Towards the Automated Annotation of Process Models

Henrik Leopold[1], Christian Meilicke[2], Michael Fellmann[3], Fabian Pittke[4], Heiner Stuckenschmidt[2], and Jan Mendling[4]

[1] VU University Amsterdam,
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands
h.leopold@vu.nl
[2] Universität Mannheim, 68159 Mannheim, Germany
christian|heiner@informatik.uni-mannheim.de
[3] Universität Osnabrück, Katharinenstr. 3, 49074 Osnabrück, Germany
michael.fellmann@uni-osnabrueck.de
[4] WU Vienna, Welthandelsplatz 1, 1020 Vienna, Austria
fabian.pittke|jan.mendling@wu.ac.at

Abstract. Many techniques for the advanced analysis of process models build on the annotation of process models with elements from predefined vocabularies such as taxonomies. However, the manual annotation of process models is cumbersome and sometimes even hardly manageable taking the size of taxonomies into account. In this paper, we present the first approach for automatically annotating process models with the concepts of a taxonomy. Our approach builds on the corpus-based method of second-order similarity, different similarity functions, and a Markov Logic formalization. An evaluation with a set of 12 process models consisting of 148 activities and the PCF taxonomy consisting of 1,131 concepts demonstrates that our approach produces satisfying results.

Keywords: Process Model, Taxonomy, Automatic Annotation

1 Introduction

Nowadays, many organizations use business process models for documenting and improving their operations. However, only a few have recognized the full potential their process models offer. In particular semantic technologies facilitate a wide range of possibilities that go beyond the documentation of business operations [27]. For example, there are techniques available that use process models for checking business process compliance [9,22], for checking the interoperability of business processes [10], and for discovering semantic weaknesses in business processes [2]. However, the limitation of all these approaches is that they build on an existing annotation of the process model activities, for instance, with concepts from a taxonomy. Recognizing this drawback, user-friendly approaches for semantic annotation have been proposed [4]. Still, the manual effort that is required for annotating process models is considerable and, in many cases, even

hardly manageable taking into account that taxonomies often contain hundreds or even thousands of concepts.

In this paper, we present the first approach for automatically annotating process models with the concepts of a taxonomy. At this stage, we focus on activity-based taxonomies such as the Supply-Chain Operations Reference-model (SCOR) [24], the MIT process handbook [17], and the Process Classification Framework (PCF) [1]. To this end, we define an approach that combines semantic similarity measurement with probabilistic optimization. In particular, we use different types of similarity between the process model and the taxonomy as well as the distance between the taxonomy concepts to guide the matching with a Markov Logic formalization. In contrast to prior approaches in the domain of process modeling, we do not measure the similarity using WordNet, but build on the more powerful corpus-based approach of second-order similarity. An evaluation of our approach with a set of 12 process models consisting of 148 activities and the PCF taxonomy with 1,131 concepts shows that our technique performs significantly better than a naive baseline and indeed produces satisfying results.

The rest of the paper is structured as follows. Section 2 illustrates the problem of automatically annotating process models with taxonomy concepts. Section 3 introduces the similarity functions we use for computing the input for our probabilistic optimization. Section 4 introduces Markov Logic Networks and defines the probabilistic optimization problem using a Markov Logic formalization. Section 5 presents the evaluation of our approach. Section 6 discusses related work before Section 7 concludes the paper.

2 Problem Illustration

The goal of this paper is to present an approach for the automated annotation of process models with the concepts of an activity-based taxonomy. It builds on two types of input: a process model P consisting of a set of activities A_p and an activity taxonomy T, which is specified as follows:

Definition 1 (Activity Taxonomy). An activity taxonomy is a tuple $T = (A_t, r, H)$ such that
- A_t is a finite and non-empty set of activities. We refer to them as *concepts*.
- $r \in A$ represents the taxonomy root.
- $H \subseteq A \times (A \setminus \{r\})$ is the set of parent-child relationships such that $(a_1, a_2) \in H$ if a_1 is a parent of a_2.
- H is an acyclic and coherent relation such that each concept $a \in A \setminus \{r\}$ has exactly one direct parent.

We further use $C = \{a \mid (r, a) \in H\}$ to refer to the direct children of the taxonomy root r. They represent the roots of what we refer to as taxonomy categories. The function $c(a) = \{c_a \mid c_a \in C \wedge (c_a, a) \in H^+\}$ returns the category a concept $a \in A_t$ belongs to. Note that the taxonomy categories are disjoint and, hence, $|c(a)| = 1$.

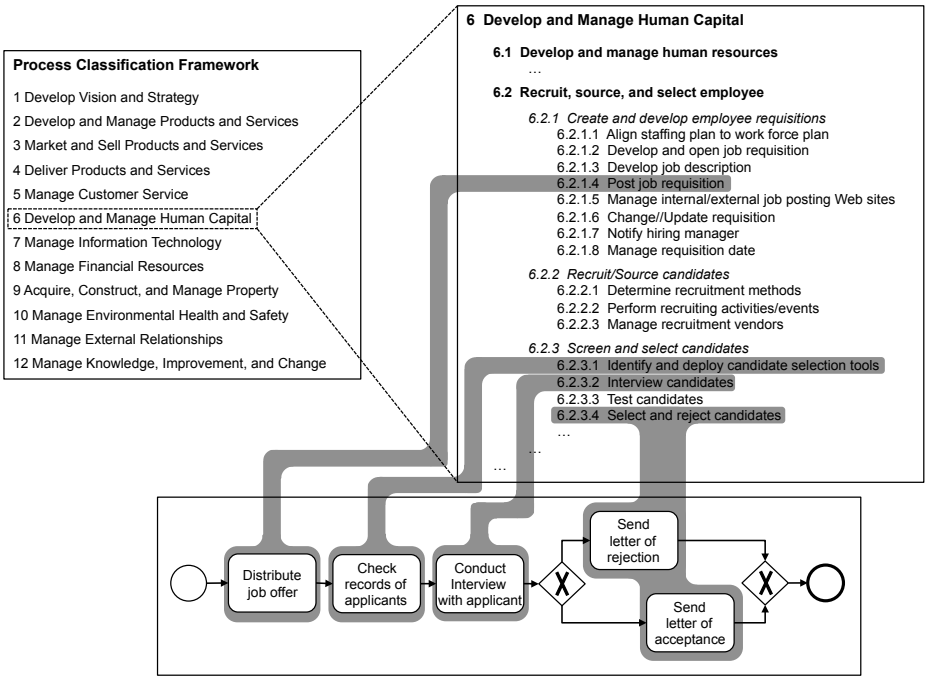

Fig. 1. Correspondences between an exemplary process model and the PCF taxonomy

The annotation of a process model P with the concepts of an activity taxonomy T are captured by the relation $\mathcal{A} : A_p \times A_t$. An element $(a_p, a_t) \in \mathcal{A}$ defines that the activity a_p is annotated with the concept a_t, i.e., they both represent similar semantics. Note that each activity is annotated with at most one concept. However, one concept can be used as annotation for several activities.

Figure 1 illustrates the challenges associated with the automated annotation by showing a simple hiring process and its annotations with the concepts from the PCF taxonomy. In total, the process consists of five activities. First, the job offer is distributed. Afterwards, the records of the applicant are checked and an interview is conducted. Based on the result of the interview, the applicant is either rejected or accepted. Once the corresponding letter was sent, the process is finished. The grey shades visualize the annotations of the activities of the process model with the taxonomy concepts. We observe that all activities belong to the category *Develop and Manage Human Capital*. Considering the annotations in more detail, it becomes clear that the automatic identification of these annotations is by no means trivial. While the activity *Distribute job offer* and the corresponding concept *Post job requisition* at least share the common word *job*, the connection between *Check records of applicants* and *Identify and deploy candidate selection tools* is purely semantic. A similar situation can be observed for the activities *Send letter of rejection* and *Send letter of acceptance*, which are both annotated with the concept *Select and reject candidates*.

Beyond the challenge of recognizing semantic relationships between activities and concepts, we have to take the large number of taxonomy concepts into account. In total, version 5.2 of the PCF taxonomy contains 1,131 concepts. While the exemplary process from Figure 1 only contains activities relating to concepts from the same category, this is no justifiable assumption for a usable approach. In practice, we may encounter cross-sectional processes, which are related to several categories. Hence, we have to consider all concepts from the taxonomy as potential annotation candidates.

To the best of our knowledge, there is currently no technique available that is capable of automatically annotating process model activities with the concepts of a taxonomy. Hence, we define such an approach in the subsequent sections.

3 Similarity between Process Models and Taxonomies

In this section, we introduce the similarity functions we use for generating the input for our optimization problem. In total, we define three separate functions: a function for capturing the similarity between activities and concepts, a function for capturing the similarity between the process model and a taxonomy category, and a distance cost function capturing the distance between taxonomy concepts.

3.1 Similarity between Activities and Taxonomy Concepts

To measure the similarity between a process model activity and a taxonomy concept, we automatically decompose them into their semantic components. As pointed out in [18], activities can be characterized by three components: an action, a business object on which the action is performed, and an optional additional information fragment that is providing further details. As an example, consider the process model activity *Perform interview with employee*. It consists of the action *perform*, the business object *interview*, and the additional information fragment *with employee*. The same procedure can be applied to the concepts of an activity taxonomy. In order to accomplish this decomposition in an automated way, we employ the technique defined in [15].

Building on the decomposition, we compute the semantic similarity between the actions, business objects, and additional information fragments of the considered activity-concept pair. A challenge in this context is the usage of specific terminology from business settings, which is often not fully captured by standard natural language tools such as WordNet [19]. Hence, we determine the similarity between two components using a corpus-based method called second-order similarity [11]. The approach of second-order similarity is based on the statistical analysis of co-occurrences in large text collections and has been implemented in several tools such as NLS [5] or DISCO [13]. In comparison to WordNet, second-order similarity has the advantage that it is not restricted to a set of manually predefined term relations and, hence, is more powerful for our purposes. In order to calculate the semantic similarity between a process model activity a_p

and an taxonomy concept a_t, we introduce three functions: a component similarity function *cpsim*, a coverage function *cov*, and a activity-concept similarity function *sim*, combining the latter two into a final result.

The function *cpsim* calculates the semantic similarity between two components cp_1 and cp_2 derived from an activity-concept pair. In general, the second-order similarity sim_{SO} is returned. In case one or both of the concepts represent an empty string, *cpsim* returns zero.

$$cpsim(cp_1, cp_2) = \begin{cases} 0 & \text{if } cp_1 = \epsilon \vee cp_2 = \epsilon \\ sim_{SO}(cp_1, cp_2) & \text{if } cp_1 \neq \epsilon \wedge cp_2 \neq \epsilon \end{cases} \tag{1}$$

The coverage function *cov* is used to determine the number of components of an activity or a concept $a \in A_p \cup A_t$. Note that the index *act* in the definition denotes an action, *bo* a business object and *add* an additional information fragment.

$$cov(a) = |\{cp \mid cp \neq \epsilon \wedge cp \in a_{act}, a_{bo}, a_{add}\}| \tag{2}$$

To combine the similarity results from the previously defined functions, we introduce the function *sim*. It calculates the arithmetic mean of the similarity values for action, business object, and the additional information fragment. This is accomplished by dividing the sum of $cpsim_{act}$, $cpsim_{bo}$ and $cpsim_{add}$ by the maximum coverage among the input activity-concept pair $a_p \in A_p$ and $a_t \in A_t$. As a result, we obtain the overall semantic similarity for an activity-concept pair.

$$sim(a_p, a_t) = \frac{cpsim_{act}(a_p, a_t) + cpsim_{bo}(a_p, a_t) + cpsim_{add}(a_p, a_t)}{\underset{a \in \{a_p, a_t\}}{\arg\max}\ cov(a)} \tag{3}$$

By calculating *sim* for every activity-concept pair, we obtain a set of similarity values. These values form the basis for our automatic annotation approach.

3.2 Similarity between Process Models and Taxonomy Categories

For activities containing frequently occurring words the sole consideration of the similarity function *sim* may not be sufficient for identifying the best fitting concept. As an example, consider the activity *Develop strategy*, whose business object *strategy* occurs in six categories of the PCF taxonomy. In such a situation, it would be helpful to quantify the similarity between the entire process model and the different taxonomy categories. The resulting values could complement the individual similarity scores derived from *sim* in order to truly identify the best fitting candidate.

To quantify the similarity between a process model and a taxonomy category, we adopt the general idea of term frequency-inverse document frequency (tf-idf) and the vector space model from the domain of information retrieval [23] and modify them to meet the characteristics of our problem. Using the tf-idf it is possible to determine the discriminative power of a word. In the context of a

taxonomy, the tf-idf assigns high weights to words that frequently occur in a particular category but rarely in the entire taxonomy. Contrarily, words that occur in many or even all categories have hardly any discriminative power and hence receive a low weight. Let A_c denote the concepts from a category $c \in C$ and $f(w, c)$ the frequency of a word w among the concepts of c. Then, the tf-idf is defined as follows:

$$tf\text{-}idf(w, c) = f(w, c) \times log \frac{|C|}{|c \in C : w \in A_c|} \tag{4}$$

Based on the tf-idf values, we can create a vector representation for the entire process model as well as for taxonomy categories. Therefore, we calculate the tf-idf values for each word from A_p and A_c and store them in the vectors v_p and v_c. As a result, we obtain a vector representation of the process model and the category in a vector space. Using the cosine similarity, it is now possible to measure the distance between these vectors and to quantify the similarity between a process model and a category. Accordingly, we introduce the similarity function rel, which we define as follows:

$$rel(c) = cos(v_p, v_c) = \frac{v_p \cdot v_c}{||v_p||\ ||v_c||} = \frac{\sum_{i=1}^{n} v_{p_i} \times v_{ci}}{\sqrt{\sum_{i=1}^{n} (v_{p_i})^2} \times \sqrt{\sum_{i=1}^{n} (v_{ci})^2}} \tag{5}$$

By calculating rel for each category $c \in C$, we receive a set of similarity values which complement the similarity values from sim.

3.3 Distance Costs between Taxonomy Concepts

Besides the semantic perspective, we also need to take the structure of the process model and the taxonomy into account. Assuming that process models describe the underlying process in a rather coherent fashion, we would not expect large "leaps" between the annotations of two neighboring activities. For instance, we would assume that two subsequent activities are rather annotated with concepts *6.2.1* and *6.2.6* than with *7.1* and *3.2.1.5*. To penalize such leaps, we introduce a distance cost function dc based on concept similarity introduced by Wu and Palmer [28]. It quantifies the distance between two concepts based on the graph structure of the taxonomy and their least common superconcept a_s. Given two concepts $a_1, a_2 \in A_t$, we define dc as follows:

$$dc(a_1, a_2) = \Big(-0.5 + \frac{2 \times N3}{N1 + N2 + 2 \times N3} \Big) \times 2 \tag{6}$$

where N1 is the number of concepts on the path from a_1 to a_s, N2 is the number of concepts on the path from a_2 to a_s, and N3 is the number of concepts on the path from a_s to the taxonomy root r.

By calculating dc for each concept pair, we obtain a set of distance cost values in the interval [-1,1], i.e., only big leaps are penalized. Together with the

previously introduced similarity values they form the input for our probabilistic annotation model.

4 Using Markov Logic for Automatic Annotation

Markov Logic (ML) is a formalism that combines first order logic with undirected probabilistic models, i.e., Markov Networks [21]. A Markov Logic formalization of a given problem consists of a set of weighted and unweighted logical formulae. These formulae describe observations relevant to the concrete problem instance and general constraints that have to hold for each instance of the problem class. By replacing variables with concrete values, which is called grounding, it is possible to transform the Markov Logic formalization into a Markov Logic Network. For technical details we refer to [21].

Two types of inference can be applied to a Markov Logic Network, known as marginal inference and maximum a-posteriori (MAP) inference. In the context of our work, we are interested in MAP inference. MAP inference computes the most probable assignment of truth values to the ground atoms of the given formalization. The MAP state, which is the result of applying MAP inference, corresponds in our setting to the most probable annotation of activities with concepts. Since the underlying probabilistic model is log linear, the MAP state is the solution which is maximal with respect to the sum of weights attached to the formulae.

In the previous section, we introduced functions for measuring similarity, relatedness, and distance costs. To incorporate these functions into our Markov Logic formalization, we introduce a predicate for two of these functions and weigh each grounded atom with the value that results from applying the corresponding similarity function. For the sake of simplicity, we use the same names for these predicates as introduced for the corresponding similarity measures, i.e., we use

- $sim(a_p, a_t)$ to express that activity a_p and concept a_t are similar.
- $dc(a_t, a'_t)$ to express that a_t and a'_t are located close to each other.

These predicates are used to generate a comprehensive set of weighted atoms by computing all possible groundings. Additionally, we add unweighted formulae that describe the sequence flow in the given process model and the structure of the taxonomy. In particular, we use the predicates

- $suc(a_p, a'_p)$ to express that activity a_p directly succeeds activity a'_p,
- $cat(c, a_t)$ to express that a_t belongs to the category c, i.e., $c(a_t) = c$.

Now we define the constraints that relate the given evidence to the *annotate* predicate. The groundings of the *annotate* predicate correspond to the solution of the annotation problem. First of all, we add a constraint that enforces to annotate each activity with only one concept, i.e., we define the predicate *annotate* to be functional.

$$\langle annotate(a_p, a_t) \wedge annotate(a_p, a'_t) \rightarrow a_t = a'_t,\ \infty \rangle \tag{7}$$

Note that we represent a weighted formula as a pair, where the first element is the formula itself and the second element is the associated weight. Using ∞ as weight we refer to a hard (unweighted) formula that has to be true in every possible world. Now we link the activity-concept similarity to the *annotate* predicate.

$$\langle sim(a_p, a_t) \rightarrow annotate(a_p, a_t),\ \infty \rangle \tag{8}$$

This formula means that the weight attached to $sim(a_p, a_t)$ is added only to the objective of our optimization problem if $annotate(a_p, a_t)$ is part of the solution. The model we defined so far will create a functional mapping as MAP state that is optimal with respect to the similarity weights given in our evidence with the guarantee that the mapping is functional. We extend this model by taking the relatedness score into account. For each category c we add one weighted formula using the following schema.

$$\langle cat(c, a_t) \wedge annotate(a_p, a_t),\ rel(c) \rangle \tag{9}$$

If c is the category to which a_t belongs to and if a_p is annotated with a_t, the relevance score $rel(c)$ of category c is added to the objective. By adding these weighted rules, we ensure that concepts from a category that are more relevant with respect to the given process model are preferred over concepts from less relevant categories.

Finally, we want to penalize pairs of annotations where two consecutive activities a_p and a'_p are annotated with two concepts a_t and a'_t that are located at different places in the taxonomy. Since a_p and are a'_p are directly connected by a sequence flow, we would expect that their counterparts are located closely to each other in the taxonomy. The following constraint enforces that we have to add the distance cost, which is the weight associated to $dc(a_t, a'_t)$, whenever two consecutive activities a_p and a'_p are annotated with a_t and a'_t.

$$\langle annotate(a_p, a_t) \wedge annotate(a'_p, a'_t) \wedge suc(a_p, a'_p) \rightarrow dc(a_t, a'_t), \infty \rangle \tag{10}$$

The positive impact of this constraint can be best explained with the help of Figure 1. Suppose that the concepts *6.2.3.2* and *6.1.2.3* have similarly high similarity values for the activity *Conduct Interview with Applicant*. Further suppose that our approach detected a high similarity between *Send letter of rejection* and concept *6.2.3.4*. Due to Formula 10, *6.2.3.2* will now be preferred over *6.1.2.3* as annotation for *Conduct Interview with Applicant*, because there is a large distance between *6.1.2.3* and *6.2.3.4*, while *6.2.3.2* and *6.2.3.4* are located close to each other. This illustrates nicely that our approach solves the annotation problem as a whole taking interdependencies between potential annotations into account.

5 Evaluation

To demonstrate the applicability of our approach, we conduct an evaluation with a set of manually annotated process models and the PCF taxonomy. The goal of

the evaluation is to learn how well our approach can approximate the manual annotation. Section 5.1 introduces the data set we use for the evaluation. Section 5.2 introduces the details of the evaluation setup. Finally, Section 5.3 presents the evaluation results.

5.1 Test Collection

As we are the first to present an automated approach for annotating process models with taxonomy concepts, there is currently no commonly accepted test sample available. Hence, we use a set of BPMN process models that was created during a PCF case study by students from the University of Osnabrück, Germany. In the context of this case study, groups of students were asked to model a set of three fictitious business processes from the area of change management, product development, and human resources using BPMN. In addition, they had to annotate the activities of the models with the corresponding PCF concepts. We use the original annotations created by the students without any modifications. The resulting model set comprises twelve manually annotated BPMN process models consisting of a total of 148 activities.

Table 1. Overview of the test collection

Model	#Activities	Topic	PCF Categories
1	9	Change Management	12
2	8	Change Management	12
3	9	Change Management	12
4	9	Change Management	12
5	12	Product Development	2, 3
6	15	Product Development	1, 2, 3, 4
7	14	Product Development	2, 3
8	16	Product Development	2
9	10	Human Resources	6
10	11	Human Resources	6
11	19	Human Resources	6
12	15	Human Resources	6

Table 1 gives an overview of the model characteristics including the number of activities, the main topic of the model, and the PCF categories the activities of the process model were assigned to. It shows that the models vary in size as well as their coverage of the different PCF categories. Moreover, some models are cross sectional (i.e., models 5, 6, and 7) while others only belong to a single PCF category. Thus, we believe that the test set is well-suited to demonstrate the applicability of our annotation approach.

5.2 Setup

For evaluating the approach presented in this paper, we implemented it in the context of a prototype. The prototype is based on the activity analysis technique from [15], the second-order similarity implementation DISCO [13], and the Markov Logic Network implementation RockIt [20]. We used our prototype to automatically generate the annotations for the models from our test set and the text-based PCF taxonomy version 5.2. We then compared the automatically generated annotations $\mathcal{A}$ with the manual annotation from the students $\mathcal{R}$. Based on this comparison, we can assess the quality of the annotation by computing the metrics precision and recall:

$$pre(\mathcal{A}, \mathcal{R}) = \frac{|\mathcal{A} \cap \mathcal{R}|}{|\mathcal{A}|} \qquad rec(\mathcal{A}, \mathcal{R}) = \frac{|\mathcal{A} \cap \mathcal{R}|}{|\mathcal{R}|}$$

In our context, precision is the number of correct annotations computed by our approach divided by the number of annotations our approach proposed. Recall is the number of correct annotations computed by our approach divided by the total number of annotations according to the manually created gold standard.

However, the drawback of the standard precision and recall metrics for our context is that they only consider annotations that are correct up to the last sub category. As an example, consider an activity a_p that was manually annotated with the concept *6.2.1.4*. If our approach proposes to annotate a_p with the concept *6.1.2.2*, this would be simply considered as incorrect although the first three levels of the manually annotated concept were actually identified correctly. To provide for a more fine granular perspective on our results, we introduce a level-based form of precision and, recall which is in line with the approach presented in [8].

To this end, we introduce a function $parent_i(a_t)$, which returns the i^{th} parent of a concept $a_t \in A_t$. We define $parent_i(a_t) = parent(parent_{i-1})$ for all $i > 0$ and $parent_i(a_t) = a_t$ for all $i \leq 0$. Thus, for instance, $parent_2(a_t)$ returns the concept *6.2* for the input concept *6.2.1.4*. We further introduce a function $level(a_t)$, which returns the level of a concept a_t in the taxonomy. It, for example, returns 4 for the concept *6.2.1.4* and 2 for the concept *6.2*. Based on these definitions, we introduce a function l_n, which maps a set of annotations $\mathcal{A}$ to a set of less fine-grained annotations from level n:

$$l_n(\mathcal{A}) = \bigcup_{(a_p, a_t) \in \mathcal{A}} (a_p, parent_{level(a_t)-n}(a_t))$$

As an example, consider the set of annotations {(*Distribute job offer*, *6.2.1.4*), (*Recruit employees*, *6.2.2*)}. For these annotations, l_2 would return {(*Distribute job offer*, *6.2*), (*Recruit employees*, *6.2*)} and l_1 would return {(*Distribute job offer*, *6*), (*Recruit employees*, *6*)}. Based on l_n, we are now able to define the level-based form of precision and recall.

$$pre_n(\mathcal{A}, \mathcal{R}) = \frac{|l_n(\mathcal{A}) \cap l_n(\mathcal{R})|}{|\mathcal{A}|} \qquad rec_n(\mathcal{A}, \mathcal{R}) = \frac{|l_n(\mathcal{A}) \cap l_n(\mathcal{R})|}{|\mathcal{R}|}$$

As the PCF taxonomy that is used in the context of this evaluation has four levels, we accordingly use four levels of precision and recall to evaluate our results. In addition, we report the f-measure fm_n for each level, which is the harmonic mean of pre_n and rec_n. Respectively, the metrics pre_4, rec_4, fm_4 provide information about annotations that are correct up to the forth level and pre_1, rec_1, fm_1 provide information about annotations that are at least correct with respect to the main category.

5.3 Results

To demonstrate the applicability of the approach presented in this paper, we tested different configurations:

- **Baseline**: As baseline configuration, we annotated each activity $a_p \in A_p$ with the concept $a_t \in A_t$ with the highest value for $sim(a_p, a_t)$. We did not include any aspects from the above introduced ML formalization.
- **ML with Wu & Palmer**: For this configuration we used our ML formalization to compute the best annotations based on the activity-concept similarity sim and the distance dc between taxonomy concepts.
- **ML with Category Weighting**: For this configuration we used our ML formalization to compute the best annotations based on the activity-concept similarity sim and the category weights rel.
- **Full ML Configuration**: For the full configuration we included all previously discussed aspects into the ML formalization: the activity-concept similarity sim, the distance dc between taxonomy concepts, and the category weights rel.

Table 2 summarizes the results of our experiments. It shows that the baseline configuration without Markov logic yields quite low results. The value of 0.20 for the metric pre_1 indicates that only one out of five activities is annotated with a concept from the correct main category. The consideration of the additional similarity measures in the context of our Markov implementation improves the results significantly. While the sole use of the Wu & Palmer distance improves the results only slightly, the sole use of the category weighting already has a big effect. The f-measure fm_1 of the category weighting configuration rises from 0.20 to 0.76 and fm_2 rises from 0.18 to 0.38. Apparently, the additional category weight helps to rule out candidates that have high values for sim, but generally cannot be related to the process model. The positive effect of the Wu & Palmer distance can be observed for the full configuration. The combination of the category weighting and the Wu & Palmer distance causes an additional increase of fm_2 from 0.38 to 0.45. The value of fm_1, however, remains identical. This effect can be explained by the fact that the Wu & Palmer distance does not cause the approach to consider additional (and potentially correct) concepts. It rather improves the coherence of the annotations among the already considered candidates. Hence, it improves fm_2 without affecting fm_1. In addition, it does not compromise fm_3 and fm_4 too much. The small losses can be explained by a dominance of the Wu

Table 2. Evaluation Results

Configuration	n	pre_n	rec_n	fm_n
Basline	1	0.20	0.21	0.20
	2	0.11	0.11	0.11
	3	0.07	0.08	0.07
	4	0.03	0.03	0.03
ML with Wu & Palmer	1	0.29	0.23	0.25
	2	0.21	0.17	0.18
	3	0.10	0.08	0.08
	4	0.04	0.03	0.04
ML with Category Weighting	1	0.76	0.77	0.76
	2	0.37	0.38	0.38
	3	0.16	0.17	0.16
	4	0.07	0.08	0.08
Full ML Configuration	1	0.76	0.77	0.76
	2	0.44	0.45	0.45
	3	0.14	0.14	0.14
	4	0.05	0.06	0.06

& Palmer distance on the third and fourth level where the *sim* value would have actually indicated the correct annotation.

Altogether, the results suggest that both the category weights as well as the Wu & Palmer distance are helpful for finding correct and ruling out incorrect annotations. Taking the huge complexity of the annotation problem into account - each activity has more than thousand potential annotations - the results have to be considered as good. About 76% of all activities were annotated with a concept from the correct main category and 44% of all activities were annotated with the correct subcategory. Figure 2 gives an indication of where our approach could be improved by showing the values of fm_n for each model separately.

From the numbers from Figure 2 we can learn that particularly the zero values for models 3 and 8 negatively affect the overall result. The reason for these numbers can be found in the use of highly specific words for which we were not able to obtain a similarity value. As a result, the weight *rel* dominates the small values from *sim* and causes an erroneous annotation of the entire process model. In fact, the use of specific words also causes a major share of the incorrect annotations among the other models. As an example, consider the activity *Sales contact OEM for details* from model 8. Due to the domain specific abbreviation *OEM*, the values *sim* do not help us to identify the correct concepts from the taxonomy.

Besides the problem with specific words, the reason for the sub optimal results on the levels three and four is given by the fact that some concepts are semantically quite close. As an example, consider the activity *Recommend*

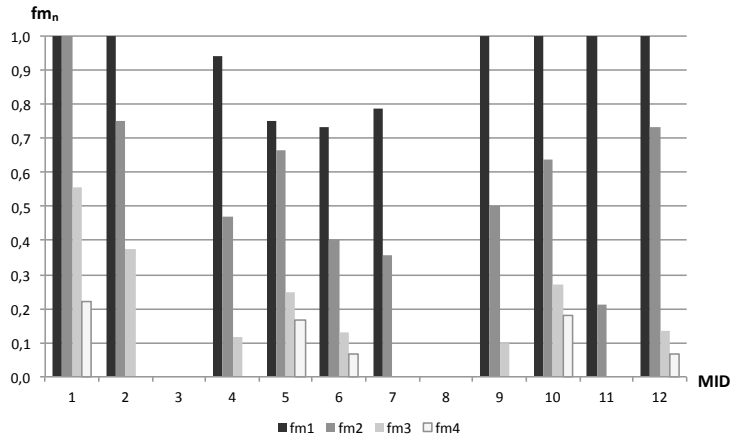

Fig. 2. Detailed results of fm_n for each model from the test set

candidate from model 9. Our approach annotated this activity with the concept *6.2.4.3 Recommend/not recommend candidate* whereas the manual annotation assigned it to *6.2.3.4 Select and reject candidates*. Although the manual annotation is an arguably better choice than the computed one, both concepts represent semantically similar choices.

6 Related Work

The work presented in this paper relates to two major streams of research: process model annotation and process model matching.

Research addressing process model annotation typically aims at describing general guidelines and strategies [16] or the benefits and potentials associated with the annotation [25]. Some approaches also automatically filter relevant concepts from the considered ontology [4,7,3]. The final decision about the annotation is, however, still taken by the user. Hence, we are, to the best of our knowledge, the first who present an automatic approach for annotating process models.

Process model matching aims at the automatic identification of correspondences between two process models. In prior work, a plethora of process model matching approaches has been proposed [6]. Typically, they build on a combination of structural or behavioral properties with different types of textual similarity. Some rely on rather simplistic techniques such as the Levenshtein distance [26], others use WordNet for computing textual similarity [14,12]. However, so far, no approach has considered the use of second-order similarity. Besides these conceptual differences, it is worth noting that the overall complexity of automated annotation is considerably higher. While a process model typically does not consist of more than 30 activities, taxonomies often contain more than thousand concepts [1].

7 Conclusion

In this paper, we presented the first approach for automatically annotating process models with concepts of a taxonomy. Our approach uses a Markov Logic formalization to combine the results of different similarity functions. In contrast to prior approaches from the area of process modeling, we did not use WordNet for measuring the relatedness of words but the corpus-based method of second-order similarity. An evaluation of our approach with 12 process models consisting of 148 activities and the PCF taxonomy consisting of 1,131 concepts showed that our approach performs significantly better than a naive baseline and is able to compute satisfying results.

As for future work, we consider two main directions. First, we aim at improving the performance of our approach. Promising directions for accomplishing this goal include the consideration of additional information of the process model such as the control flow and the improvement of the similarity measurement by training DISCO with domain-specific corpora. Second, we plan to study to what extent our approach can be transferred to other types of ontologies as, for instance, the MIT Process Handbook, which uses both part-of as well as is-a relations to structure business activities.

References

1. APQC. Apqc process classification framework (pcf) - cross industry - pdf version 5.2.0. Technical report, 2012.
2. J. Becker, P. Bergener, M. Räckers, B. Weiß, and A. Winkelmann. Pattern-based semi-automatic analysis of weaknesses in semantic business process models in the banking sector. 2010.
3. A. Bögl, M. Schrefl, G. Pomberger, and N. Weber. Semantic annotation of epc models in engineering domains to facilitate an automated identification of common modelling practices. In *ICEIS*, pages 155–171, 2008.
4. M. Born, F. Dörr, and I. Weber. User-friendly semantic annotation in business process modeling. In *Web Information Systems Engineering–WISE 2007 Workshops*, pages 260–271. Springer, 2007.
5. Z. Cai, D. S. McNamara, M. Louwerse, X. Hu, M. Rowe, and A. C. Graesser. Nls: A non-latent similarity algorithm. In *Proc. 26th Ann. Meeting of the Cognitive Science Soc.(CogSci'04)*, pages 180–185, 2004.
6. U. Cayoglu, R. Dijkman, M. Dumas, P. Fettke, L. Garcıa-Banuelos, P. Hake, C. Klinkmüller, H. Leopold, A. Ludwig, P. Loos, et al. The process model matching contest 2013. In *4th International Workshop on Process Model Collections: Management and Reuse (PMC-MR'13)*, 2013.
7. C. Di Francescomarino and P. Tonella. Supporting ontology-based semantic annotation of business processes with automated suggestions. In *Enterprise, Business-Process and Information Systems Modeling*, pages 211–223. Springer, 2009.
8. M. Ehrig, J. Euzenat, et al. Relaxed precision and recall for ontology matching. In *Proc. K-Cap 2005 workshop on Integrating ontology*, pages 25–32, 2005.
9. G. Governatori, J. Hoffmann, S. Sadiq, and I. Weber. Detecting regulatory compliance for business process models through semantic annotations. In *Business Process Management Workshops*, pages 5–17. Springer, 2009.

10. P. Hofferer. Achieving business process model interoperability using metamodels and ontologies. 2007.
11. A. Islam and D. Inkpen. Second order co-occurrence pmi for determining the semantic similarity of words. In *LREC*, pages 1033–1038, 2006.
12. C. Klinkmüller, I. Weber, J. Mendling, H. Leopold, and A. Ludwig. Increasing recall of process model matching by improved activity label matching. In *BPM*, pages 211–218. Springer, 2013.
13. P. Kolb. Disco: A multilingual database of distributionally similar words. *Proceedings of KONVENS-2008, Berlin*, 2008.
14. H. Leopold, M. Niepert, M. Weidlich, J. Mendling, R. M. Dijkman, and H. Stuckenschmidt. Probabilistic optimization of semantic process model matching. In *BPM*, pages 319–334, 2012.
15. H. Leopold, S. Smirnov, and J. Mendling. On the refactoring of activity labels in business process models. *Information Systems*, 37(5):443–459, 2012.
16. Y. Lin, D. Strasunskas, S. Hakkarainen, J. Krogstie, and A. Solvberg. Semantic annotation framework to manage semantic heterogeneity of process models. In *CAiSE*, pages 433–446. Springer, 2006.
17. T. W. Malone, K. Crowston, and G. A. Herman. *Organizing business knowledge: the MIT process handbook*. MIT press, 2003.
18. J. Mendling, H. A. Reijers, and J. Recker. Activity Labeling in Process Modeling: Empirical Insights and Recommendations. *Information Systems*, 35(4):467–482, 2010.
19. G. Miller and C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA, 1998.
20. J. Noessner, M. Niepert, and H. Stuckenschmidt. Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models. In *AAAI Workshop: Statistical Relational Artificial Intelligence*, 2013.
21. M. Richardson and P. Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.
22. S. Sadiq, G. Governatori, and K. Namiri. Modeling control objectives for business process compliance. In *BPM*, pages 149–164. Springer, 2007.
23. G. Salton and M. J. McGill. Introduction to modern information retrieval. 1983.
24. S. Stephens. Supply chain operations reference model version 5.0: A new tool to improve supply chain efficiency and achieve best practice. *Information Systems Frontiers*, 3(4):471–476, 2001.
25. O. Thomas and M. Fellmann. Semantic Process Modeling – Design and Implementation of an Ontology-Based Representation of Business Processes. *Business & Information Systems Engineering*, 1(6):438–451, 2009.
26. M. Weidlich, R. M. Dijkman, and J. Mendling. The icop framework: Identification of correspondences between process models. In *CAiSE*, LNCS, pages 483–498. Springer, 2010.
27. B. Wetzstein, Z. Ma, A. Filipowska, M. Kaczmarek, S. Bhiri, S. Losada, J.-M. Lopez-Cob, and L. Cicurel. Semantic business process management: A lifecycle based requirements analysis. In *SBPM*, 2007.
28. Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics, 1994.