Identifying Candidate Tasks for Robotic Process Automation in Textual Process Descriptions

Henrik Leopold, Han van der Aa(✉), and Hajo A. Reijers

Vrije Universiteit Amsterdam, De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
{h.leopold,j.h.vander.aa,h.a.reijers}@vu.nl

Abstract. The continuous digitization requires organizations to improve the automation of their business processes. Among others, this has lead to an increased interest in Robotic Process Automation (RPA). RPA solutions emerge in the form of software that automatically executes repetitive and routine tasks. While the benefits of RPA on cost savings and other relevant performance indicators have been demonstrated in different contexts, one of the key challenges for RPA endeavors is to effectively identify processes and tasks that are suitable for automation. Textual process descriptions, such as work instructions, provide rich and important insights about this matter. However, organizations often maintain hundreds or even thousands of them, which makes a manual analysis unfeasible for larger organizations. Recognizing the large manual effort required to determine the current degree of automation in an organization's business processes, we use this paper to propose an approach that is able to automatically do so. More specifically, we leverage supervised machine learning to automatically identify whether a task described in a textual process description is manual, an interaction of a human with an information system or automated. An evaluation with a set of 424 activities from a total of 47 textual process descriptions demonstrates that our approach produces satisfactory results.

1 Introduction

Many organizations currently face the challenge of keeping up with the increasing digitization. Among others, it requires them to adapt existing business models and to respectively improve the automation of their business processes [28]. While the former is a rather strategic task, the latter calls for specific operational solutions. One of the most recent developments to increase the level of automation is referred to as Robotic Process Automation (RPA). In essence, RPA emerges in the form of software-based solutions that automatically execute repetitive and routine tasks [5]. In this way, knowledge workers can dedicate their time and effort to more complex and value adding tasks.

While the benefits of RPA have been demonstrated in different contexts [7,20], one of the key challenges is to effectively identify processes and tasks that

J. Gulden et al. (Eds.): BPMDS 2018/EMMSAD 2018, LNBIP 318, pp. 67–81, 2018.
https://doi.org/10.1007/978-3-319-91704-7_5

are suitable for automation [5]. So far, research has focused on the establishment of criteria [11,37] and step-by-step guidelines [9] as means to support organizations in addressing this challenge. However, what all these methods have in common is that they require a manual analysis of the current *degree of automation*, i.e., they depend on the manual identification of tasks and (sub-)processes that are automated or supported by an information system. This identification task requires a thorough analysis of process-related documentations such as process models and textual process documentations. While especially the latter often provides rich and detailed insights, organizations typically maintain hundreds or even thousands of them [3]. As a result, these methods do not scale for organizations with hundreds of processes

Recognizing the large manual effort required to determine the current degree of automation in an organization's business processes, we use this paper to propose an approach that is able to automatically do so. More specifically, we combine supervised machine learning and natural language processing techniques to automatically identify whether a task described in a textual process description is a (1) manual task, (2) user task (interaction of a human with an information system) or (3) automated task. An evaluation with a set of 424 activities from a total 47 textual process descriptions demonstrates that our approach produces satisfactory results. Therefore, our approach can be employed to reduce the effort required to determine the degree of automation in an organization's processes, as a first step in RPA endeavors.

The rest of the paper is organized as follows. Section 2 illustrates the problem we address using a running example. Section 3 introduces our approach for automatically determining the degree of automation of textual process descriptions on a conceptual level. Section 4 presents the results of our evaluation. Section 5 discusses related work before Sect. 6 concludes the paper.

2 Problem Statement

In this section, we illustrate the problem of automatically identifying the degree of automation of tasks described in a textual process description. Building on the three categories of task automation introduced in [9], our goal is to classify each task from a given textual process description as either (1) manual, (2) user task (interaction of a human with an information system) or (3) automated. Figure 1 shows an exemplary textual process description, the associated relevant process tasks, and their degree of automation.

Figure 1 shows that this textual process description contains two manual tasks, two user tasks, and two automated tasks. The manual tasks include the decision of the supervisor about the vacation request (task 3) and the completion of the management procedures by the HR representative (task 6). The user tasks are the two tasks in the process that are executed using the help of an information system. That is, the submission and the reception of the vacation request (tasks 1 and 2). The automated tasks are tasks executed by the ERP system. This includes returning the application to the employee (task 4) as well as generating

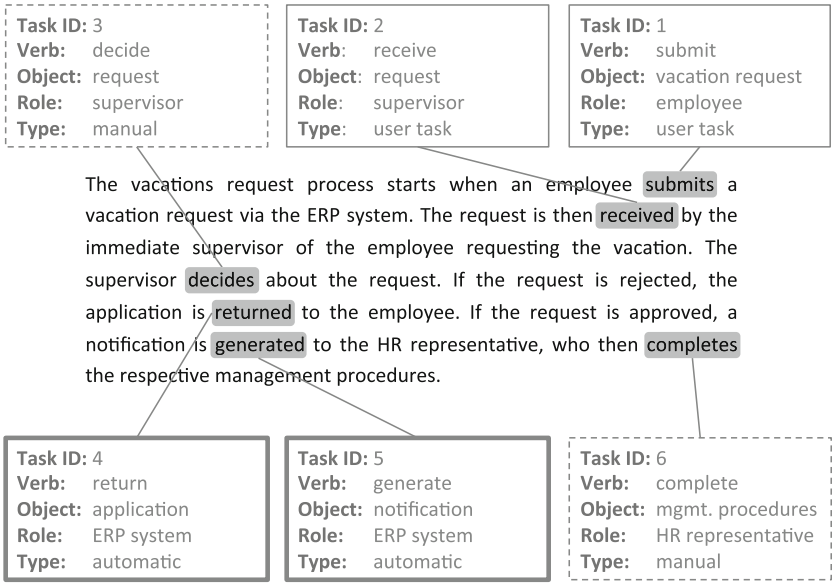

Fig. 1. Process description with highlighted activities and their degree of automation

the notification to the HR representative (task 5). Analyzing this scenario in more detail, reveals that the automatic classification of these tasks is associated with two main challenges:

1. *Identification of tasks*: Before a task can be classified, an automated approach must be able to detect the tasks described in a text. Note, for example, that the verb "*starts*", the verb "*rejected*" as well as the verb "*approved*" do not relate to tasks. The first is not relevant to the classification task at hand because it represents a piece of meta information about the process. The latter two tasks are not relevant because they rather relate to conditions than to tasks, i.e., "*if the request is rejected*" describes a state, rather than an activity being performed. Besides identifying relevant verbs, the identification of tasks also requires to properly infer the *object* to which a verb refers and the resource that executes the task, i.e. the *role*.
2. *Consideration of context*: To reliably predict whether a certain activity is a manual, user, or automated task, an automated approach must be able to take a number of contextual factors into account. Consider, for instance, the receipt of the vacation request (task 2). While in this process description the request is submitted to an information system, this might not be the case in other processes (a request could be also received orally or in writing). The fact that an information system is mentioned in the first sentence, must respectively be considered when classifying a task described later in the process.

In prior work, only the former challenge has been addressed. The technique for generating process models from natural language texts proposed by

Friedrich et al. [10] can reliably recognize and extract tasks from textual process descriptions. To the best of our knowledge, there does not exist any technique that addresses the second challenge. In this paper, we do so by operationalizing the problem of automatically identifying the degree of task automation as multi-class classification problem. In the next section, we elaborate on the details of our proposed solution.

3 Conceptual Approach

In this section, we present our approach for automatically identifying the degree of automation of tasks described in a textual process description. Section 3.1 first gives an overview of the approach. Section 3.2 introduces the dataset we use in this paper, before Sect. 3.3 through Sect. 3.5 elaborate on the details of our approach.

3.1 Overview

The overall architecture of our three-step approach is visualized in Fig. 2. The approach takes as input a textual process descriptions and returns a list of process tasks that are classified according to their degree of automation.

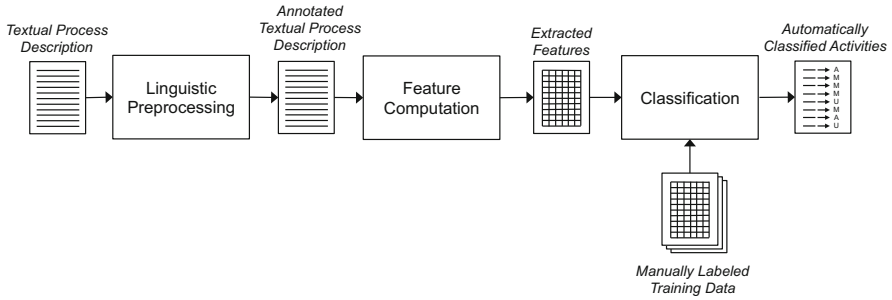

Fig. 2. Overview of the proposed approach

The first step is to parse the text and to identify the relevant linguistic entities and relations that denote tasks in a process description. For instance, we determine which words represent verbs and to which objects they relate. The result of this preprocessing step is a textual process description annotated with the linguistic information related to the process' tasks. The second step is the computation of the features we use for prediction. In particular, we compute features related to the verbs and objects that characterize tasks in a process, the resources that execute tasks, and a feature characterizing terms from IT domains. The output of this step is a feature table that contains the extracted tasks and their corresponding features. In the third step, we perform a classification based on the computed features. In the context of this paper, we use an SVM,

which is a supervised machine learning algorithm that automatically classifies the input based on a set of manually labeled training instances. The output of the classification is a list of tasks, each automatically classified as manual, user, or automated.

In the following sections, we elaborate on each step in more detail. Because of the supervised nature of our classification approach, we begin with introducing our dataset.

3.2 Dataset

For this paper we use a subset of a collection of textual process descriptions introduced in [10]. The collection contains 47 process descriptions from 10 different industrial and scholarly sources. We removed one of these sources (i.e. 14 process descriptions) because the textual descriptions from this source were obtained using Google Translate and their language quality was insufficient for our purposes. To obtain the required classifications for the 424 tasks described in this dataset, two researchers independently classified each task as manual, user, or automated. Conflicts were resolved by involving a third researcher. Table 1 gives an overview of the characteristics of the resulting dataset.

Table 1. Characteristics of dataset

ID	Source	Type	D	S	W/S	MT	UT	AT
1	HU Berlin	Academic	4	10.0	18.1	52	4	1
2	TU Berlin	Academic	2	34.0	21.2	42	38	11
3	QUT	Academic	8	6.1	18.3	51	20	1
4	TU Eindhoven	Academic	1	40.0	18.5	36	8	0
5	Vendor tutorials	Industry	4	9.0	18.2	9	23	2
6	inubit AG	Industry	4	11.5	18.4	9	23	3
7	BPM Practitioners	Industry	1	7	9.7	7	1	0
8	BPMN practice Handbook	Textbook	3	4.7	17.0	14	6	1
9	BPMN guide	Textbook	6	7.0	20.8	30	30	2
Total			**33**	**9.7**	**16.8**	**250**	**153**	**21**

Legend: D = Number of process descriptions per source, S = Average number of sentences, W/S = Average number of words per sentence, MT = Total number of manual tasks per source, UT = Total number of user tasks per source, AT = Total number of automated tasks per source

The data from Table 1 illustrates that the process descriptions from our dataset differ with respect to many dimensions. Most notably, they differ in size. The average number of sentences ranges from 4.7 to 34.0. The longest process description contains a total of 40 sentences. The descriptions also differ in the average length of the sentences. While the descriptions from the *BPM Practitioners* source contain rather short sentences (9.7 words), the process descriptions

from the *TU Berlin* source contain relatively long sentences (21.2 words). The process descriptions also differ with respect to the degree of automation. Some sources contain process descriptions mostly covering manual tasks (e.g. the *HU Berlin* source), others contain a quite considerable number of automated tasks (e.g. the *TU Berlin* source). Lastly, the process descriptions differ in terms of how explicitly and unambiguously they describe the process behavior. Among others, this results from the variety of authors that created the textual descriptions.

3.3 Linguistic Preprocessing

The goal of the linguistic preprocessing step is to automatically extract verbs, object, and roles related to tasks described in the input text. To accomplish this, we build on a technique that was originally developed for the extraction of process models from natural language text [10]. This technique, which is regarded as state-of-the-art [31], combines linguistic tools such as the Stanford Parser [18] and VerbNet [33] to, among others, identify verbs, objects, and roles. The advantage of this technique is its high accuracy and its ability to resolve so-called anaphoric references such as "*it*" and "*they*". To illustrate the working principle of the technique, consider the first sentence from the running example in Fig. 1:

> "*The vacations request process starts when an employee submits a vacation request via the ERP system.*"

The first step is the application of the Stanford Parser, which automatically detects the part of speech of each word as well as the grammatical relations between them. The result of the part-of-speech tagging looks as follows.

> "*The/DT vacations/NNS request/NN process/NN starts/VBZ when/WRB an/DT employee/NN submits/VBZ a/DT vacation/NN request/NN via/IN the/DT ERP/NNP system/NN ./.*"

We can see that the Stanford Parser correctly identifies two verbs "*starts*" and "*submits*" (indicated by the tag "*VBZ*"). The dependency analysis of the Stanford Parser further reveals to which subjects and objects these verbs relate:

nsubj(starts-5, process-4)
nsubj(submits-9, employee-8)
dobj(submits-9, request-12)
compound(request-12, vacation-11)

The verb "*starts*" relates to the subject "*process*" and the verb "*submits*" relates the subject "*employee*" as well as the object "*request*". The Stanford Parser also recognizes that "*vacation request*" is a compound noun (i.e., a noun that consists of several words). Based on the part-of-speech tagging output and the dependency relations, the technique from [10] automatically extracts task records consisting of a verb, an object, and the executing role. It also recognizes that the verb "*start*" in this context represents meta information and not a relevant task. It is respectively not included as a task record. The final set of task records then represents the input to the next step of our approach.

3.4 Feature Computation

The selection and computation of suitable features is the key task when building a machine learning-based solution [8]. Therefore, we manually analyzed which characteristics in our dataset affect the degree of automation of a task. As a result, we selected and implemented four features:

- *Verb feature (categorical)*
- *Object feature (categorical)*
- *Resource type (human/non-human)*
- *IT domain (yes/no)*

In the following paragraphs we elaborate on the definition and rationale of each feature as well as its computation.

Verb Feature. The *verb* feature is a categorical feature and relates to the verb used in the context of a task. The main idea behind this feature is that certain verbs are more likely to be associated with automated tasks than others. As an example, consider the verbs "*generate*" or "*transmit*", which likely relate to automated tasks. The verbs "*analyze*" and "*decide*", by contrast, are more likely to relate to manual tasks. The advantage of introducing a verb feature over using predefined verb classed (such as the Levin verb classes [27]) is that a verb feature does not tie a verb to a specific automation class. The verb "*generate*", for instance, might as well be used in the context of "*generate ideas*" and, thus, refer to a manual task. Such a context-related use can be taken into account when the verb is considered as part of a set of features.

The computation of this feature is straightforward since it is explicitly included in the task record from the linguistic preprocessing step.

Object Feature. The *object* feature is a categorical feature and captures the object that the verb of the task relates to. The rationale behind this feature is, similar to the verb feature, that certain objects are more likely to be associated with automated tasks than others. As an example, consider the two verb-object combinations "*send letter*" and "*send e-mail*". Although both contain the verb "*send*", the object reveals that the former relates to a manual and the latter relates to a user task (sending an e-email certainly requires the interaction with a computer). While the number of objects we may encounter in textual process descriptions is much higher than the number of verbs, including the object as a feature might still help to differentiate different degrees of task automation.

Similar to the verb feature, the computation of this feature is straightforward since it is part of the task record from the linguistic preprocessing step.

Resource Type Feature. The *resource type* feature is a binary feature that characterizes the resource executing a task as either "*human*" or "*non-human*". The reason for encoding the resource as a binary feature instead of a classical

categorical feature is the high number of resources that can execute a task. Depending on the domain of the considered process, resources may, among others, relate to specific roles (e.g. "*manager*" or "*accountant*"), departments (e.g., "*HR department*" or "*accounting department*"), and also systems ("*ERP system*" or "*information system*"). Despite this variety, the key characteristic revealing whether a task is likely to be automated is the type of the resource, that is, whether the resource is human or not. Apparently, a human resource can only relate to a manual or user task, while a non-human resource can also execute automated task (especially when the non-human resource represents an IT system).

Unlike the computation of the verb and the object feature, the computation of the resource type feature is not trivial. The task record from the linguistic preprocessing step only contains the actual resource and no indication of the resource type. To determine the resource type, we use the lexical database WordNet [29]. WordNet groups English words into sets of synonyms, so-called synsets. For each of the 117,000 synsets WordNet contains, it provides short definitions, examples, and a number of semantic relations to other synsets. To compute this feature, we leverage the *hypernym* relationship from WordNet. In general, a hypernym is a more generic term for a given word. For instance, the word "*vehicle*" is the hypernym of "*car*" and the word "*bird*" is the hypernym of "*eagle*". Based on this notion of hypernymy and the hierarchical organization of WordNet, we are able to infer for a given resource whether its hypernym is "*physical entity*", "*abstract entity*" or a "*person*". Based on this hypernym information we then can automatically categorize whether a resource is human or non-human.

IT Domain Feature. The *IT domain* feature is a binary feature that reveals whether a task relates to the IT domain or not. The rationale behind this feature is that a task that relates to the IT domain is likely to be a user task or even an automated task. As example, consider the text fragment "*the customer submits a complaint via the complaint management system*". This fragment contains the human actor "*customer*", the verb "*submit*" and the object "*complaint*". Neither of these elements clearly indicates a degree of automation. However, the fragment also mentions a "*complaint management system*". The goal of the IT domain feature is to take such IT-related context into account.

To compute this feature, we leverage the glossary of computer terms developed by the University of Utah[1]. Besides a comprehensive coverage of technical terms, this list also contains verbs and adjectives that are used in an IT context. If a considered sentence, contains one or more terms from this list, the IT domain feature receives the value "*yes*" for any task that is part of this sentence.

[1] http://www.math.utah.edu/~wisnia/glossary.html.

3.5 Classification

In the final step of our approach, the actual classification of tasks from unseen process descriptions takes place. As described in the previous section, there is not a single feature that independently reveals the degree of automation of a given task. It rather depends on the specific context of the task in the process. To be able to still classify unseen tasks, we employ a Support Vector Machine [6], a supervised machine learning algorithm. The advantages of SVMs are, among others, that they can deal well with relatively small datasets, they have a low risk of overfitting, and they scale well. For these reasons SVMs have also been frequently applied in the context of other text classification tasks [16,35].

The core strategy of an SVM is to find a so-called *hyperplane* that best divides a dataset. While in a two-dimensional space a simple line would be sufficient to do so, an SVM maps the data to higher and higher dimensions until a hyperplane can be formed that clearly segregates the data. Since an SVM is a supervised machine learning algorithm, it needs to be trained on a manually labeled dataset.

In the next section, we describe how we implemented the approach outlined in this section and demonstrate the effectiveness of the approach through a quantitative evaluation.

4 Evaluation

This section reports on our evaluation experiments. We first elaborate on the evaluation setup and the implementation of our approach. Then, we provide a detailed discussion of the results.

4.1 Setup

The goal of the evaluation experiments is to demonstrate that the approach introduced in this paper can reliably determine the degree of automation of previously unseen textual process descriptions. To this end, we implemented our approach as a Java prototype. Besides the code from [10], which we use to extract tasks from a textual process descriptions, we build on the machine learning library Weka [15] to implement the SVM, and JWNL [36] for incorporating and accessing the lexical database WordNet.

To evaluate the performance of our approach, we conducted a repeated 10-fold cross validation using our dataset. The idea behind this validation approach is to randomly split the data set into 10 mutually exclusive subsets (so-called folds) of about equal size. The SVM is then trained on 9 of the 10 folds and tested on the remaining (unseen) fold. This process is repeated 10 times such that, in the end, all data has been used for both training and testing. The advantage of this evaluation method is that it does not require to partition the data set into training and test data. We ran four different configurations of our approach:

1. Training on action feature only (A)
2. Training on action and object feature (A+O)

3. Training on action, object, and resource type feature (A+O+RT)
4. Training on all feature (Full)

To quantify the performance of each configuration, we use the standard metrics precision, recall, and F1-measure. In our context, *precision* for a particular class is given by the number of tasks that were correctly assigned to this class divided by the total number of tasks that were assigned to this class. *Recall* is given by the number of tasks that were correctly assigned to this class divided by the total number of tasks belong to that class. The *F1-measure* is the harmonic mean of the two. Note that precision, recall, and F1-measure are computed for each class individually. To also provide aggregate results, we conduct micro averaging. That is, we use the number of tasks belonging to a particular class to weight the respective precision and recall values. A macro perspective (i.e. applying no weights) would provide a distorted picture because the three classes vary in size.

4.2 Results

The results of the 10-fold cross validation are presented in Table 2. Besides precision, recall, F1-measure for each class and configuration, it also shows the number of correctly and incorrectly classified instances.

Table 2. Results from 10-fold cross validation

		A	A+O	A+O+RT	Full
	Correct	320	320	340	342
	Incorrect	104	104	84	82
Manual	Precision	0.75	0.75	0.80	0.81
	Recall	0.83	0.83	0.90	0.90
	F1-Measure	0.79	0.79	0.85	0.85
User	Precision	0.75	0.75	0.80	0.80
	Recall	0.67	0.67	0.70	0.70
	F1-Measure	0.71	0.71	0.75	0.75
Automated	Precision	0.82	0.82	1.00	0.92
	Recall	0.43	0.43	0.43	0.52
	F1-Measure	0.56	0.56	0.60	0.66
Total (mic.)	Precision	0.75	0.75	0.80	0.81
	Recall	0.75	0.75	0.80	0.80
	F1-Measure	0.75	0.75	0.80	0.81

In general, the results from Table 2 reveal that our approach works well. Out of the 424 task instances, our approach classified 342 correctly. This yields an overall F1-measure of 0.81. Taking a look at the contribution of the individual features shows that the action feature is of particular importance. Apparently the discriminating power of the action feature is considerable, already resulting

in an overall F1-measure of 0.75. We can further see that adding the object feature has no effect at all. However, the resource type feature results in a further improvement of the overall F1-measure to 0.80. The IT domain feature has little effect, but apparently leads to the correct classification of at least two additional task instances.

Analyzing the results for individual classes in more detail shows that there are quite some differences among the classes. Most notably, the F1-measure for the *automated* class (0.66) is much lower than the F1-measure of the *manual* (0.85) and the *user class* (0.75). This is, however, not particularly surprising when taking the class sizes into account. The automated class only contains 21 instances, which clearly makes it a minority class. It is worth noting that the rather low F1-measure mainly results from a low recall (0.52). The precision reveals that automated task are correctly classified in 92% of the cases.

To further illustrate the results, Fig. 3 shows the ROC curves and the corresponding AUC (area under the curve) values for the total configuration.[2] ROC curves are graphical representations of the proportion of true positives versus the proportion of false positives and often used to illustrate the capabilities of a binary classifier. The AUC value represents the probability that a classifier ranks a randomly chosen positive instance higher than a randomly chosen negative one. The AUC value varies between 0 and 1. An uninformative classifier yields an AUC value of 0.5, a perfect classifier respectively yields an AUC value of 1.0. The AUC values for our approach (ranging from 0.75 to 0.78 depending on the class) indicate that our approach represents a classifier with a good performance.

To get insights into the limits of our approach, we conducted an error analysis. More specifically, we investigated which task instances were classified incorrectly and why. In essence, we observed two main types of misclassifications: (1) misclassifications due to a deviating use of feature attributes and (2) misclassifications due to insufficient training data. The first category relates to instances that were classified erroneously because the feature attributes are typically associated with another class. As an example, consider the manual task "*attach the new sct document*". For this task, our approach misses the fact that the "*sct document*" is actually a physical document. It classifies it as a user task because the verb "*send*" is often associated with user tasks in our dataset (e.g. consider "*send e-mail*"). Another example is the user task "*check notes*", which our approach classified as a manual task. Here it did not recognize the context of an information system and bases its decision on the verb "*check*" and the object "*notes*", which are often associated with manual tasks. The second category of misclassifications relates to cases where our approach erroneously classified a task because it has not seen enough training data. For example, consider the user task "*transmit a response comment*", which our approach classified as a

[2] Note that the way Weka generates ROC curves results in only as many threshold values as there are distinct probability values assigned to the positive class. Therefore, the ROC curves from Fig. 3 are only based on three data points. This, however, does not reduce their informative value.

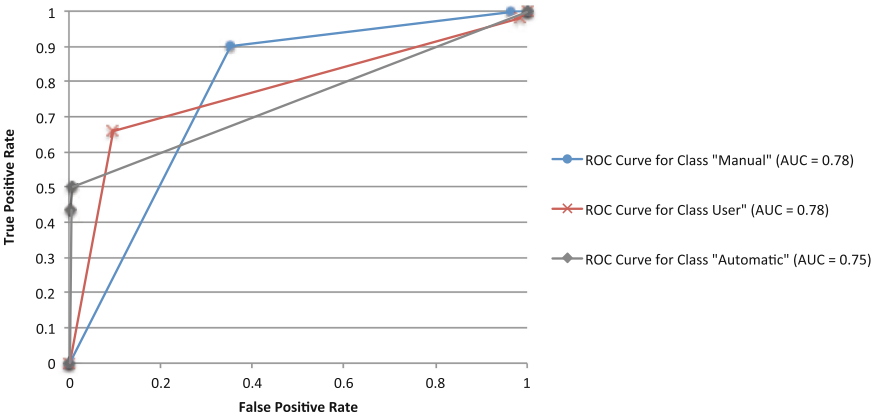

Fig. 3. ROC Curves for each class

manual task. Here the problem is that our approach has not observed a sufficient number of instances using the verb "*transmit*", which clearly relates to the use of an information system.

Despite these misclassifications, we can state that the presented approach represents a promising solution for automatically determining the degree of task automation.

5 Related Work

This paper relates to two major streams of research: (1) the application of Natural Language Process (NLP) technology in the context of business process analysis and (2) process automation.

A variety of authors have applied NLP technology in the context of business process analysis. Their works can be subdivided into techniques that analyze the natural language inside process models and techniques that analyze the natural language outside of process models, typically captured in textual process descriptions. Techniques analyzing the natural language inside process models typically focus on activity and event labels. Among others, there exist techniques for checking the correctness and consistency of process model labels [23,26,30], techniques for identifying similar parts of process models [19,25], and techniques for inferring information from process models such as service candidates [12,24]. Other approaches focus on the analysis of process-related text documents, such as approaches for the automated elicitation of process models from texts, cf. [1, 10,14] and the comparison of natural language texts to process models [2,32], and process querying based on the analysis of textual process descriptions [22].

The focus on *automation* in the context of Business Process Management is not a recent development. In particular research on workflow management and automation reaches back over 20 years [4,13,34]. Research on RPA, by contrast, is still relatively scarce. Lacity and Willcocks investigated how organizations apply RPA in practice [20,21,37]. They found that most applications of

RPA have been done for automating tasks of service business processes, such as validating the sale of insurance premiums, generating utility bills, and keeping employee records up-to date. Their study also revealed the overall potential of RPA ranging from a significant increase in turnaround times and greater workforce flexibility to cost savings of up to 30%. Other authors also studied the risks associated with BPA. For instance, Kirchmer [17] argues that RPA has the potential to make mistakes faster and with higher certainty because there is often no human check before executing an action. Davenport and Kirby also tried to answer the question of what machines are currently capable of. They argue that there are four levels of intelligence that machines can potentially master: (1) support for humans, (2) repetitive task automation, (3) context awareness and learning, and (4) self-awareness. Currently, they conclude, machines are capable of mastering level 1 and 2. Level 3 is only covered to a limited extend, level 4 not at all. They, however, stress that machines are advancing and that it is important to understand how human capabilities fit into the picture [7].

6 Conclusion

In this paper, we proposed a machine learning-based approach that automatically identifies and classifies tasks from textual process descriptions as manual, user, or automated. The goal of our technique is to reduce the effort that is required to identify suitable candidates for robotic process automation. An evaluation with 424 activities from a total of 47 textual process descriptions showed that our approach achieves an F-measure of 0.81 and, therefore, produces satisfactory results.

Despite these positive results, it is important to consider our results in the light of some limitations. First, it should be noted that the dataset we used in this paper is not representative. Textual process descriptions in practice may deviate from the ones in our dataset in different ways. However, we tried to maximize the external validity of our evaluation by choosing a dataset that combines different sources. What is more, our approach could be easily retrained on other datasets to further increase its performance. Second, our approach cannot guarantee that suitable automation candidates are identified. It rather gives an overview of the current degree of automation, which can then serve as input for further analysis.

In future work, we plan to improve the performance of our approach by including additional features and testing other classifiers. What is more, we intend to apply our approach in organizations in order to obtain feedback about its usefulness in practice.

References

1. 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

2. Van der Aa, H., Leopold, H., Reijers, H.A.: Comparing textual descriptions to process models: the automatic detection of inconsistencies. Inf. Syst. **64**, 447–460 (2017)
3. Van der Aa, H., Leopold, H., van de Weerd, I., Reijers, H.A.: Causes and consequences of fragmented process information: insights from a case study. In: Proceedings of the Annual Americas' Conference on Information Systems (2017)
4. van der Aalst, W.M.P., Barros, A.P., ter Hofstede, A.H.M., Kiepuszewski, B.: Advanced workflow patterns. In: Scheuermann, P., Etzion, O. (eds.) CoopIS 2000. LNCS, vol. 1901, pp. 18–29. Springer, Heidelberg (2000). https://doi.org/10.1007/10722620_2
5. Aguirre, S., Rodriguez, A.: Automation of a business process using robotic process automation (RPA): a case study. In: Figueroa-García, J.C., López-Santana, E.R., Villa-Ramírez, J.L., Ferro-Escobar, R. (eds.) WEA 2017. CCIS, vol. 742, pp. 65–71. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66963-2_7
6. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
7. Davenport, T.H., Kirby, J.: Just how smart are smart machines? MIT Sloan Manage. Rev. **57**(3), 21 (2016)
8. Domingos, P.: A few useful things to know about machine learning. Commun. ACM **55**(10), 78–87 (2012)
9. Dumas, M., Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33143-5
10. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 482–496. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21640-4_36
11. Fung, H.P.: Criteria, use cases and effects of information technology process automation (ITPA). Browser Download This Paper (2014)
12. Gacitua-Decar, V., Pahl, C.: Automatic business process pattern matching for enterprise services design. In: IEEE Congress on Services Part II, pp. 111–118 (2009)
13. Georgakopoulos, D., Hornick, M., Sheth, A.: An overview of workflow management: from process modeling to workflow automation infrastructure. Distrib. Parallel Databases **3**(2), 119–153 (1995)
14. Ghose, A.K., Koliadis, G., Chueng, A.: Process discovery from model and text artefacts. In: Proceedings of the IEEE Congress on Services, pp. 167–174. IEEE Computer Society (2007)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newslett. **11**(1), 10–18 (2009)
16. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026683
17. Kirchmer, M.: Robotic process automation-pragmatic solution or dangerous illusion? BTOES Insights, June 2017
18. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: 41st Meeting of the Association for Computational Linguistics, pp. 423–430 (2003)

19. Klinkmüller, C., Weber, I., Mendling, J., Leopold, H., Ludwig, A.: Increasing recall of process model matching by improved activity label matching. In: Daniel, F., Wang, J., Weber, B. (eds.) BPM 2013. LNCS, vol. 8094, pp. 211–218. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40176-3_17
20. Lacity, M., Willcocks, L.P., Craig, A.: Robotic process automation at telefonica O2 (2015)
21. Lacity, M.C., Willcocks, L.P.: A new approach to automating services. MIT Sloan Manage. Rev. **58**(1), 41 (2016)
22. Leopold, H., van der Aa, H., Pittke, F., Raffel, M., Mendling, J., Reijers, H.A.: Searching textual and model-based process descriptions based on a unified data format. Softw. Syst. Model. 1–16 (2017)
23. Leopold, H., Eid-Sabbagh, R.H., Mendling, J., Azevedo, L.G., Baião, F.A.: Detection of naming convention violations in process models for different languages. Decis. Support Syst. **56**, 310–325 (2013)
24. Leopold, H., Mendling, J.: Automatic derivation of service candidates from business process model repositories. In: Abramowicz, W., Kriksciuniene, D., Sakalauskas, V. (eds.) BIS 2012. LNBIP, vol. 117, pp. 84–95. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30359-3_8
25. Leopold, H., Niepert, M., Weidlich, M., Mendling, J., Dijkman, R., Stuckenschmidt, H.: Probabilistic optimization of semantic process model matching. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 319–334. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32885-5_25
26. Leopold, H., Smirnov, S., Mendling, J.: On the refactoring of activity labels in business process models. Inf. Syst. **37**(5), 443–459 (2012)
27. Levin, B.: English Verb Classes and Alternations: A Preliminary Investigation. University of Chicago Press, Chicago (1993)
28. Leyh, C., Bley, K., Seek, S.: Elicitation of processes in business process management in the era of digitization – the same techniques as decades ago? In: Piazolo, F., Geist, V., Brehm, L., Schmidt, R. (eds.) ERP Future 2016. LNBIP, vol. 285, pp. 42–56. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-58801-8_4
29. Miller, G., Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
30. Pittke, F., Leopold, H., Mendling, J.: Automatic detection and resolution of lexical ambiguity in process models (2015)
31. Riefer, M., Ternis, S.F., Thaler, T.: Mining process models from natural language text: A state-of-the-art analysis. In: Multikonferenz Wirtschaftsinformatik (MKWI-16). Universität Illmenau , Illmenau, Germany, 9–11 March 2016
32. Sànchez-Ferreres, J., Carmona, J., Padró, L.: Aligning textual and graphical descriptions of processes through ILP techniques. In: Dubois, E., Pohl, K. (eds.) CAiSE 2017. LNCS, vol. 10253, pp. 413–427. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59536-8_26
33. Schuler, K.K.: Verbnet: a broad-coverage, comprehensive verb lexicon. Ph.D. thesis, Philadelphia, PA, USA (2005)
34. Stohr, E.A., Zhao, J.L.: Workflow automation: overview and research issues. Inf. Syst. Front. **3**(3), 281–296 (2001)
35. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. **2**, 45–66 (2001)
36. Walenz, B., Didion, J.: JWNL: Java wordnet library (2011)
37. Willcocks, L., Lacity, M.C.: Service Automation: Robots and the Future of Work. Steve Brookes Publishing, Warwickshire (2016)