our paper takes as a departing point the taxonomy of ambiguities in BPM [13]. Considering the taxonomy, ambiguities in process descriptions fall under the category of *descriptive ambiguities*. They can be classified into four distinct types:

Unclear Relations (T1). It arises when relationships between process elements are not specified, confusing whether tasks are sequential, parallel, or mutually exclusive. For instance, the statement “*Prepare the meeting agenda and distribute it to participants*” does not clarify if these actions should be done simultaneously or sequentially. A clearer version would be: “*First, prepare the meeting agenda. After the agenda is prepared, distribute it to the participants.*”

Unclear References (T2). This ambiguity occurs when it is unclear which part of the process a statement refers to, creating confusion about which elements are involved or affected. For example, in the sentence “*Manager requires employees to submit files to the system, he should review the document before approval*” the phrase “*He should review (...)*” does not specify who “*He*” refers to, causing ambiguity. A clearer version would be “*(...) The project manager should review the document before approval.*”

Underspecifications (T3). Ambiguity of this nature occurs when a process description leaves out key details, either by choice or oversight. Such omissions—whether intended for flexibility, due to insufficient knowledge, or accidentally—can make roles, tasks, or necessary actions within the process unclear. For instance, in “*Submit the report by the end of the month*”, there is uncertainty regarding who is tasked with the report submission. A more explicit version would be: “*The team leader must submit the monthly financial report to the*

finance department by the end of the month". Likewise, the use of abbreviations or technical jargon like *"The system should support SSL"* can introduce confusion without clarification. A more precise statement could be: *"The system should support the Secure Sockets Layer (SSL) protocol to ensure data encryption during transmission"*.

Inconsistent Specifications (T4). This ambiguity arises from conflicting requirements or descriptions within the same process specification, complicating the creation of a consistent process model. Such inconsistencies force the modeler to choose which instructions to follow, potentially resulting in a process that deviates from intended practices. For instance, *"Submit the application online. Send the completed form via mail to the office."* is unclear. A more consistent instruction would be: *"Submit the application online. Do not send the form by mail; keep a digital copy for your records"*. It should be noted that ambiguity judgments often require considering the entire business process model, as interpretations may vary across different contexts and observers.

Process descriptions are not the only BPM artifacts prone to these types of ambiguities; process models and event logs can also exhibit similar and additional ambiguities, as noted in [13]. For instance, [20] highlights label consistency as a common problem in process models, and [12] documents how ambiguous process models add cognitive load and lower performance in the understanding of process models. Nevertheless, our study focuses solely on process descriptions, which are well-suited for analysis through pure NLP techniques. Identifying ambiguities in artifacts that combine text with diagrams is reserved for future work.

3 Dataset Collection: Ambiguities in BPM Descriptions

Our first contribution is set to answer the research question *RQ1: To what extent are ambiguities present in process descriptions?*. To answer this question, we instantiated the taxonomy in [13] and evaluated the presence of ambiguities in a realistic dataset of process descriptions. For the construction of the dataset, we considered balancing a set of criteria to provide a baseline. First, we chose only descriptions written in English as the original language. Second, we wanted to limit potential overfitting when training with texts from the same authors, who tend to repeat their writing style, lexical diversity, and lexical sophistication [24]. Third, we considered the intent to express imperative and declarative processes, as their writing styles are different [4]. While the BPM community has often used the PET dataset [5] for many downstream NLP tasks in process extraction, this dataset does not fit our requirements, as its process descriptions only consider sources collected by one author [14], and it only considers imperative processes¹. To tackle these shortcomings, we combined two sources of process descriptions. The first includes a dataset of 37 process descriptions combining imperative and declarative style targets [23]. In particular, 3 process descriptions were sourced

¹ Moreover, [2] and [13] have already identified sources of ambiguities for [5], without quantifying its extent.

from the BPM Academic Initiative, 15 described declarative process models, and 19 described imperative process models sourced from process ethnographies as part of university coursework. The second dataset included 34 process descriptions created at an MSc course in BPM at the Technical University of Denmark. In this course, students executed ethnographies of multiple organizational processes to generate process descriptions, following a reconciliation process with multiple observers to generate each process description. From the 34 process descriptions sourced in this way, 33 descriptions had a declarative style, and one had an imperative style. All data used in this study were voluntarily shared for research purposes.

3.1 Dataset Analysis via Semantic Annotations

To analyze the extent to which process descriptions are affected by ambiguities, we followed a semantic annotation approach. It included the following steps:

1. **Data preprocessing:** We focused on preserving the original content's integrity while enabling sentence-level analysis. Special characters were converted to prevent splitting errors. Next, we eliminated formatting, transforming elements like bullet points, numbers, and lists into standard sentences. We also corrected any sentences containing punctuation errors from earlier stages. Finally, titles and headings were reviewed to ensure proper contextual understanding, occasionally requiring manual adjustments.
2. **Construction of an annotation guideline:** This step considered the generation of a set of annotation rules and examples to make the annotation process unambiguous and replicable for multiple annotators. The resulting guidelines can be accessed from our GitHub repository².
3. **Annotation process:** Each sentence was labeled in a binary way to denote whether an ambiguity type is present. The tagging system included the ambiguity type, the reasons for the annotation, and the number of ambiguities encountered in the sentence. Any sentence can bear multiple ambiguity tags, as their types are non-exclusive. The annotation follows the patterns provided in the annotation guidelines and involves multiple consensus rounds by the authors. Initially, sentences are annotated based on the guidelines' criteria, and these annotations are reviewed in expert meetings to ensure accuracy and consistency. The experts included one associate professor and two postdocs active in BPM, and developers of the ambiguity taxonomy in [13], as well as one expert with seven years of experience in the development of NLP applications in BPM. Their feedback led to final annotations integrated into the dataset, with three iterations of the annotation guidelines between February and April 2024. The annotation highlighted ambiguities affecting the process model's construction, specifically those in the sentences immediately preceding and following the process description. Figure 1 illustrates an example of the annotation framework applied using the Inception tool [18].

² <https://github.com/hugoalopez-dtu/ambiguityDetector>.

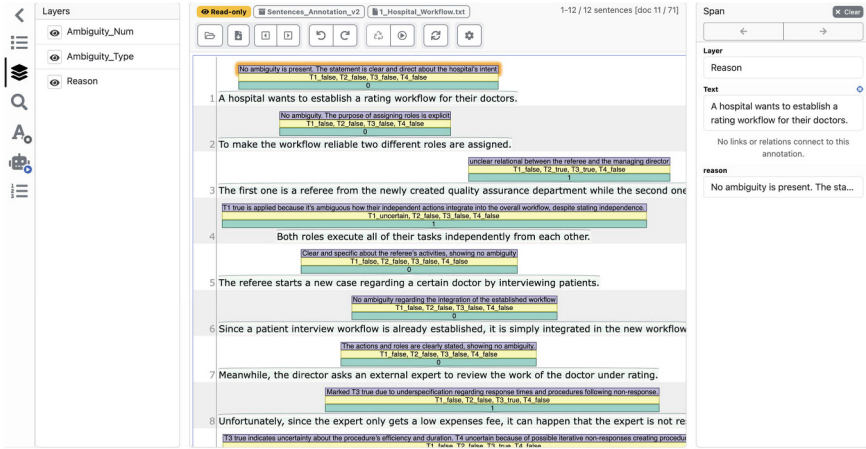


Fig. 1. Ambiguity annotation example.

3.2 Results on the Presence of Ambiguities in Process Descriptions

Following preprocessing and filtration, we analyzed 71 BPM descriptions and 1,039 sentences. Among these, 67 descriptions (accounting for 94.3% of the total) exhibited at least one instance of ambiguity, with an average of 4.60 ambiguities noted per description and up to a maximum of 31 ambiguities in any given description. Figure 2 shows the distribution per number of ambiguities. On a sentence level, we found 255 sentences exhibiting at least one ambiguity, roughly 24.54% of the total. The distribution of sentences based on the number of ambiguities detected is shown in Fig. 4. Approximately 187 sentences contained a single type of ambiguity, with fewer sentences having multiple types. Approximately 10 sentences included three types of ambiguities, and no sentence featured all ambiguity types. This pattern suggests that ambiguities, though evident in BPM descriptions, are typically singular within sentences, emphasizing the need to tackle each ambiguity individually to enhance clarity and comprehension.

To illustrate the existence of ambiguities, we will use one annotated process description (originally published in the BPMAI initiative [32] and part of the PET dataset [5]). Figure 3 exhibits the Computer Repair Service description, annotated with its ambiguities according to the annotation guidelines. We found four ambiguous sentences in a six-sentence process description. We will discuss the ambiguities per type. Line 3 shows a case of unclear relations (T1): it is unclear that activities should be executed sequentially, in parallel, in non-deterministic choice, or with/without repetitions. Line 2 exhibits a case of unclear references (T2): the fragment “*the process continues*” is unclear in defining how. Line 5 shows a case of underspecification (T3): while the fragment describes “*proper system functionality testing*”, it is unclear what constitutes a “proper” test. Finally, multiple ambiguities can appear in a single sentence. For example, the case in Line 6 shows a problem with underspecifications (T3) (e.g.,

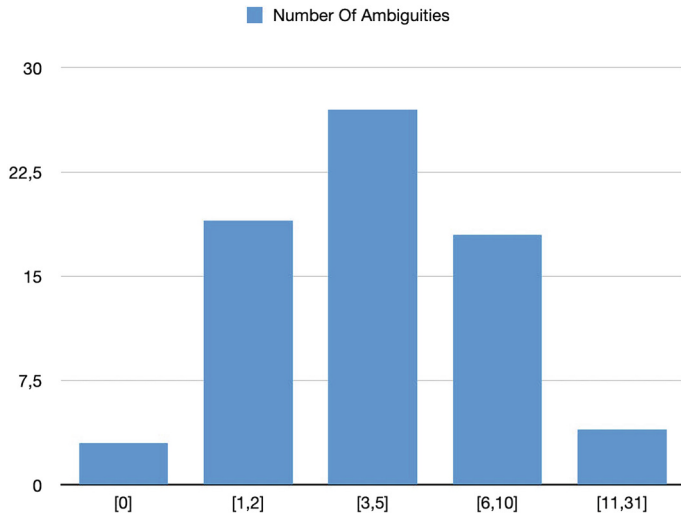


Fig. 2. Distribution of number of ambiguities.

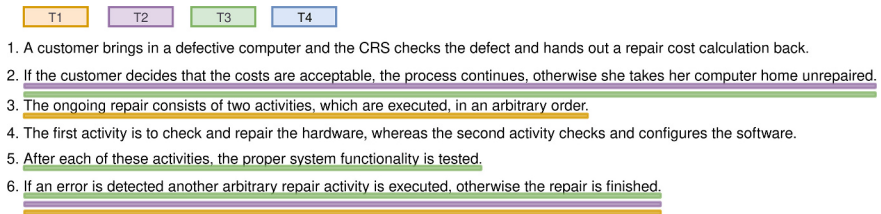


Fig. 3. Annotation of ambiguities in the computer repair service example.

what constitutes an arbitrary repair activity?), unclear references (T2) (e.g., is the error of local scope of the repair, or global and covers diagnostic?), and unclear relations (T1) (in what order should activities be orchestrated?).

Inconsistent specifications were the type of ambiguity with the smallest representation in the dataset. We will illustrate one case. The sentences “*When the meal is served, the waiter must stand by the customer to finish his dish.*” and “*Finally, the client can leave and the waiter is free to take another incoming customer, as well as the client is capable of re-entering the restaurant.*” are part of the same process description. The contradiction occurs whenever the customer receives his meal and leaves, leaving the waiter in the position to follow the customer until finishing his dish (somewhere else) or start attending other customers. No matter the action, the waiter will be violating one of the rules.

We explored whether the specification style influences the number of ambiguities. Table 1 shows the ambiguity distribution across different specification styles. Here, our analysis was centered at the document level, where the categorization was given if most sentences in a description described rules or flows.

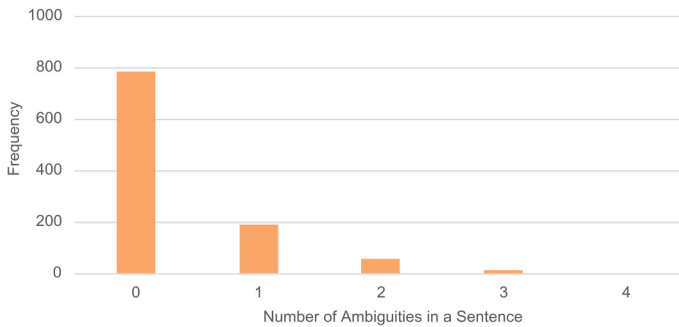


Fig. 4. Sentence distribution by number of ambiguities.

Table 1. Distribution of ambiguities across declarative and imperative specification styles

Spec. Style	N	Sentences	T1(%)	T2(%)	T3(%)	T4(%)	$\frac{\sum_1^4 T_i}{\sum \text{Sentences}}$
Declarative	50	708	71(0,1003)	29(0,0410)	105(0,1483)	7(0,0099)	0,2994
Imperative	21	331	51(0,1541)	22(0,0665)	51(0,1541)	3(0,0091)	0,3834

We notice that the distribution of ambiguity types is approximately constant in both categories, except for T1. We suggest that the increase in unclear relations may be related to the sentence structure used to describe flows. Most declarative specifications have sentences with derivatives of the pattern *if Guard then Activity*, for instance “*If the request is cancelled by the retailer the process will be cancelled*”. In contrast, imperative specifications tend to specify an implicit order that can be confusing. For example: “*when serving the pizza, the oven guy must first put the pizza in a box, slice it and put it on the cashier’s table*”.

After identifying ambiguities in the text, we aim to find the most current ones to prioritize computational support. Figure 5 shows the frequency distribution of each type of ambiguity. T3 (Underspecifications) is the most prevalent type, with 154 occurrences. This is followed by T1 (Unclear Relations) at 121 occurrences. Type T2 (Unclear References) has a total of 50 occurrences, while type T4 (Inconsistent Specifications) is the least frequent, with only 10 occurrences. The next question is whether a correlation exists between the types of ambiguity. Figure 6 illustrates the confusion matrix for each ambiguity type. We can observe a moderate positive correlation (0.29) between the T1 and T3 types. Conversely, T2 and T4 have low correlations with other groups, indicating that inconsistent specifications or unclear references are less likely to coexist in the same sentence.

To summarize, the semantic analysis performed allows us to conclude that 1) ambiguities are a widespread phenomenon in process descriptions, affecting almost every sample examined, including imperative and declarative process descriptions alike, 2) Ambiguities are not a spurious phenomenon, and process descriptions may have multiple types of ambiguities in combination, even in

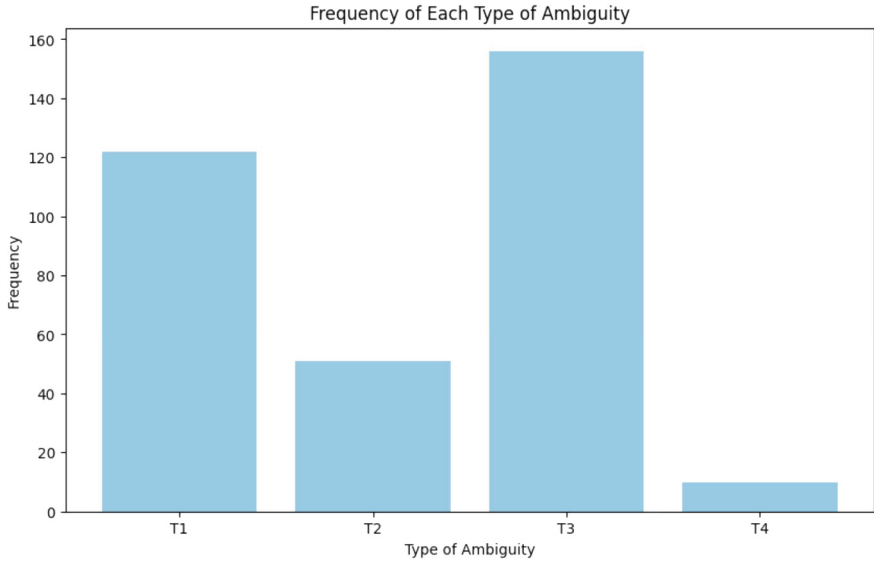


Fig. 5. Frequency of each type of ambiguity in the process description dataset.

the same sentences, 3) The most common ambiguity types existing in process descriptions are underspecifications and unclear relations, while ambiguities due to inconsistencies are rare, and 4) While most ambiguity types are independent, we can observe a moderate positive correlation between underspecifications and unclear relations.

4 Detection of Ambiguities Using Deep Learning

In this section, we explore our second research question: *RQ2: How effectively can machine learning techniques automate the process of detecting ambiguity?*. To answer it, we leverage our annotated dataset to build and fine-tune models, later comparing them against standard precision, recall, and F1 scores. In the following, we will report our model selection, training, and results.

Model Selection. The dataset analysis in Sect. 3 already provides us with some hints of the inability to build very precise classifiers since the imbalance between positive and negative classes will impact the accuracy of the predictions. While one could improve precision by a series of techniques, including data augmentation, oversampling, or even the syntactic manipulation of examples using LLMs, we decided to preserve the authenticity of the dataset to explore its applicability as is. Thus, we narrowed RQ2 into the sub-question *RQ2-1 What is the impact of contextual information provided by LLMs in detecting ambiguities in process descriptions?*. To answer this question, we considered four models. Our

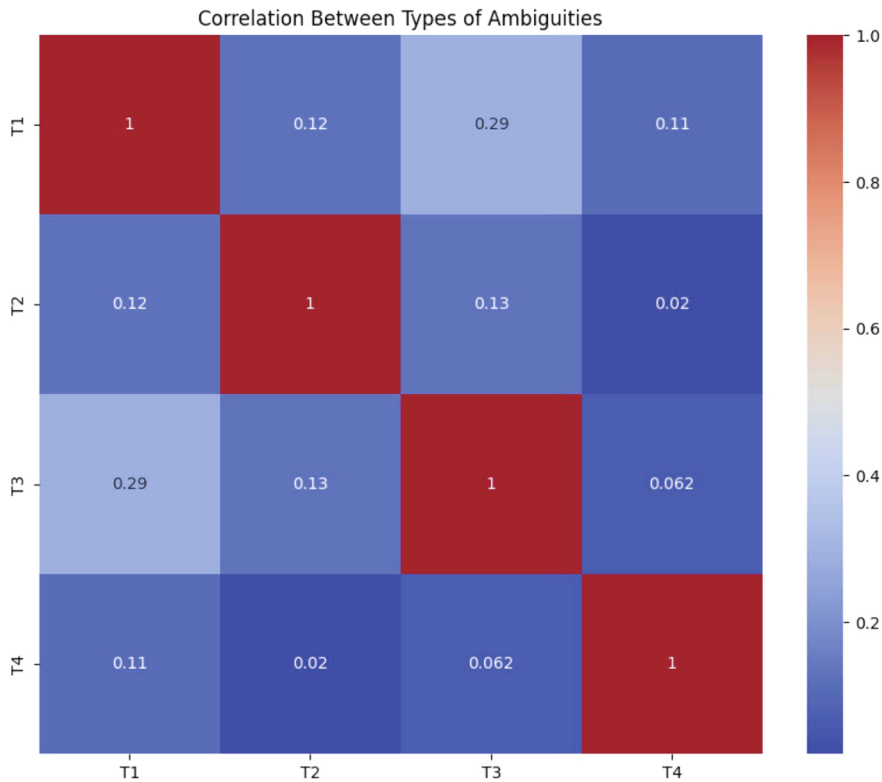


Fig. 6. Confusion Matrix for Ambiguity Types.

baseline was a trained Naive Bayesian classifier. For LLMs, we considered different architectures. First, an encoder-decoder architecture represented by Bidirectional Encoder Representations from Transformers (BERT-base) [8]. The second was Meta’s Llama-2 7B decoder-only transformer model [30]. Finally, we used the Guideline-following Large Language Model for Information Extraction (GoLLIE 7B) [27]. Our model selection criteria included reproducibility, representativeness, and a variety of architectures. All models are open source, facilitating reproducibility. While Bert and Llama are classic baseline models that can be fine-tuned for the purpose, Gollie constitutes a novel approach for decoder-only architectures. We included it since it is trained to follow annotation guidelines, facilitating our comparison with the manual annotation tasks in our dataset. Bert and Llama were trained and fine-tuned to optimize their performance for ambiguity detection. In the case of Gollie, we used the annotation guidelines as part of the prompts given to the information extraction task.

Data Preparation. We used a 5-fold cross-validation to ensure robust performance evaluation. This ensures that the models are trained on a subset of the

```

    @dataclass
    class Underspecification (Template):
        """Underspecification occurs when a process description
        deliberately or accidentally omits important details.
        Such omissions can be due to the need for flexibility,
        limited domain knowledge, or oversight. This ambiguity can
        result in an incomplete understanding of roles, tasks, or
        required actions within the process.
        """

        mention: str
        """
        The name of the characteristic. This can often be captured
        by less detailed adjectives. Examples include: "uncertain",
        "relative", "maybe", "as", "like".
        """

        is_ambiguity: bool # Whether the adjective in the sentence
        # has the ambiguous feature of Underspecification, 'True' if
        # it does, 'False' otherwise.
    
```

Fig. 7. Exemplary prompts used for Ambiguity Detection in Gollie (similar prompts were used for other T-classes)

data and evaluated on unseen sentences to assess their generalization capabilities. The labels for each ambiguity type were binarized for classification tasks.

Model Training. The annotated data was used to train models to detect sentence-level ambiguities. The training strategy is designed to classify sentences into positive and negative cases for each type of ambiguity. By training models to recognize both the presence (true) and absence (false) of ambiguities, we aim to improve the accuracy and reliability of ambiguity detection across the dataset. For the Bayesian classifier, we used a TF-IDF vectorizer with 5-fold cross-validation with shuffling for randomness. The same number of folds was used in BERT, as well as 5 epochs, a batch size of 8 training examples, and 500 warmup steps. We applied fine-tuning for the Llama model. This included 3 epochs, 4 gradient accumulation steps, 16 bit-mixed precision training, 500 warmup steps, and 0.01 weight decay. For Gollie, we defined a set of prompts to describe ambiguity detection for each type. The prompts strictly followed our definitions in the annotation guidelines. Figure 7 shows an example of a prompt for ambiguity detection (valid for T3; other cases are similar).

4.1 Results on the Automated Detection of Ambiguities

In this section, we analyze the performance of various models used to detect ambiguities in process descriptions. We evaluate each model based on metrics such as precision, recall, F1-score, and accuracy. Cross-validation results are averaged to ensure robust performance estimates.

Table 2. Detection results, best scores are written in bold

Ambiguity Type	Model	Precision	Recall	F1-Score
T1 = False	Naïve Bayes	0,92	0,57	0,67
	Bert	0,88	0,99	0,94
	Llama2	0,88	0,90	0,89
	Gollie	0,91	0,93	0,92
T1 = True	Naïve Bayes	0,16	0,57	0,23
	Bert	0,04	0,01	0,01
	Llama2	0,55	0,45	0,50
	Gollie	0,60	0,50	0,55
T2 = False	Naïve Bayes	0,94	0,54	0,62
	Bert	0,95	1,00	0,97
	Llama2	0,95	0,96	0,95
	Gollie	0,94	0,95	0,95
T2 = True	Naïve Bayes	0,06	0,45	0,10
	Bert	0,00	0,00	0,00
	Llama2	0,30	0,20	0,24
	Gollie	0,35	0,30	0,33
T3 = False	Naïve Bayes	0,86	0,54	0,62
	Bert	0,85	0,97	0,91
	Llama2	0,90	0,92	0,91
	Gollie	0,89	0,91	0,90
T3 = True	Naïve Bayes	0,13	0,45	0,19
	Bert	0,38	0,03	0,06
	Llama2	0,60	0,50	0,55
	Gollie	0,62	0,55	0,59
T4 = False	Naïve Bayes	0,99	0,37	0,49
	Bert	0,99	1,00	1,00
	Llama2	0,98	0,99	0,98
	Gollie	0,99	0,98	0,99
T4 = True	Naïve Bayes	0,01	0,73	0,02
	Bert	0,00	0,00	0,00
	Llama2	0,10	0,12	0,11
	Gollie	0,15	0,20	0,18

Table 2 reports our results. We notice several aspects. As anticipated, we can observe an overall better performance in the identification of sentences without ambiguities. We believe these results are influenced by the imbalance in the dataset, where more non-ambiguous sentences have been identified. For instance,

Llama2’s accuracy for the classification of negative and positive T1 ambiguities is 0,88 and 0,55, respectively; however, when reviewing the macro-average for the same classes, its value drops to 0,715, illustrating an imbalance between negative and positive labels. Second, despite their overall lower scores in ambiguity detection, some ambiguity types (e.g., T1, T3) have F1-scores above 0,50 and roughly 3x better performance than the Naive Bayesian classifiers, suggesting that the context used for the training of the LLMs improves their ambiguity identification capabilities. Third, our results suggest that the performance of LLMs is pretty similar for the task, with a slight advantage to the Guideline Learning approach from Gollie.

5 Related Work

Ambiguity is a well-documented challenge across various fields, particularly in Requirements Engineering (RE) and Business Process Management (BPM). Numerous studies have explored different types of ambiguities in various artifacts, such as process descriptions, contracts, software requirement specifications, and user stories (see [33] for a recent review). Traditional methods for detecting ambiguity include rule-based approaches and classical machine-learning techniques. Rule-based methods rely on predefined linguistic rules to identify ambiguous expressions. While simple to implement, they often lack scalability and adaptability. Classical machine-learning models, such as Naive Bayes, employ statistical techniques for ambiguity detection but require extensive feature engineering and struggle with complex or context-dependent ambiguities. More advanced approaches leverage transformer models like BERT [31] and GPT-3, which are pre-trained and fine-tuned on large datasets for ambiguity detection. These models, particularly when applied to requirement specifications, effectively capture contextual nuances, significantly enhancing detection accuracy. However, they demand substantial computational resources and extensively annotated datasets [3]. Hybrid techniques combine rule-based methods with machine learning to balance robustness and adaptability, offering improved performance in identifying ambiguous requirements [26]. However, maintaining rule-based components can become increasingly cumbersome as the rule set expands. Additionally, crowd-sourcing approaches, such as CrowdTruth [17], employ consensus-based methods to refine ambiguity detection. While these methods benefit from collective input, their effectiveness depends on well-defined annotation guidelines and the quality of annotators, which can impact consistency and output reliability. In the context of BPM, the closest related work is by van der Aa et al., which addresses ambiguity detection using behavioral spaces [2]. While some ambiguity types overlap, such as *type ambiguity* and *scope ambiguity*, which correspond to ambiguity types T3 and T2 in our taxonomy, this study focuses on a subset of ambiguity types and relies on parsing predefined indicator sets. This reliance limits its generalizability to other structures. In contrast, our approach analyzes ambiguity at the sentence level. This aspect, combined with the context coming from language models, is intended to aid in the generalization of our method.

The work by Soares Silva et al. [29] explores a dataset of 68 process descriptions and identifies a set of linguistic templates for imperative process models. We see these works as complementary in the sense that their linguistic analysis can be complemented with the generalization of sentence-level analysis, while it would be interesting to use similar sentence-pattern analysis to provide user feedback.

6 Reproducibility

We adhere to the open science principles. Our Zenodo repository [9] includes the annotated dataset and our annotation guidelines. Our GitHub repository³ contains the source code used in training and evaluation tasks.

7 Implications of the Results

Our results raise important questions on the limitations of the state of the art in tackling ambiguity as an inherent social problem. Most process descriptions are authored by humans, and ambiguities will be inherently present. Text-to-Process pipelines have so far overseen this, with potential risks in biased outputs. We can see three alternatives to deal with ambiguity in process descriptions:

1. **Controlled Natural Language:** Controlled Natural Language (CNL) alternatives such as FRET [15] or Gherkin [11] provide languages where requirements are described in a restricted natural-language grammar, rendering ambiguity detection simpler. This approach is challenging to scale beyond the computer science field, and it is also difficult to adopt for process owners with diverse backgrounds, as well as to apply to existing documentation.
2. **Conversational Interfaces:** As described in the introduction, most Text-to-Process pipelines do not consider a feedback loop. We suggest that to provide more accurate results, T2P should include ambiguity detection as an important task, next to NER and relation detection. As we mentioned earlier, some ambiguity types are introduced by design, and automated resolution could potentially bias interpretations. Thus, we suggest that T3 and T4 ambiguities are detected and provided to the user for resolution. Conversely, T2 ambiguities could benefit from existing NLP tasks such as co-reference resolution. For T1 ambiguities, we suggest exploring the detection and visualization of ambiguities, possibly following suit of the behavioural spaces approach in [2] and assistive approaches [22].
3. **Beyond accuracy evaluations:** Finally, our study suggests potential shortcomings of accuracy metrics when dealing with T2P pipelines. Precision, Recall, and F1 scores may only work in the case of non-ambiguous descriptions. For ambiguous descriptions, we suggest that authors disclose their ambiguity resolution procedure, which could involve crowd-sourcing, inter-annotator agreements, or the consideration of multiple reference models.

³ <https://github.com/hugualopez-dtu/ambiguityDetector>.

8 Discussion and Conclusion

This paper studied ambiguity in business process specifications, highlighting its prevalence in realistic datasets, where 67 out of 71 (94,3%) process descriptions included at least one ambiguity. To our knowledge, this is the first systematic study of ambiguities in process descriptions. Given the emergence of text-to-process pipelines in BPM, we consider this an important aspect to highlight, which can improve the quality of the models generated once ambiguities are detected. Moreover, our study showed that, while prevalent in virtually every process description, the number of ambiguities is low, and creating balanced datasets will still be an important future task if machine learning classifiers are to be used. Addressing data imbalance through augmentation is crucial, particularly for underrepresented ambiguity types. This paper showcased some promising advances in the application of LLMs to the ambiguity identification task, and we would expect further improvements with larger models.

As part of our threats to validity, we can consider possible errors due to manipulation of the sources and annotator bias. Our preprocessing steps involved slight manipulation of the source texts (e.g., adding sentence terminators when a new line was used). While most of the alterations were clear from the context, they may potentially change the result. Another threat may come from the annotation process, where one team member performed the annotations and a second member revised the annotated dataset. This process risks an anchoring bias. We mitigated the impact via the generation of an annotation guideline and the discussion of each unclear annotation, leading to multiple annotation revisions until an agreement was achieved. Future work will consider inter-annotator agreements and the extension of our study to larger process descriptions, such as Service Operating Procedures [6] and Clinical Guidelines [16]. Finally, we would like to provide implementation support for ambiguity detection.

Acknowledgments. This work was supported by the research grant “Center for Digital Compliance (DICE)” (VIL57420) from VILLUM FONDEN, and by the Innovation Foundation project “Explainable Hybrid-AI for Computational Law and Accurate Legal Chatbots” 4355-00018B XHAILe. This work has received funding from the Swiss National Science Foundation under Grant No. IZSTZ0_208497 (*ProAmbitIon* project).

Disclosure of Interests. The authors have no competing interests to declare relevant to this article’s content.

References

1. van der Aa, H., Di Ciccio, C., Leopold, H., Reijers, H.A.: Extracting declarative process models from natural language. In: Giorgini, P., Weber, B. (eds.) CAiSE 2019. LNCS, vol. 11483, pp. 365–382. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21290-2_23
2. van der Aa, H., Leopold, H., Reijers, H.A.: Dealing with behavioral ambiguity in textual process descriptions. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM

2016. LNCS, vol. 9850, pp. 271–288. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_16
3. Alharbi, S.: Ambiguity detection in requirements classification task using fine-tuned transformation technique. In: CS & IT Conference Proceedings, vol. 12. CS & IT Conference Proceedings (2022)
4. Andaloussi, A.A., Burattin, A., Slaats, T., Kindler, E., Weber, B.: On the declarative paradigm in hybrid business process representations: a conceptual framework and a systematic literature study. *Inf. Syst.* **91**, 101505 (2020)
5. Bellan, P., van der Aa, H., Dragoni, M., Ghidini, C., Ponzetto, S.P.: Pet: an annotated dataset for process extraction from natural language text tasks. In: Cabanillas, C., Garmann-Johnsen, N.F., Koschmider, A. (eds.) BPM 2022. LNBIP, vol. 460, pp. 315–321. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-25383-6_23
6. Bellan, P., Dragoni, M., Ghidini, C.: A qualitative analysis of the state of the art in process extraction from text. In: Proceedings of the AIXIA 2020 Discussion Papers Workshop, vol. 2776, pp. 19–30. CEUR-WS. org (2020)
7. Bellan, P., Dragoni, M., Ghidini, C.: Extracting business process entities and relations from text using pre-trained language models and in-context learning. In: Almeida, J.P.A., Karastoyanova, D., Guizzardi, G., Montali, M., Maggi, F.M., Fonseca, C.M. (eds.) EDOC 2022. LNCS, vol. 13585, pp. 182–199. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-17604-3_11
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. [arXiv:1810.04805](https://arxiv.org/abs/1810.04805) [cs] (2019). <https://arxiv.org/abs/1810.04805>, [arXiv: 1810.04805](https://arxiv.org/abs/1810.04805)
9. Feng, B., Lindner, J., Abbad-Andaloussi, A., Franceschetti, M., López, H.A.: Ambiguity detection in business process descriptions - annotated dataset (2025). <https://doi.org/10.5281/zenodo.15586881>
10. Ferreira, R.C.B., Thom, L.H., Fantinato, M.: A semi-automatic approach to identify business process elements in natural language texts. In: International Conference on Enterprise Information Systems, vol. 2, pp. 250–261. SciTePress (2017)
11. Fischer, T., Dghyam, D.: Formal model validation through acceptance tests. In: Collart-Dutilleul, S., Lecomte, T., Romanovsky, A. (eds.) RSSRail 2019. LNCS, vol. 11495, pp. 159–169. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-18744-6_10
12. Franceschetti, M., Abbad-Andaloussi, A., Schreiber, C., López, H.A., Weber, B.: Exploring the cognitive effects of ambiguity in process models. In: Marrella, A., Resinas, M., Jans, M., Rosemann, M. (eds.) BPM 2024. LNCS, vol. 14940, pp. 493–510. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-70396-6_28
13. Franceschetti, M., Seiger, R., López, H.A., Burattin, A., García-Bañuelos, L., Weber, B.: A characterisation of ambiguity in BPM. In: Almeida, J.P.A., Borbinha, J., Guizzardi, G., Link, S., Zdravkovic, J. (eds.) ER 2023. LNCS, vol. 14320, pp. 277–295. Springer, Cham (2023)
14. Friedrich, F.: Automated generation of business process models from natural language input. M. Sc., School of Business and Economics. Humboldt-Universität zu Berlin (2010)
15. Giannakopoulou, D., Mavridou, A., Rhein, J., Pressburger, T., Schumann, J., Shi, N.: Formal requirements elicitation with fret. In: International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ-2020) (2020)

16. Grathwol, D., van der Aa, H., López, H.A.: Automating pathway extraction from clinical guidelines: a conceptual model, datasets and initial experiments. In: Comuzzi, M., Grigori, D., Sellami, M., Zhou, Z. (eds.) CoopIS 2024. LNCS, vol. 15506, pp. 296–312. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-81375-7_17
17. Inel, O., et al.: CrowdTruth: machine-human computation framework for harnessing disagreement in gathering annotated data. In: Mika, P., et al. (eds.) ISWC 2014. LNCS, vol. 8797, pp. 486–504. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11915-1_31
18. Klie, J.C., Bugert, M., Boullosa, B., de Castilho, R.E., Gurevych, I.: The inception platform: machine-assisted and knowledge-oriented interactive annotation. In: Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations, pp. 5–9. Association for Computational Linguistics (2018). <http://tubiblio.ulb.tu-darmstadt.de/106270/>, event Title: The 27th International Conference on Computational Linguistics (COLING 2018)
19. Kourani, H., Berti, A., Schuster, D., van der Aalst, W.M.: Process modeling with large language models. In: van der Aa, H., Bork, D., Schmidt, R., Sturm, A. (eds.) BPMDS EMMSAD 2024 2024. LNBIP, vol. 511, pp. 229–244. Springer, Cham (2024)
20. Leopold, H., Smirnov, S., Mendling, J.: On labeling quality in business process models. In: 8. Workshop der Gesellschaft für Informatik eV (GI) und Treffen ihres Arbeitskreises „Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)“, Nüttgens, M. et al.(eds.), Berlin, pp. 42–57. Citeseer (2009)
21. López, H.A.: Challenges in legal process discovery. In: ITBPM@ BPM, pp. 68–73 (2021)
22. López, H.A., Marquard, M., Muttenthaler, L., Strømsted, R.: Assisted declarative process creation from natural language descriptions. In: 2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW), pp. 96–99. IEEE (2019)
23. López, H.A., Strømsted, R., Niyodusenga, J.-M., Marquard, M.: Declarative process discovery: linking process and textual views. In: Nurcan, S., Korthaus, A. (eds.) CAiSE 2021. LNBIP, vol. 424, pp. 109–117. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-79108-7_13
24. Lu, C.: Examining scientific writing styles from the perspective of linguistic complexity. *J. Am. Soc. Inf. Sci.* **70**(5), 462–475 (2019)
25. Nour Eldin, A., Assy, N., Anesini, O., Dalmas, B., Gaaloul, W.: Nala2BPMN: automating BPMN model generation with large language models. In: Comuzzi, M., Grigori, D., Sellami, M., Zhou, Z. (eds.) CoopIS 2024. LNCS, vol. 15506, pp. 398–404. Springer, Cham (2024)
26. Osman, M., Zahrin, M.: Ambi detect: an ambiguous software requirements specification detection tool. *Turk. J. Comput. Math. Educ. (TURCOMAT)* **12**(3), 2023–2028 (2021)
27. Sainz, O., García-Ferrero, I., Agerri, R., de Lacalle, O.L., Rigau, G., Agirre, E.: Golli: Annotation guidelines improve zero-shot information-extraction. *arXiv preprint arXiv:2310.03668* (2023)
28. Shah, U.S., Jinwala, D.C.: Resolving ambiguities in natural language software requirements: a comprehensive survey. *ACM SIGSOFT Softw. Eng. Notes* **40**(5), 1–7 (2015)
29. Silva, T.S., Thom, L.H., Weber, A., de Oliveira, J.P.M., Fantinato, M.: Empirical analysis of sentence templates and ambiguity issues for business process descriptions. In: Panetto, H., Debruyne, C., Proper, H.A., Ardagna, C.A., Roman, D.,

- Meersman, R. (eds.) OTM 2018. LNCS, vol. 11229, pp. 279–297. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02610-3_16
30. Touvron, H., et al.: Llama 2: open foundation and fine-tuned chat models. arXiv preprint [arXiv:2307.09288](https://arxiv.org/abs/2307.09288) (2023)
31. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
32. Weske, M., Decker, G., Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A.: Model collection of the business process management academic initiative (2020). <https://doi.org/10.5281/zenodo.3758705>
33. Zhao, L., et al.: Natural language processing for requirements engineering: a systematic mapping study. ACM Comput. Surv. (CSUR) **54**(3), 1–41 (2021)