GUIDO: A Hybrid Approach to Guideline Discovery & Ordering from Natural Language Texts

Nils Freyer[a], Dustin Thewes[b] and Matthias Meinecke[c]
FB7 Operations Management, FH Aachen University of Applied Sciences, Aachen, Germany

Keywords: Natural Language Processing, Text Mining, Process Model Extraction, Business Process Intelligence.

Abstract: Extracting workflow nets from textual descriptions can be used to simplify guidelines or formalize textual descriptions of formal processes like business processes and algorithms. The task of manually extracting processes, however, requires domain expertise and effort. While automatic process model extraction is desirable, annotating texts with formalized process models is expensive. Therefore, there are only a few machine-learning-based extraction approaches. Rule-based approaches, in turn, require domain specificity to work well and can rarely distinguish relevant and irrelevant information in textual descriptions. In this paper, we present GUIDO, a hybrid approach to the process model extraction task that first, classifies sentences regarding their relevance to the process model, using a BERT-based sentence classifier, and second, extracts a process model from the sentences classified as relevant, using dependency parsing. The presented approach achieves significantly better results than a pure rule-based approach. GUIDO achieves an average behavioral similarity score of 0.93. Still, in comparison to purely machine-learning-based approaches, the annotation costs stay low.

1 INTRODUCTION

To fulfill a task or execute a process in a predetermined way, especially when lacking the respective expertise, one often needs to follow *guidelines*. Guidelines are commonly given as unstructured texts. Examples from their domain space are business processes, technical standards, cooking recipes, medical guidelines explaining the standard procedures to medical professionals, or the description of algorithms. Understanding, updating, and conformance-checking a guideline requires sufficient proficiency in the language, adequate reading comprehension, and often adequate domain expertise (e.g., a medical degree).

In contrast to unstructured texts, process models may be described using formalized process modeling. Process models encode order, decision rules, and loops in the notation, only requiring labeling of the activities, constraints, and decision rules as texts (Mendling et al., 2014). However, transforming unstructured text into structured process models requires expertise in process modeling and thus, yields an expensive task (Friedrich et al., 2011; Frederiks and van der Weide, 2006).

[a] https://orcid.org/0000-0002-4460-3650
[b] https://orcid.org/0000-0002-1301-8926
[c] https://orcid.org/0009-0008-3055-5505

The assisted extraction of formalized process models from text is an active field of research and could alleviate those problems (López et al., 2019). Contemporary approaches are either pure rule-based, usually specific to a domain, or purely machine-learning-based, requiring large amounts of annotated data for a specific domain and language. As extracting process models manually is time-consuming and expensive, using pure machine-learning-based approaches is either restricted to domains with a sufficient amount of annotated data or requires large corpora to be annotated, making it inapplicable for smaller extraction domains.

We propose GUIDO, a Guideline Discovery & Ordering approach that extracts process models from natural language text (cf. section 4). GUIDO first uses a BERT sequence classifier to identify and filter sentences relevant to the process. Second, it uses a language rule-based model to extract the processes' activities, interactivity relations, and temporal order. Finally, GUIDO uses the extracted relations to formalize the process model as a workflow net. We demonstrate the proposed approach with German recipes, achieving an F1-score of 0.973 for sentence classification and an average behavioral similarity score between generated process models and human-expert-made process models of 0.93 (cf. sec-

Freyer, N., Thewes, D. and Meinecke, M.
GUIDO: A Hybrid Approach to Guideline Discovery Ordering from Natural Language Texts.
DOI: 10.5220/0012084400003541
In *Proceedings of the 12th International Conference on Data Science, Technology and Applications (DATA 2023)*, pages 335-342
ISBN: 978-989-758-664-4; ISSN: 2184-285X

tion 5). The code and data for this project are publicly available at https://github.com/nils-freyer/GUIDO

2 ETHICAL CONSIDERATIONS

While this paper investigates extracting process models on German recipes, the approach applies to a more extensive section of the domain space, including more safety- and security-relevant domains. The approach introduced in this paper merely offers assistance in extracting process models. Both the rule-based component and the machine-learning-based component of the approach may not generalize to use cases outside the evaluation scenario. Furthermore, pre-trained BERT models will introduce biases to the text classification (Liang et al., 2021). Depending on the application domain, discriminatory outcomes should be examined carefully.

3 RELATED WORK

Process Model Extraction (PME) is considered a Text to Model challenge, including identifying activities and their sequence or concurrency (Mendling et al., 2014). PME approaches can be categorized broadly as *rule-based*, *machine-learning-based*, or *hybrid*, combining rule and machine-learning-based approaches.

Rule-Based Approaches. Rule-based approaches mainly use grammatical features of a text and are applied to both extracting declarative (Aa et al., 2019; Winter and Rinderle-Ma, 2018) and imperative (Zhang et al., 2012; Walter et al., 2011; Schumacher et al., 2012) process models. Although they perform domain-specifically well, restrictions have to be made to identify activities as, e.g., verb centrality (Walter et al., 2011; Qian et al., 2020) or constraint markers (Aa et al., 2019; Winter and Rinderle-Ma, 2018, 2019) requiring domain-specific knowledge on potential heuristics.

Machine-Learning-Based Approaches. Machine-learning-based approaches such as conditional random fields, support vector machines, and neural text classification was used for the detection of the process relevant sentences (Leopold et al., 2018; Qian et al., 2020). Furthermore, Qian et al (Qian et al., 2020) identified process model extra as a multi-grained text classification task. They developed a hierarchical neural network to classify relevant sentences and generate the extracted process model. While the results are promising, a multi-grained, annotated dataset is needed. Additionally, to the related task of extracting linear temporal logic from natural language texts, a neural machine translation approach was proposed (Brunello et al., 2019).

Hybrid Approaches. Little work has combined rule-based and machine-learning-based PME approaches. Relatedly, Winter and Rinderle-Ma (2019) used constraint markers as *shall, must, should*, to identify sentences containing declarative process information and used sentence embeddings and clusterings to find related constraints. However, these examples do not implement hybrid approaches for the extraction of process models.

To the best of our knowledge, there were no implementations and evaluation on German texts yet. Especially rule-based approaches will differ language-wise. Furthermore, GUIDO is the first hybrid PME approach, using generally known approaches in a novel hybrid way in order to reduce labeling costs and maximize generalizability and accuracy.

4 GUIDO AS A MULTI-LEVEL EXTRACTION MODEL

As described by Qian et al. (2020), the PME task can be formulated as a hierarchical information extraction task. That is, we can subdivide the task into sentence classification, activity extraction and activity ordering. This section introduces basic preliminaries, notations and outlines the proposed solutions to each of the sub-tasks.

4.1 Preliminaries

Within our research, we chose to use Petri nets (Chen and Marwedel, 1990) and more specifically workflow nets (Van der Aalst, 1998) to formalize imperative process models.

Definition (Workflow Net)**.** A Petri net is a tuple $N = (P, T, F)$, where P is a set of places, T is a set of transitions, $P \cap T = \emptyset$, and $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation of the network.

A workflow net is a Petri net $\mathcal{W} = (P, T, F)$, such that there is a unique source and a unique sink to all paths in the net. Especially in our domain, workflow nets, as a subclass of Petri nets, are a reasonable choice, as any recipe has a dedicated set of end states and thus, can be converted to a workflow net. The transitions of the Petri net describe the activities of the process. An activity is typically constituted by the act (verb), its subjects and objects, as well as its modifiers.

Definition (Activity)**.** Given a vocabulary V, an activity is a tuple $a = (v, s, o, m) \in \mathcal{P}(V)^4$, where v is a set of verbs, s is a set of subjects, o is a set of objects and m is a set of modifiers declaring the activity. Given a text $\mathcal{T} = (S_1, \dots, S_n)$ with sentences $S_1, \dots, S_n \in V^m$, $m \in \mathfrak{N}$, $\mathcal{A}(\mathcal{T})$ denotes the set of activities in $\mathcal{T}$ and consequently $\mathcal{A}(S)$ denotes the set of activities in a given sentence S.

For instance (”foam”,”butter”, $\emptyset$,”in a hot pan”), is the activity we want to extract from the sentence ”Foam butter in a hot pan”. Therefore, if we want to extract a workflow net $\mathcal{W}$ from a text $\mathcal{T}$ we derive the following extraction task.

Definition (Process Model Extraction Task)**.** Given a text $\mathcal{T} = (S_1, \dots, S_n)$, extract a workflow net $N = (P, T, F)$, s.t. $T = \mathcal{A}(\mathcal{T})$ and F spans the temporal relation of $\mathcal{A}(\mathcal{T})$ in $\mathcal{T}$.

4.2 Model Architecture

Understanding PME task as a hierarchical information extraction task, first, we need to classify whether a particular sentence S of a text $\mathcal{T}$ contains an activity $a \in \mathcal{A}(\mathcal{T})$. Second, we need to extract all $a_1, \dots, a_k$ in $\mathcal{A}(S)$. Finally, we need to extract the temporal order T of $\mathcal{A}(\mathcal{T})$ (cf. Figure 1), in order to derive the flow relation F of the workflow net.

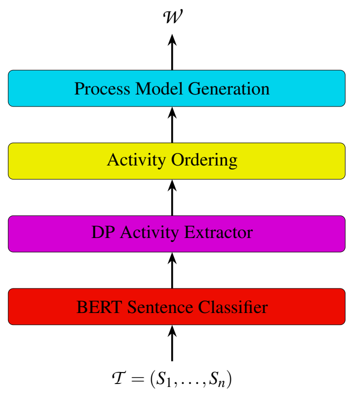


Figure 1: Hierarchical model architecture.

Each sub-task was implemented and evaluated separately in addition to the total evaluation of the extracted workflow nets. Therefore, they can be used independently to create baselines for the hybrid approach.

4.3 Sentence Classifiers

The sentence classification level of GUIDO has to perform the binary classification task $\mathcal{A}(S) = \emptyset$, given a sentence S in a text $\mathcal{T}$, i.e., whether a sentence contains an activity or not. We implemented and tested three different classification strategies and compared them to a rule-based baseline strategy.

VVIMP Rule-Based Baseline. As a rule-based approach, we implemented a heuristic that classifies a sentence as process relevant if there is no subject that is not a child of an imperative in the dependency tree.

LSTM Classifier. A simple LSTM (Hochreiter and Schmidhuber, 1997) with a text-classification head was implemented and fully configured by hydra configurations. The LSTM was optimized by a hyper parameter search with 5 workers. The documents were vectorized using either pre-trained and fine tuned GloVe[1] vectors or pre-trained FastTexts[2] vectors.

Logistic Regression. A binary logistic regression classifier was implemented using tfidf document vectorization.

BERT Sequence Classifier. The huggingface's BERT (Devlin et al., 2019) for sequence classification was used[3], using a linear layer for classification on the pooled output of the BERT model. The pre-trained German BERT transformer model (Chan et al., 2020) was used to initialize the model. The German BERT model was chosen over the multilingual pre-trained BERT, as it has shown superior performance on common evaluation sets (Chan et al., 2020).

4.4 Activity Extraction by Dependency Grammar

The next level of GUIDO performs the task of activity extraction. Given a sentence S with $\mathcal{A}(S) \neq \emptyset$, we want to extract all activity relations $a_1, \dots, a_n \in \mathcal{A}(S)$. Machine-learning-based relation extraction models require complexly annotated corpora. Therefore, to reduce annotation costs, we chose to implement a rule-based relation extraction approach, us-

[1] Pre-trained glove vectors taken from: https://www.deepset.ai/german-word-embeddings

[2] Pre-trained FastText vectors taken from: https://fasttext.cc/docs/en/crawl-vectors.html

[3] https://huggingface.co/docs/transformers/v4.26.0/en/model_doc/bert#transformers.TFBertForSequenceClassification

ing dependency grammar (Nivre, 2005). Dependency grammar is a school of grammar that describes the hierarchical structure of sentences based on dependencies between words within a sentence. NLP frameworks such as spaCy have incorporated dependency parsers into their pipelines (Honnibal et al., 2020), trained on large news corpora. Thus, using dependency parsers, POS tags, and STTS tags (Albert et al., 2003), does not require further manual labeling. Dependency grammar-based approaches were proposed to be used for the extraction of process activities from text (Sintoris and Vergidis, 2017; Kolb et al., 2013; Zhang et al., 2012) as well as for similar tasks such as the translation of sentences to linear temporal logic (Brunello et al., 2019) or the extraction of declarative process constraints from natural language texts (Winter and Rinderle-Ma, 2018; Aa et al., 2019). A major pitfall of using a dependency grammar for activity extraction are non-relevant sentences and subordinate clauses. Therefore, it was primarily applied to documents with strict language norms, e.g., laws, where rule-based classifiers, taking markers as *must* or *should* as indicators of a relevant sentence, work particularly well. As we use a sentence classifier to avoid irrelevant sentences, handling subordinate clauses remains on the activity extraction level of the PME task.

Extraction Rules. By assumption, we extract activities from relevant sentences only. Therefore, activities are expressed as verbs with dependent subjects, objects, and modifiers. In rare cases, activities may be expressed as passivized subjects(Aa et al., 2019).

Figure 2 shows the exemplary dependency tree of a sentence $S =$ *"Butter in einer heißen Pfanne aufschäumen lassen."* (Engl.: *"Foam butter in a hot pan."*) of a text $\mathcal{T} = (S)$. By traversing the dependency graph for all verbs in S we obtain the activity set $\mathcal{A}(\mathcal{T}) = \{(v,s,o,m)\}$ with:

- $v = \{\text{aufschäumen, lassen}\}$
- $s = \emptyset$
- $o = \{\text{Butter}\}$
- $m = \{\text{in einer heißen Pfanne}\}$

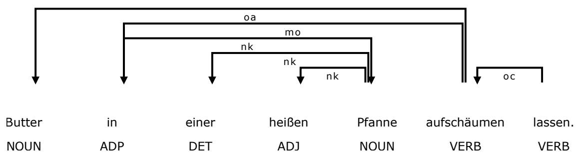


Figure 2: Dependency tree of a German recipe sentence.

Negations. The negation of an activity constitutes a special case. Figure 3 illustrates the dependency tree of S with negotiation. The dependency parser tags negation dependencies as *ng* and thus, allows us to extract negations (Aa et al., 2019; Albert et al., 2003). We omit negations in our extraction approach. However, negations could easily be added to the activity if needed.

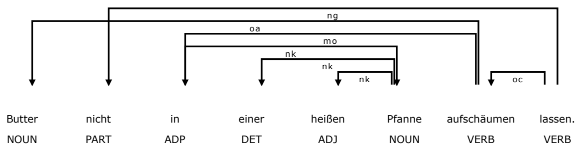


Figure 3: Dependency tree of a negated German recipe sentence.

Quantification. Not every activity described in a text is mandatory. While constraint-markers, as declared by (Aa et al., 2019; Winter and Rinderle-Ma, 2019, 2018), do not suffice for the generic classification of sentences containing process information, they indicate, if present, whether there *exists* a path in the supposed workflow net $\mathcal{W}$ of a text $\mathcal{T}$ containing a related activity a or if *all* paths of the workflow net contain a. We used GermaNet (Hamp and Feldweg, 1997) to obtain a more complete list of constraint markers as given in Table 1. By default, if not further specified, we assume an activity to be mandatory.

Table 1: Quantifying constraint markers.

Marker	Word
EXISTS	können, dürfen, mögen, sollten, kann, vielleicht, optional, eventuell, gegebenenfalls
ALL	müssen

Irrelevant Subordinate Clauses. Although we may assume to extract activities from relevant sentences only, we may not assume every sentence's verb to be relevant. For instance, the sentence $S =$ *Butter in einer heißen Pfanne aufschäumen lassen, das schmeckt mir am besten* contains the relation $a_1 = a$ as in Figure 2. However, simply extracting all verbs and their dependents would also yield $a_2 = (\{\text{schmeckt}\}, \{\text{das}\}, \emptyset, \{\text{am besten}\})$. A simple heuristic to handle such clauses is to use the VVIMP tag from (Albert et al., 2003) as incorporated into the spaCy framework. However, as recipes are not formalized, some are written in a descriptive form or a first-person narrative. Therefore, such recipes would not be handled well. A second heuristic may be the recognition of a switch in writing style. If a sentence contains an imperative and a non-imperative verb, we may assume the imperative verb to be an activity and the non-imperative to be descriptive. The effect of the heuristic is examined in section 6.

4.5 Activity Ordering: Interactivity Relation Extraction

By default, we implicitly assume the described activities in the process model to be ordered as their appearance in the text orders them. However, interactivity relations explicitly describe the activity ordering and can be classified as *AND*, *OR*, or *BEFORE* relations. To obtain the order in which the activities described in the text should be executed, we need to be able to extract these interactivity relations. In the simpler case, these relations are expressed within a sentence. Coordinating conjuncts in combination with synonym databases such as WordNet (Miller, 1998) or the German GermaNet (Hamp and Feldweg, 1997) as tagged by the dependency parser can be used to identify conjunctions and disjunctions of activities to extract *AND* or *OR* relations. Temporal adverbs can be identified using WordNet/GermaNet as well (cf. (Aa et al., 2019)). *BEFORE* relations that are described across sentences can be handled using coreference resolution to identify the referenced activities from previous sentences, or using simple heuristics. For instance, it is reasonable to assume that a temporal adverb as *inzwischen* (Engl. meanwhile) indicates an *AND* relation to the activities of the previous sentence. In sum, we identified the following heuristics:

- coordinating conjuncts within sentences
- temporal adverbs within sentences (if not dependent on the first activity):
  1. if indicating *AND* relation: add *AND* relation to previous activity
  2. if indicating *BEFORE* relation: add *BEFORE* relation to activities in the previous sentence
- temporal adverbs across sentences (if dependent on the first activity in the sentence):
  1. if indicating *AND* relation: add *AND* relation to activities of previous sentence
  2. if indicating *BEFORE* relation and only one activity within sentence: add *BEFORE* relation to activities in the previous sentence

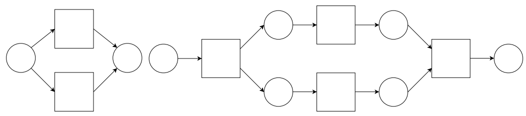
The indicator synonyms are given in Table 2

4.6 Generating Process Models

From the previous steps, we obtain a set of activities and a set of binary relations between activities. The remaining task is the creation of a workflow net. To do so, we first, create a workflow net for each sentence by applying patterns (cf. Figure 4) for OR, AND, and BEFORE relations extracted as described in subsection 4.4.

Table 2: Temporal Adverbs for the extraction of interactivity relations.

Adverb	Adverb
BEFORE	zuvor, davor, vorab, vordem, vorher, vorweg, zuerst, zunächst, anfänglich, anfangs, eingangs, erst, vorerst
AND	inzwischen, dabei, währenddessen, dazwischen, inzwischen, mittlerweile, solange, zwischenzeitlich, derweil, einstweilen



(a) (b)

Figure 4: (a) OR pattern (b) AND pattern.

Then, the sub nets are merged to the final workflow net $\mathcal{W}$ of the recipe $\mathcal{T}$ by either appending the sub net to the previous sub net or, if the the first activity in a sentence indicates a parallelization, the sub net is added using AND pattern as a parallel to the previous sub net (cf. Algorithm 1).

Algorithm 1: Workflow net generation.

function GENERATEWORKFLOWNET($\mathcal{T}$)
    $pn := NewPetriNet()$
    $last\_sn := pn$
    $N := len(\mathcal{T})$
    for $i \in \{1, \ldots, N\}$ **do**
        $sn := get\_sub\_net(\mathcal{A}(S_i))$
        if $parrallel(S_i)$ **then**
            $pn.add\_parallel(last\_sn, sn)$
        else
            $pn.append(sn)$
        end if
        $last\_sn := sn$
    end for
end function

5 EXPERIMENTS

Rule-based and machine-learning-based approaches to PME formulate a trade-off. While rule-based approaches require the adoption of rules to suit domain-specific formulations and conventions, machine-learning-based approaches require large corpora of complexly annotated data. Thus, as formulated by e.g. (Qian et al., 2020), we may divide PME into different tasks to be solved either machine-learning-based or rule-based.

5.1 Data & Data Preparation

Recipes from the German recipe website Chefkoch[4] were used to train the sentence classifiers and evaluate GUIDO. The dataset contains 44672 unique sentences from 4291 recipes, from which we sub-sampled and annotated 2030 recipes for binary classification and 50 mutually exclusive recipes for workflow net annotation, to compare the extracted process model to.

For the sake of training the BERT text classifier, we identified and replaced URLs by a unique $\$URL$ token, using regular expressions. The rule-based PME levels do not require further text normalization.

Table 3: Sentence corpus statistics where S denotes Sentences after balancing by down-sampling.

Set	# S	% S	# Relevant	% Relevant
Train	1533	60%	773	50.42%
Dev	512	20%	240	46.86%
Test	511	20%	265	51.75%

Sentence & Workflow Net Labeling. A sentence dataset was build using the spaCy dependency-parser-based sentence tokenizer (Honnibal et al., 2020). Two annotators labeled the sentences. To increase the process quality of the labeling process and increase the quality of the labeled dataset, labeling guidelines were written before labeling[5]. If there was uncertainty in assigning a label in a given sentence, the annotator discussed the label with the other annotator and updated the labeling guidelines with the result of the discussion. Subsequently to the sentence annotation process, the sentences were further sub-sampled to obtain a balanced dataset of 3150 annotated sentences, as irrelevant sentence make about 10% of the sentence population only. The sub-sampled sentence corpus was split into *train*, *test* and *dev* sets for training and evaluation. The statistics of the annotated sentence corpus are given in Table 3. A set of 50 recipes with 616 sentences in total was annotated with corresponding workflow nets by a single annotator.

5.2 Evaluation

To evaluate the performance GUIDO, the text classification and the PME task are evaluated separately. The text classification task was evaluated according to its *F1-Score* on a validation set of size $N = 512$. A total of 50 recipes were annotated manually using ProM[6], in order to obtain similarity metrics. As, in the case of PME, we need a metric that compares the behavior of workflow nets rather than the syntactical equivalence of the output to the annotation, we implemented a behavioral similarity score based on causal footprints, an abstract representation of a Petri net's behavior. (Mendling et al., 2007). We applied the similarity metric to a rule-based baseline model, GUIDO with heuristics to handle subordinate clauses and GUIDO without additional heuristics. All experiments were done on using a single machine with an Intel Xeon processor, a NVIDIA GeForce RTX-A5000 GPU with 16 GB of VRAM, and 64 GB of RAM, running on Ubuntu 20.04, which has a estimated carbon efficiency of 0.432 $kgCO_2eq/kWh$. A cumulative of 0.5 hours of computation was performed on hardware of type RTX A5000 (TDP of 230W) for training. A cumulative of 30 hours of computation was performed on hardware of type Intel Xeon W-11855M (TDP of 45W) for evaluating. Total emissions are estimated to be 0.65 $kgCO_2eq$ of which 0 percents were directly offset. Estimations were conducted using the MachineLearning Impact calculator presented in Lacoste et al. (2019).

To conduct our experiments, we fully parameterized the project using a hydra-config[7]. A parallelized grid search was used for parameter tuning. Furthermore, we used the mlflow framework[8] for visualizing training and evaluation metrics.

6 RESULTS

In this section, we first compare the proposed BERT sentence classifier with three baseline models, evaluated on 512 unseen sentences. Then, we evaluate GUIDO on 50 unseen recipes, containing 616 sentences.

Sentence Classification. Multiple approaches were evaluated in addition to the BERT sentence classifier and compared to the VVIMP baseline (cf. Table 4). The simple VVIMP heuristics classifies a sentence as

[4]https://www.kaggle.com/datasets/sterby/german-recipes-dataset

[5]cf. https://github.com/nils-freyer/GUIDO/wiki/Labeling-Guideline

[6]https://promtools.org/

[7]https://hydra.cc

[8]https://mlflow.org

Table 4: Classifier F1-Scores for the base line heuristic (VVIMP), the logistic regression classifier (Log Reg), the LSTM classifier with FastText (LSTM FT), the LSTM classifier with GloVe (LSTM GloVe) and the BERT sentence classifier.

Score	VVIMP	Log Reg	LSTM FT	LSTM GloVe	**BERT**
F1	0.58	0.90	0.91	0.92	**0.973**

process relevant, i.e., containing at least one activity, if there is no subject that is not a child of an imperative in the dependency tree, resulting in an F1-Score of ≈ 0.81. Further, the documents were tfidf-vectorized. A binary logistic regression classifier was trained and obtained an F1-Score of ≈ 0.90. A simple LSTM with a text-classification head obtained an F1-Score of ≈ 0.91 on fine-tuned GloVe vectors and ≈ 0.92 on pre-trained multilingual fasttext vector. Finally, the BERT sentence classifier outperformed the baseline models with a final F1-Score of ≈ 0.973 with batch size 16, 5 epochs and learning rate $3e^{-5}$.

Process Model Extraction. We compared the 50 annotated workflow nets to the extracted workflow nets by GUIDO + VVIMP heuristic, GUIDO - VVIMP heuristic, and to the extracted workflow nets of a purely rule-based approach. The results (cf. Table 5) show significant improvements for the rule-based process extractor when adding the text classification level with an average similarity score of ≈ 0.93 over ≈ 0.84. The usage of a VVIMP heuristic to handle subordinate clauses did not have a significant effect on the performance of GUIDO, as only one verb was classified as an imperative by the tagger.

Table 5: CFP behavioral similarities.

Model	Rule-Based	GUIDO - VVIMP	**GUDIO + VVIMP**
CFP-Sim	0.84	0.93	**0.93**

7 DISCUSSION & FUTURE WORK

The proposed PME model GUIDO shows good performance given a reduced labeling effort of 2030 binary annotated sentences compared to purely machine-learning-based approaches. The additional step of a sentence classifier significantly improves the performance of rule-based PME models compared to purely rule-based PME models and therefore, formulates a compromise to the annotation cost and specificity trade-off. The rule-based level of GUIDO was designed in a generic way, applicable to multiple domains. Additionally, the approach is easily transferable for rule-based Declarative PME tasks (Aa et al., 2019; López et al., 2019).

The most common errors of GUIDO were miss classifications of sentences and irrelevant subordinate clauses. Common taggers perform poorly on process data, as they were mostly trained on news data (Han et al., 2019). In particular, news data rarely contains imperatives and thus has a high miss classification rate for the VVIMP tags we use for handling irrelevant subordinate clauses. A further limitation to the PME task and results is the fine graindness of the desired process model. Throughout this paper, we assumed repetitive activities to be a single activity. For instance *"wiederholt umrühren"* (Engl.: *"stir repeatedly"*) would not result in a cycle in the Petri net but be a single transition. Such cycles should be incorporated and evaluated in future work. GUIDO was trained and applied to German recipes only in this paper, containing imperative process models only. In future work, we will evaluate our approach on declarative guidelines. While there is a lack of comparison for German workflow net extraction, the behavioral similarity scores achieved by GUIDO seem competitive to related work in other languages (Qian et al., 2020). Especially the rule-based level of GUIDO is grammar specific, which is specific to the German language. Thus, we will adopt and evaluate GUIDO on English recipes in the future to get further insights on it performance compared to existing state of the art approaches.

ACKNOWLEDGMENTS

This research has been developed and funded by the project Assist.me (grant number 16KN090726) of the German Federal Ministry of Economic Affairs and Climate Action (Bundesministerium für Wirtschaft und Klimaschutz (BMWK)).

REFERENCES

Aa, H. v. d., Ciccio, C. D., Leopold, H., and Reijers, H. A. (2019). Extracting declarative process models from natural language. In *International Conference on Advanced Information Systems Engineering*, pages 365–382. Springer.

Albert, S., Anderssen, J., Bader, R., Becker, S., Bracht, T., Brants, S., Brants, T., Demberg, V., Dipper, S., Eisenberg, P., et al. (2003). tiger annotationsschema. *Universität des Saarlandes and Universität Stuttgart and Universität Potsdam*, pages 1–148.

Brunello, A., Montanari, A., and Reynolds, M. (2019). Synthesis of ltl formulas from natural language texts: State of the art and research directions. In *26th International symposium on temporal representation and reasoning (TIME 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Chan, B., Möller, T., Pietsch, M., and Soni, T. (2020). Open sourcing german bert model.

Chen, J.-J. and Marwedel, P. (1990). Petri nets. *IEEE Transactions on knowledge and data engineering*, 2(3):311–319.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs].

Frederiks, P. J. M. and van der Weide, T. P. (2006). Information modeling: The process and the required competencies of its participants. *Data & Knowledge Engineering*, 58(1):4–20.

Friedrich, F., Mendling, J., and Puhlmann, F. (2011). Process model generation from natural language text. In King, R., editor, *Active Flow and Combustion Control 2018*, volume 141, pages 482–496. Springer International Publishing. Series Title: Notes on Numerical Fluid Mechanics and Multidisciplinary Design.

Hamp, B. and Feldweg, H. (1997). Germanet-a lexical-semantic net for german. In *Automatic information extraction and building of lexical semantic resources for NLP applications*.

Han, X., Dang, Y., Mei, L., Wang, Y., Li, S., and Zhou, X. (2019). A novel part of speech tagging framework for nlp based business process management. In *2019 IEEE International Conference on Web Services (ICWS)*, pages 383–387. IEEE.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A., et al. (2020). spacy: Industrial-strength natural language processing in python.

Kolb, J., Leopold, H., Mendling, J., and Reichert, M. (2013). Creating and updating personalized and verbalized business process descriptions. In *The Practice of Enterprise Modeling: 6th IFIP WG 8.1 Working Conference, PoEM 2013, Riga, Latvia, November 6-7, 2013, Proceedings 6*, pages 191–205. Springer.

Lacoste, A., Luccioni, A., Schmidt, V., and Dandres, T. (2019). Quantifying the carbon emissions of machine learning. *arXiv preprint arXiv:1910.09700*.

Leopold, H., van Der Aa, H., and Reijers, H. A. (2018). Identifying candidate tasks for robotic process automation in textual process descriptions. In *Enterprise, Business-Process and Information Systems Modeling: 19th International Conference, BPMDS 2018, 23rd International Conference, EMMSAD 2018, Held at CAiSE 2018, Tallinn, Estonia, June 11-12, 2018, Proceedings 19*, pages 67–81. Springer.

Liang, P. P., Wu, C., Morency, L.-P., and Salakhutdinov, R. (2021). Towards understanding and mitigating social biases in language models. In *International Conference on Machine Learning*, pages 6565–6576. PMLR.

López, H. A., Marquard, M., Muttenthaler, L., and Strømsted, R. (2019). Assisted declarative process creation from natural language descriptions. In *2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW)*, pages 96–99. ISSN: 2325-6605.

Mendling, J., Leopold, H., and Pittke, F. (2014). 25 challenges of semantic process modeling. *International Journal of Information Systems and Software Engineering for Big Companies (IJISEBC)*, 1(1):78–94.

Mendling, J., van Dongen, B. F., and van der Aalst, W. M. (2007). On the degree of behavioral similarity between business process models. In *EPK*, volume 303, pages 39–58.

Miller, G. A. (1998). *WordNet: An electronic lexical database*. MIT press.

Nivre, J. (2005). Dependency grammar and dependency parsing. *MSI report*, 5133(1959):1–32.

Qian, C., Wen, L., Kumar, A., Lin, L., Lin, L., Zong, Z., Li, S., and Wang, J. (2020). An approach for process model extraction by multi-grained text classification. In *International Conference on Advanced Information Systems Engineering*, pages 268–282. Springer.

Schumacher, P., Minor, M., Walter, K., and Bergmann, R. (2012). Extraction of procedural knowledge from the web: A comparison of two workflow extraction approaches. In *Proceedings of the 21st International Conference on World Wide Web*, pages 739–747.

Sintoris, K. and Vergidis, K. (2017). Extracting business process models using natural language processing (nlp) techniques. In *2017 IEEE 19th conference on business informatics (CBI)*, volume 1, pages 135–139. IEEE.

Van der Aalst, W. M. (1998). The application of petri nets to workflow management. *Journal of circuits, systems, and computers*, 8(01):21–66.

Walter, K., Minor, M., and Bergmann, R. (2011). Workflow extraction from cooking recipes. In *Proceedings of the ICCBR 2011 Workshops*, pages 207–216.

Winter, K. and Rinderle-Ma, S. (2018). Detecting constraints and their relations from regulatory documents using nlp techniques. In *On the Move to Meaningful Internet Systems. OTM 2018 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2018, Valletta, Malta, October 22-26, 2018, Proceedings, Part I*, pages 261–278. Springer.

Winter, K. and Rinderle-Ma, S. (2019). Deriving and combining mixed graphs from regulatory documents based on constraint relations. In *International Conference on Advanced Information Systems Engineering*, pages 430–445. Springer.

Zhang, Z., Webster, P., Uren, V. S., Varga, A., and Ciravegna, F. (2012). Automatically extracting procedural knowledge from instructional texts using natural language processing. In *LREC*, volume 2012, pages 520–527. Citeseer.