Listen to me: Improving Process Model Matching through User Feedback

Christopher Klinkmüller[1], Henrik Leopold[2], Ingo Weber[3,4], Jan Mendling[2], and André Ludwig[1]

[1] Information Systems Institute, University of Leipzig, Leipzig, Germany*
{klinkmueller,ludwig}@wifa.uni-leipzig.de
[2] Wirtschaftsuniversität Wien, Augasse 2-6, A-1090 Vienna, Austria
{henrik.leopold,jan.mendling}@wu.ac.at
[3] Software Systems Research Group, NICTA, Sydney, Australia**
ingo.weber@nicta.com.au
[4] School of Computer Science & Engineering, University of New South Wales

Abstract. Many use cases in business process management rely on the identification of correspondences between process models. However, the sparse information in process models makes matching a fundamentally hard problem. Consequently, existing approaches yield a matching quality which is too low to be useful in practice. Therefore we propose to investigate user feedback to improve the matching quality. To this end, we analyze which information is suitable for learning. On this basis, we design an approach that performs matching in an iterative, mixed-initiative approach: we determine correspondences between two models automatically, let the user correct them and analyze this input to adapt the matching algorithm. Then, we continue with presenting the results for the next two models. This approach improves the matching quality, as showcased by a comparative evaluation. From this study, we also derive strategies on how to maximize the quality while limiting the workload.

Keywords: BPM, process similarity, process model matching

1 Introduction

More and more organizations use process models as a tool for managing their operations. Typical use cases for process models range from process documentation to the implementation of workflow systems. Once a repository of process models reaches a certain size, there are several important use cases which require the comparison of process models. Examples include validating a technical

* The work presented in this paper was partly funded by the German Federal Ministry of Education and Research under the projects LSEM (BMBF 03IPT504X) and LogiLeit (BMBF 03IPT504A).

** NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.

implementation of a business process against a business-centered specification [2], process model search [6, 13, 10], or identifying clones in process models [7].

The demand for techniques that are capable of comparing process models has led to the development of a variety of *process model matchers*. These matchers, e.g. [24, 14, 11], are usually designed for universal applicability. That is, they are based on common matching metrics used to assess pairs of activities and define classification rules which are believed to provide meaningful indications of similarity for activities in any pair of process models. However, the insufficient accuracy of these approaches [3] suggests that the assumption of universal applicability is too strict, and might hinder effective application in practice.

For this reason, we seize the idea of an adaptive matcher. A similar idea was discussed in [23] where characteristics of a certain process model collection are analyzed to select well-suited matchers for the collection. In contrast to this approach, we build on an *iterative, mixed-initiative approach that utilizes user feedback to constantly adapt the matching algorithm*. It aims to maximize the matching quality improvements while introducing a minimized additional effort for the user in correcting correspondences. Therefore, correspondences between two models are presented to the user, and she is asked to add missing and remove incorrect ones. Based on this feedback, the matching algorithm is updated by analyzing the user's decisions. Once this is done, the correspondences between the next two models are presented. This inquiry of feedback is repeated until the matching system provides good results. The quality of this approach strongly depends on the information that is considered during the analysis. Thus, we substantiate its design by deriving indicators from related research and by assessing their suitability for feedback analysis. Furthermore, an evaluation based on benchmark matching samples demonstrates the benefits of the approach.

The rest of the paper is organized as follows. Section 2 defines process model matching and introduces the state of the art. Section 3 provides an overview of correspondence indicators derived from related research and investigates their potential for user feedback analysis. Based on this survey, Section 4 defines our approach that incorporates feedback. Section 5 evaluates the approach using simulated feedback from gold standards. Finally, Section 6 concludes the paper.

2 Foundations: Problem Illustration and Related Work

This section introduces the problem of process model matching in more detail. First, we explain the problem of matching process models in Subsection 2.1. Afterwards, we review the state of the art in Subsection 2.2.

2.1 Problem Illustration

In accordance with ontology matching [8], process model matching is the process of identifying an *alignment* between two process models. In this paper, a process model is regarded as a *business process graph* as defined in [4]: a process model consists of labeled nodes of different types and directed edges connecting

them. While the edges define the control flow of the process, the nodes express activities, gateways, etc. This abstract notion of process models permits the application of our work to other notations like Petri nets, Event-driven Process Chains (EPCs) or Business Process Model and Notation (BPMN).

Definition 1 (Process model, Set of activities). *Let $\mathcal{L}$ be a set of labels and $\mathcal{T}$ be a set of types. A process model p is a tuple (N, E, λ, τ), in which:*

- *N is the set of nodes;*
- *$E \subseteq N \times N$ is the set of edges;*
- *$\lambda : N \to \mathcal{L}$ is a function that maps nodes to labels; and*
- *$\tau : N \to \mathcal{T}$ is a function that assigns types to nodes.*

For a given process model $p = (N, E, \lambda, \tau)$ the set $A = \{a | a \in N \wedge \tau(a) = activity\}$ is called the set of activities, where we require $\forall a \in A, n \in N : |\{n|(a, n) \in E)\}| \leq 1$ and $|\{n|(n, a) \in E)\}| \leq 1$. Furthermore, we require that there only exists one start $(\exists n \in N, \forall n_i \in N : (n_i, n) \notin E)$ and one end node $(\exists n \in N, \forall n_i \in N : (n, n_i) \notin E)$.

Given two process models p_1, p_2 and their activity sets A_1, A_2 an *alignment* is a set of *correspondences*, i.e. activity pairs (a_1, a_2) with $a_1 \in A_1$ and $a_2 \in A_2$ that represent similar functionality. This binary relation depicts more complex correspondences between sets of activities (A_1^*, A_2^*) with $A_1^* \subseteq A_1$ and $A_2^* \subseteq A_2$ as the set of all activity pairs that they consist of $\{(a_1^*, a_2^*)|(a_1^* \in A_1^* \wedge a_2^* \in A_2^*)\}$.

Fig. 1 presents an alignment between two university admission process models which will be used as a running example throughout the paper. Both processes represent the scenario of receiving, evaluating, and deciding about an application. Hence, activities from one process related to one of these tasks are matched with activities dealing with the same task in the other process. While α_2 and β_2 constitute a one-to-one correspondence, β_6 is not matched. Moreover, there are two complex correspondences: a one-to-many correspondence formed by α_1, α_1 and β_2 and a many-to-many correspondence comprised of α_3, α_4, α_5, β_4 and β_5.

Applying a matcher to automatically determine alignments will only be useful if it yields a high quality, i.e. if it meets the user's expectations. This will be the case when the number of correctly identified correspondences (*true positives*) is high. Consequently, only a few correspondences should be missed (*false*

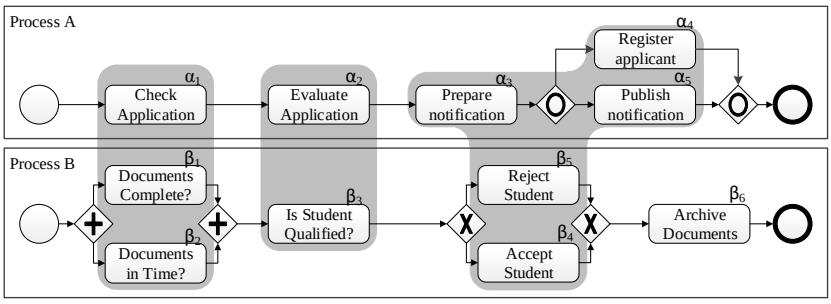

Fig. 1: An example for a process model alignment

negatives). Moreover, the results of a good matcher also contain as few erroneous correspondences (*false positives*) as possible.

2.2 Related Work

The foundations for research in process model matching can be found in various works on schema and ontology matching [1, 8] as well as in research on process model similarity. Such process similarity techniques exploit different sources of information such as text [5, 12], model structure [9, 4], or execution semantics [13, 26]. An overview is provided in [5].

Approaches for process model matching typically derive attributes of activity pairs from these techniques and aggregate these attribute in a predefined static classifier in different ways (see e.g. [24, 14, 11]). In [23], the idea of a more dynamic assembly of matchers is discussed. Therefore, matchers are allocated to properties of process model pairs. By evaluating these properties within a model collection, appropriate matchers are selected and composed.

However, up until now there is no automated technique for process matching available that achieves results comparable to those in the field of ontology matching. In fact, a comparison of techniques developed by different researchers revealed that the best matcher achieved an f-measure of 0.45 on the test data sets [3]. This calls for improving precision and recall of existing techniques. To this end, we investigate suitable matching indicators and user feedback.

3 Information for User Feedback Analysis

The goal of analyzing user feedback is to reveal models that can predict user decisions with a high chance. Therefore, indicators whose values are highly correlated to the decisions, i.e., whether activity pairs correspond or not, are needed [19]. For example, label similarity is generally seen as a good indicator: activity pairs with a high similarity tend to correspond more likely than those with a low similarity. As various information sources, e.g. structure and execution semantics, can be considered, we systematically identify suitable indicators following a two-step approach. In Subsection 3.1, we present indicators derived from related work and investigate their potential for feedback analysis in Subsection 3.2.

3.1 Indicator Definitions

Approaches from related work rely on various characteristics of activities to judge whether activities correspond. From analyzing related work, we identified five categories: *position* and *neighborhood* based on the model structure, *label specificity* and *label semantics* referring to the labels, and *execution semantics*. Thereby, some approaches rely on a certain modeling notation or do not explicitly define the characteristics. In order to assess whether these characteristics can be used for feedback analysis, we present indicators that follow the approaches from related work, but that are suited to our notation.

Basically, we define indicators as similarity functions from the set of activity pairs A^2 to the interval $[0, 1]$, where a value of 0 indicates total dissimilarity, a value of 1 identity, and values in between a degree of similarity. Most of the presented indicators utilize an attribute function $at : A \to \mathbb{R}_{\geq 0}$ which returns a value measured with regard to a certain property of an activity. Those indicators are referred to as *attribute indicators*. Given an activity pair they indicate the similarity of these activities with regard to a certain attribute.

Definition 2 (Attribute indicator). *Let A_1, A_2 be two sets of activities and $a_1 \in A_1$, $a_2 \in A_2$ be two activities. The attribute indicator i_{at} is then defined as:*

$$i_{at}(a_1, a_2) = \begin{cases} 0 & \max\limits_{a \in A_1}(at(a)) = 0 \wedge \max\limits_{a \in A_2}(at(a)) = 0 \\ 1 - \left| \frac{at(a_1)}{\max\limits_{a \in A_1}(at(a))} - \frac{at(a_2)}{\max\limits_{a \in A_2}(at(a))} \right| & else \end{cases}$$

Position. Process models might represent the same abstract process. In such cases, it is more likely for activities at similar positions to correspond than for activities whose positions differ. This idea is pursued in the *Triple-S* approach, which takes the relative position of nodes in the process models as a similarity indicator [3]. According to our definition, each process model has one start and one end node. Thus, we view these nodes as anchors and consider the distances to these nodes, i.e. the smallest number of activities on paths from a node to the start or end node, as attributes to define the attribute indicators $\sigma^{start}_{pos}, \sigma^{end}_{pos}$.

The position of an activity can also be defined with reference to the *Refined Process Structure Tree* (RPST) [24, 23]. The RPST is a hierarchical representation of a process model consisting of single-entry-single-exit fragments [20]. Each RPST fragment belongs to one of four structured classes: Trivial fragments (T) consist of two nodes connected with a single edge. A Bond (B) represents a set of fragments sharing two common nodes. Polygons (P) capture sequences of other fragments. In case a fragment cannot be classified as trivial, bond, or polygon, it is categorized as a rigid (R). Fig. 2 presents the RPST of the process A.

The idea is to view the depth of the non-trivial fragments that contain the activity as an attribute for the position of the model structure (σ^{rpst}_{pos}), i.e., the deeper an activity is located in the RPST the more decision points need to be passed to get to the activity. As activities have at most one incoming and at most one outgoing edge, there are no more than two trivial fragments an activity is part of and these trivial fragments are part of the same non-trivial fragment.

Table 1 illustrates the position indicators for (α_1, β_1) from the running example. Both activities have a distance to the start event of 0. As the structure

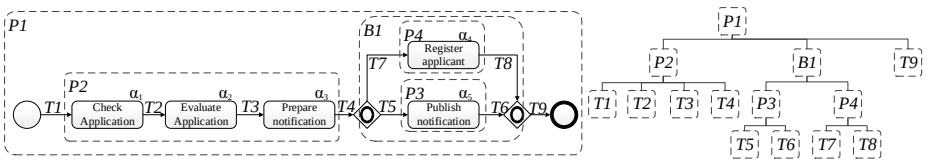

Fig. 2: The fragments of the admission process of university A and the RPST

Table 1: Attribute indicators for an activity pair from the running example.

	a_1	$\max_{a \in A_A}$	b_1	$\max_{a \in A_B}$	(a_1, b_1)		a_1	$\max_{a \in A_A}$	b_1	$\max_{a \in A_B}$	(a_1, b_1)
σ^{start}_{pos}	0	3	0	3	1.00	$\sigma_{\lvert label \rvert}$	2	2	2	3	0.67
σ^{end}_{pos}	3	3	3	3	1.00	$\sigma_{\rightsquigarrow}$	4	4	4	4	1.00
σ^{rpst}_{pos}	2	3	3	3	0.67	σ_{+}	0	0	0	1	0.00
σ^{model}_{neigh}	1	3	2	4	0.83	$\sigma_{\parallel}$	0	1	1	1	0.00
σ^{rpst}_{neigh}	2	2	0	0	0.00						

of both processes is similar they also have the same distance to the end node. Thus, both attribute indicators are 1. As activity β_1 is located in a parallel block and α_1 is not, their RPST positions differ leading to an indicator value of 0.67.

Neighborhood. Whereas the position attributes consider the global location of activities in a model, we now investigate the local structure. In this regard, the Triple-S approach [3] considers the ratios of incoming and outgoing edges. As our definition requires activities to have at most one incoming and at most one outgoing edge these ratios would not provide much information. Instead, we define the structural neighborhood indicator (σ^{model}_{neigh}) based on the undirected version of the process model. We count the activities that are connected to an activity by at least one sequence of distinct edges not containing any activities.

We also consider the RPST for comparing the local structure of activities and define the RPST neighborhood indicator (σ^{rpst}_{neigh}). Therefore, we determine the trivial fragments an activity is part of and count their sibling fragments.

Table 1 also shows examples for the neighborhood indicators. α_1 has one neighbor (α_2) and in process A α_3 has the most neighbors (α_2, α_4, α_5). Similarly, β_1 has two neighbors (β_2, β_3) and the maximum is four neighbors for β_3 (β_1, β_2, β_4, β_5). Thus, the structural neighborhood of both activities is similar (0.83). The RPST neighborhood indicator is 0, because maximum in process B is 0. Each activity belongs to two trivial fragments which are the only parts of a polygon fragment. Therefore, the size of the neighborhoods of these fragments is always 0 and therewith the maximum in process B, too.

Label Specificity. According to an analysis of matching challenges in [11], label specificity (i.e., one label containing more detailed information than another) had a big impact on the identification of correspondences. Thus, we assume activities with a similar specificity to correspond more likely than those with different specificities. An attribute indicator in this regard is defined upon the label length ($\sigma_{\lvert label \rvert}$), i.e., the more words a label contains, the more specific information it provides. It is considered for matcher selection in [23] and for label pruning in [11]. The label length is defined as the number of individual words in a label without common *stop words* like "the", "if", and "to". The individual words of an activities label are returned by the function $\Omega : \mathcal{L} \rightarrow \mathcal{P}(\mathcal{W})$. As Table 1 shows that $|\Omega(\alpha_1)| = |\Omega(\beta_1)| = 2$. Moreover, the maximum label length in process A is 2. In process B β_3 ("Is student qualified") has the longest label of length 3 whereas β_2 ("Documents in Time?") consists of two individual words, because of "in" being a stop word. Thus, the label length indicator is 0.67.

Table 2: Word occurrences and term frequencies in the admission processes

	check	application	documents	complete
occurrences	1	2	3	1
term frequency	0.33	0.67	1.00	0.33

We further assume frequently occurring words to be more specific than less frequently occurring words. This idea is also pursued for label pruning in [11]. Thus, we rely on the term frequency which is well known in information retrieval. It is defined as the number of occurrences of a certain word in a document. On the one hand, we take the union of all activity labels in the model collection as a document and define the function $tf_{coll} : \mathcal{W} \rightarrow [0, 1]$ to return the number of a word's occurrences in the model collection divided by the maximum number determined for a word in the collection. On the other hand, we define $tf_{2p} : \mathcal{W} \rightarrow [0, 1]$ by using all activity labels in the examined model pair to create the document. Based thereon, we define the term frequency indicators σ_{tf}^{coll} and σ_{tf}^{2p}.

Definition 3 (Term frequency indicators). *Let a_1, a_2 be two activities. Then, the term frequency indicators σ_{tf}^{coll} and σ_{tf}^{2p} are defined as:*

$$\sigma_{tf}^{coll}(a_1, a_2) = 1 - |\frac{1}{|\Omega(a_1)|} * \sum_{\omega \in \Omega(a_1)} tf_{coll}(\omega) - \frac{1}{|\Omega(a_2)|} * \sum_{\omega \in \Omega(a_2)} tf_{coll}(\omega)|$$

$$\sigma_{tf}^{2p}(a_1, a_2) = 1 - |\frac{1}{|\Omega(a_1)|} * \sum_{\omega \in \Omega(a_1)} tf_{2p}(\omega) - \frac{1}{|\Omega(a_2)|} * \sum_{\omega \in \Omega(a_2)} tf_{2p}(\omega)|$$

Table 2 illustrates the model pair based indicator. "Documents" occurs most often in the pair. Thus, the term frequencies are yielded by dividing the occurrence values with 3. As the average term frequency of α_1 ("Check Application") is 0.50 and for β_2 ("Documents Complete?") it is 0.67, the indicator yields 0.83.

Label Semantics. Every matching approach relies on the calculation of label similarities as an indicator to which degree activities constitute the same functionality. Prior research has shown that the basic bag-of-words similarity [11] yields good results [3]. It calculates a symmetric similarity score $\sigma.\omega : \mathcal{W}^2 \rightarrow [0..1]$ for each pair of individual words (ω_1, ω_2) with $\omega_1 \in \Omega(a_1)$ and $\omega_2 \in \Omega(a_2)$. Based thereon, it is then defined as the mean of the maximum similarity score each individual word has with any of the individual words from the other label.

Definition 4 (Basic bag-of-words similarity). *Let a_1, a_2 be two activities. The basic bag-of-word similarity $\sigma.\lambda$ is then defined as:*

$$\sigma.\lambda(a_1, a_2) = \frac{\sum_{\omega_1 \in \Omega(a_1)} \max_{\omega_2 \in \Omega(a_2)} (\sigma.\omega(\omega_1, \omega_2)) + \sum_{\omega_2 \in \Omega(a_2)} \max_{\omega_1 \in \Omega(a_1)} (\sigma.\omega(\omega_1, \omega_2))}{|\Omega(a_1)| + |\Omega(a_2)|}$$

Table 3 illustrates the computation of the basic bag-of-words similarity for α_1 ("Check Application") and β_2 ("Documents complete?"). To compute the similarity of a pair of words, we relied on the maximum of the Levenshtein similarity [15] and the Lin similarity [16]. This measure sees high values in both, syntax (Levenshtein) and semantics (Lin), as evidence for similarity.

Table 3: Example for the basic bag-of-words similarity

	document	complete	*max*
check	0.78	0.25	0.78
application	0.11	0.18	0.18
max	0.78	0.25	$\sigma.\lambda = 0.50$

Behavior. Lastly, there are approaches that account for the behavioral context of activities within a process model. Such behavioral attributes are proposed as indicators for matcher selection [23], considered for probabilistic match optimization [14] and also implemented in the ICoP framework [21]. The idea is that corresponding pairs have similar execution semantics, whereas non-corresponding pairs do not. Therefore, we rely on the notion of *behavioral profile* [22] which comprises three relations between activities in a process model defined upon the set of all possible execution sequences. Two activities are in *strict order* ($a_1 \rightsquigarrow a_2$), if there exist execution sequences that contain a_1 and a_2, but in all such sequences a_2 is executed after a_1. Two activities are *exclusive* ($a_1 + a_2$) if no sequence contains both activities, or *interleaving* ($a_1 \parallel a_2$) if there are sequences in which a_1 occurs before a_2 and there are sequences in which a_2 occurs before a_1. For each type of relation, we count the number of relations the given activity participates in. Based on these counts we define the attribute indicators $\sigma_{\rightsquigarrow}$, σ_{+} and $\sigma_{\parallel}$ which are illustrated in Table 1, too. While the (α_1, β_1) have an identical number of strict order relations (their execution can be followed by the execution of up to four activities), they do not share similar characteristics with regard to the other behavioral attributes. On the one hand, there are no exclusive activities in process A at all. Thus, the maximum in process A and the according attribute indicator yield a value of 0. On the other hand, there is one interleaving relation in each process ($\alpha_4 \parallel \alpha_5$ and $\beta_1 \parallel \beta_2$). As β_1 is part of one of these relations and α_1 not, the according indicator is 0.

3.2 Applicability Assessment

Having a set of indicators, we now need to analyze whether they can be used to derive models that can predict user's decisions. For an indicator to be applicable, there must be a correlation between its values or value ranges and the classes.

As the suitability of an indicator cannot be predicted in general, it must be estimated with regard to particular data sets (i.e., process collections) for which the set of correspondences is known (i.e., a *gold standard* of correspondences exists). To this end, we used the two process collections and respective gold standards from the matching contest in 2013 [3]: processes on birth certificates and university admission. More precisely, we took the set of all corresponding and the set of all non-corresponding activity pairs for both data sets as representative samples for both classes. At this point, it should be noted that some of the process models in the university admission data set are not sound, which is a necessary prerequisite for computing the behavior attributes. Thus, we only considered the sound university admission models for these attributes.

Table 4: p-values of the Kolmogorov–Smirnov test for the birth certificate (gray rows) and the university admission (white rows).

σ_{pos}^{start}	σ_{pos}^{end}	σ_{pos}^{rpst}	σ_{neigh}^{model}	σ_{neigh}^{rpst}	$\sigma_{\lvert label\rvert}$	σ_{tf}^{coll}	σ_{tf}^{2p}	$\sigma.\lambda$	$\sigma_{\rightsquigarrow}$	σ_{+}	$\sigma_{\parallel}$
0.001	**0.010**	0.967	0.054	**0.010**	0.581	**0.000**	0.111	**0.000**	**0.000**	0.111	0.211
0.000	0.367	0.155	0.286	0.468	0.210	0.016	0.699	**0.000**	**0.001**	0.864	0.393

To assess the correlation of classes and indicator values, we first examined the distributions of indicator values within both classes. The rationale is that classes can only be assigned to value ranges if the values are distributed differently across the classes. Therefore, we randomly drew 100 activity pairs from each class per attribute. The reason is that the number of non-corresponding activity pairs is roughly 30 times as high as the number of corresponding pairs in both data sets, which would distort our analysis. Next, we conducted a two-sided Kolmogorov-Smirnov [17] test at a significance level of 0.01 with these samples. The neutral hypothesis of this test is that the examined distributions are equal and will be rejected if the yielded p-value is lower than the significance level. Table 4 summarizes the p-values yielded for each attribute. Bold values highlight p-values that are below the significance level.

As can be seen from the table, there are only three attributes (σ_{pos}^{start}, $\sigma.\lambda$, and $\sigma_{\rightsquigarrow}$) for which the null hypothesis is rejected in both cases. From this analysis these three attributes seem suitable for classification, but we will also consider σ_{tf}^{coll} as its p-value are only marginally larger than the significance level.

We further substantiated our analysis by investigating how well each class can be assigned to a value range of an indicator. Therefore, we measured the *information gain* [19], a well established measure from statistics, as an indicator for the entropy of class assignments within subsets of activity pairs with regard to all pairs. More precisely, we calculated the values of all activity pairs for each of the four attributes (σ_{pos}^{start}, $\sigma.\lambda$, σ_{tf}^{coll}, $\sigma_{\rightsquigarrow}$). We then determined two subsets of pairs with regard to one of the attributes and to a threshold. For all pairs in the first subset the attribute value is smaller than the threshold, whereas the values of pairs in the second subset are larger than it. We considered all possible separations of activity pairs that satisfied this rule and chose the separation with the highest information gain for each attribute. The rationale is that the respective subsets constitute the best separation of corresponding and non-corresponding pairs with regard to the considered attribute. As can be seen from Table 5, $\sigma.\lambda$ yields the highest and σ_{pos}^{start} the lowest information gain, σ_{tf}^{coll} and $\sigma_{\rightsquigarrow}$ are in between. To convey a better intuition for this measure, Fig. 3 shows the distribution of the relative value frequencies for $\sigma.\lambda$ and σ_{pos}^{start} as well as for σ_{tf}^{coll} as a representative for the indicators with medium information gains.

Table 5: Information gains for the selected attributes for the birth certificate (gray rows) and the university admission (white rows).

$\sigma.\lambda$	σ_{tf}^{coll}	$\sigma_{\rightsquigarrow}$	σ_{pos}^{start}
0.056	0.023	0.016	0.005
0.027	0.010	0.007	0.002

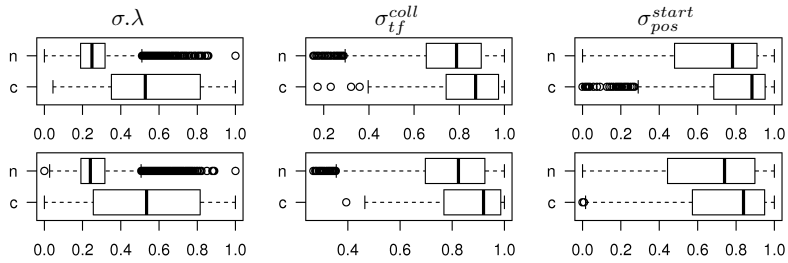

Fig. 3: Box plots for corresponding (c) and non-corresponding (n) activity pairs representing three indicators for the birth certificate (upper row) and the university admission (lower row) data sets.

According to these box plots a threshold at about 0.4 would yield a good classifier for $\sigma.\lambda$ as many corresponding and only a few non-corresponding activity pairs have values larger than this threshold. For the other indicators whose distributions differ only slightly there is no threshold which would classify that well. Thus, *we only consider $\sigma.\lambda$ for user feed back analysis* and introduce a mixed-initiative approach which aims at increasing the applicability of $\sigma.\lambda$ for separating activity pairs in the next section.

4 Word Similarity Adaptation

The incorporation of user feedback opens the opportunity to analyze the user's decisions and adjust the matching process accordingly. Here, we rely on corrections made by the user to proposed alignments. Therefore, we let the user select a pair of process models and automatically determine an alignment. Presenting it to the user, she is asked to remove incorrect and add missing correspondences. These corrections are passed to the algorithm which examines the feedback and adapts its classification mechanism. Afterwards, the next matching process can be started by the user. Fig. 4 illustrates this basic approach.

As outlined in Section 3, we will only consider the basic bag-of-words similarity $\sigma.\lambda$ for correspondence identification. Given a predefined threshold we classify all activity pairs with a basic bag-of-words similarity score higher than or equal to the threshold as correspondences.

Although our analysis shows this attribute to have the most desirable properties, there will still be false positives and false negatives leading to an unsatisfactory matching quality [3]. Hence, it is the goal of the feedback analysis to understand why mistakes were done and how they could have been avoided.

With regard to the matching process a false positive was suggested because the similarity of the activity pair was estimated too high, i.e., it should have been lower than the threshold. In case of a false negative, it is the other way around, i.e., the similarity should have been higher than the threshold. The main reasons for such wrong assessments originate not directly in the basic bag-of-words similarity, but in the underlying word similarity measure $\sigma.\omega$. Those measures are either syntactic, not considering word meaning, or semantic being based on ex-

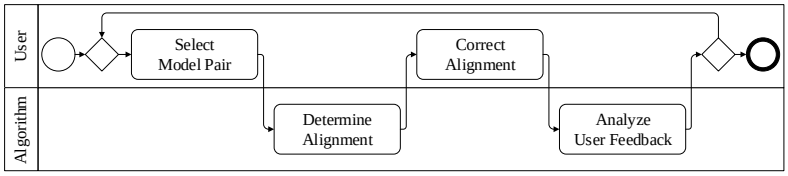

Fig. 4: Basic mixed-initiative approach to learning

ternal sources of knowledge like lexical databases or corpora [18]. As the creation of such databases or corpora incurs huge manual effort, usually matchers rely on universal ones. In both cases, i.e. syntactic matching or semantic matching using universal corpora, the word similarity measures do not sufficiently account for domain-specific information, e.g., technical vocabulary or abbreviations, and thus introduce errors.

Consequently, when the user feedback indicates a misclassification of an activity pair, our learning approach checks which pairs of words contributed to that misclassification. According to the definition of the basic bag-of-words similarity, a word pair contributes to an activity pair classification each time it yields the highest similarity score for one word in the respective activity labels. Therefore, in order to adjust the word similarities to the domain characteristics of the considered process model collection, we decrease the similarity of a pair of words whenever it contributed to a false positive, and increase the similarity for a false negative. We do so by defining two counting functions: $\gamma_{fp} : (\omega_1, \omega_2) \to \mathbb{N}$ returns the number of counted false positive contributions for a word pair, and $\gamma_{fn} : (\omega_1, \omega_2) \to \mathbb{N}$ analogously for false negative contributions. Based on these counters, we introduce a word similarity correction term.

Definition 5 (Word similarity correction). *Let ω_1, ω_2 be two words. Furthermore, let $\rho_{fp}, \rho_{fn} \in \mathbb{R}$ be two predefined learning rates. The correction function $\delta : \mathcal{W}^2 \to \mathbb{R}$ is then defined as:*

$$\delta(\omega_1, \omega_2) := \rho_{fp} \times \gamma_{fp}(\omega_1, \omega_2) + \rho_{fn} \times \gamma_{fn}(\omega_1, \omega_2)$$

Note that the counts are multiplied with learning rates; together with the threshold these are the control parameters of the approach.

Given this correction term and an ordinary word similarity measure $\sigma.\omega_o$, we introduce the *adaptive word similarity* $\sigma.\omega_\alpha$.

Definition 6 (Adaptive word similarity). *Let ω_1, ω_2 be two words. Furthermore, let $\delta : \mathcal{W}^2 \to \mathbb{R}$ be a function that returns a correction value for a word pair. The adapting word similarity function $\sigma.\omega_\alpha : \mathcal{W}^2 \to [0..1]$ is then defined as:*

$$\sigma.\omega_\alpha(\omega_1, \omega_2) := \begin{cases} 1 & \sigma.\omega_o(\omega_1, \omega_2) + \delta(\omega_1, \omega_2) > 1 \\ 0 & \sigma.\omega_o(\omega_1, \omega_2) + \delta(\omega_1, \omega_2) < 0 \\ \sigma.\omega_o(\omega_1, \omega_2) + \delta(\omega_1, \omega_2) & \textit{else} \end{cases}$$

Since $\sigma.\omega_o(\omega_1, \omega_2) + \delta(\omega_1, \omega_2)$ might return a value outside the interval $[0, 1]$, but any $\sigma.\omega$ function is expected to stay within these bounds, we enforce the

bounds as per the first and second case in the above definition. We then use $\sigma.\omega_\alpha$ as $\sigma.\omega$ in the basic bag-of-words similarity when determining the alignment between two process models.

To illustrate this approach, we refer to Table 3 which outlines the computation of $\sigma.\lambda$ for (α_1, β_1). In previous work [11] we found that a threshold above 0.6 yields good results. In this case, the matching algorithm will classify the activity pair as non-corresponding. Collecting user feedback this will be revealed as a wrong classification. Thus, the false negative counter will be increased by 2 for ("check", "complete") as this word pair yielded the highest value for both words and by one for ("complete", "check") and for ("complete", "application"). Having ρ_{fp} set to 0.1 the adaptive word similarity will now roughly be 0.6. Thus, an activity pair with the labels of α_1 and β_1 will now be classified as corresponding.

5 Evaluation

This section has two objectives. First, we want to analyze if our mixed-initiative approach improves the results of existing matchers with regard to the amount of missing and incorrect correspondences. Second, we aim to derive strategies to minimize the amount of user feedback required to achieve high matching quality.

Experiment Setup. Our evaluation utilizes the birth certificate and the university admission data sets from the matching competition [3]. The gold standards serve a dual purpose here: (i) assessing the matching quality and (ii) simulating user feedback. Therefore, going through a sequence of model pairs, we first determine an alignment for the current pair and assess the quality of this alignment. That is, we determine the number of true positives (TP), false positives (FP) and false negatives (FN) given the gold standard. We then calculate the standard measures of precision (P) $(TP/(TP+FP))$, recall (R) $(TP/(TP+FN))$, and f-measure as their harmonic mean (F) $(2 \times P \times R/(P+R))$. Next, we pass the sets of false positives and false negatives to the algorithm which adapts the word similarities accordingly. Then, we move on to the next pair. The average (AVG) and the standard deviation (STD) of all measures and model pairs are used to assess the approach's quality. These are calculated either as a running statistics *during* learning, or as an overall quality indicator *after* all model pairs have been matched and the respective feedback has been considered.

We sampled the space of possible threshold values over the interval [0,1] in steps of 0.05 as well as the space of possible false positive and false negative learning rates over the interval [0,0.2] in steps of 0.01. Moreover, we randomly

Table 6: Best results from matching contest and for word similarity adaptation

	Birth Certificate						University Admission					
	Precision		Recall		F-Measure		Precision		Recall		F-Measure	
Approach	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
Baseline	.68	.19	.33	.22	.45	.18	.56	.23	.32	.28	.41	.20
Adaptive	.73	.15	.67	.24	**.69**	.18	.60	.20	.56	.25	**.58**	.21

generated different model pair sequences in order to check the influence of the model pair order on the quality. We used the maximum of the Levenshtein [15] and the Lin [16] similarities as the ordinary similarity measure.

Matching Results. Table 6 compares the results of our mixed-initiative approach to a baseline comprised of the best results from the matching competition [3], i.e., the RefMod-Mine/NSCM results for the birth certificate and the bag-of-words similarity with label pruning for the university admission data set. The results for the mixed-initiative approach were determined for collecting user feedback over all model pairs. We observed an increase of the f-measure by 0.24 for the birth certificate and by 0.17 for the university admission data set. While the precision remained stable, there was a dramatic improvement in the recall.

Deriving strategies. To derive strategies for minimizing the user workload, we first investigated if the order in which process model pairs are considered by learning had impact on the overall quality. For this purpose, we determined the quality of the basic bag-of-words similarity for each model pair. Then, we split the model pairs for each data set into three equal-sized classes, i.e., model pairs with a high, a medium, and a low f-measure. We generated three sequences (*high*, *medium*, and *low*) where each sequence starts with 12 model pairs of the respective class, randomly ordered, followed by the remaining 24 model pairs, also in random order. Fig. 5 shows the running average f-measure after the i*th* iteration for all three sequences per data set. The results suggest that the order only has a small impact on the final quality, since the average f-measures converge to roughly the same value as the number of iterations increases. However, the running average can be misleading: if we start learning with pairs that are already matched well before learning (as in the *high* case), how much can we learn from them? To examine this aspect, we ran a different experiment, where learning is stopped after the i*th* iteration, and the f-measure over *all* pairs is computed. The results are shown in Fig. 6, left. Looking at the data, one might hypothesize that here the user workload *per model pair* is lower in the *high* case than for the other sequences. Thus, we also counted the number of changes a user has to do until learning is stopped. These effort indicators are shown in Fig. 6, right.

First of all, it can be seen that – regardless of the order – the amount of corrections is roughly growing linearly without big differences across the sequences.

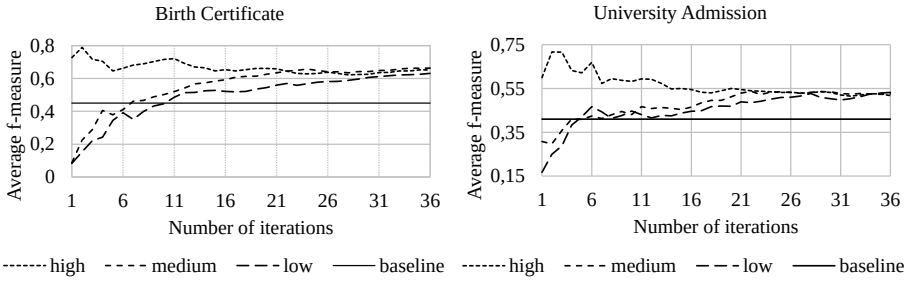

Fig. 5: Running average f-measure after i*th* iteration

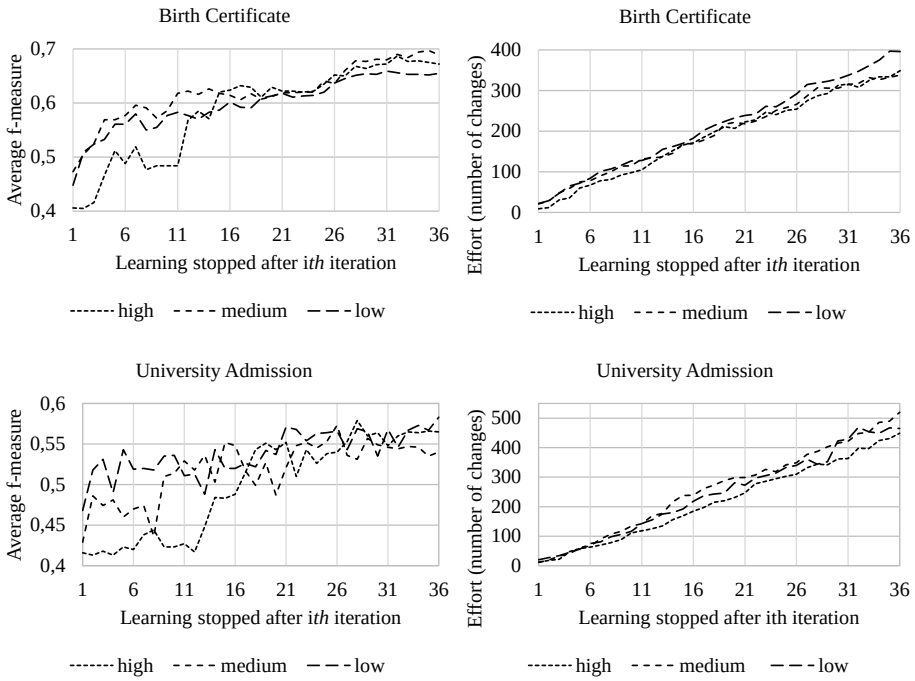

Fig. 6: Overall average f-measure over all 36 model pairs and the user workload after learning for i iterations

Furthermore, the f-measure curves for all three sequences approach each other with a growing number of iterations used to learn. When learning is stopped early, the best results are yielded for the *low* and the *medium* sequences: feedback on models has a larger impact if matching quality is low beforehand. Finally, regardless of the order, 2/3rds of the improvements are obtained from learning from about half the model pairs ($i = 16$). In practice it is not possible to sort model pairs with regard to the f-measure upfront. But as feedback collection and learning are progressing, the relative improvements can be measured. As soon as the improvements from additional feedback level off, learning can be stopped.

Discussion. The evaluation showed that the incorporation of user feedback led to strong improvements compared to the top matchers of the matching competition [3]. When feedback was collected for all model pairs, the f-measure increased by 41% and 53% for the two data sets. Even when reducing the workload by only collecting feedback for half of the model pairs, big improvements were obtained.

The major concern of experiments on process model matching relate to external validity, i.e. in how far the results of our study can be generalized [25]. In this regard, the size of the two data sets restricts the validity of both, the indicator assessment and the evaluation. Furthermore, the processes in both data sets represent the same abstract processes. Hence, some structural and behavioral characteristics might be underrepresented limiting the significance of the indicator assessment. This problem also has implications on the evaluation of

the word similarity adaptation, as the processes only cover a small number of tasks from both domains and a rather limited vocabulary used to describe them. Thus, words might tend to occur more often than in other model collections. Consequently, feedback might be collected for the same word pair more often than usual limiting the generalization of the quality improvements and strategies to minimize the user's efforts. Lastly, the indicator assessment does not allow for a general judgement on the sources of information, as there might exist other encodings of indicators which better exploit these characteristics. Therefore, enlarging the data sets by including data sets which characteristics differ from the once considered in this paper and considering more indicators are two important steps in future work.

6 Conclusions and Future Work

In this paper, we investigated user feedback as a mean for improving the quality of process model matching. Thus, we first reviewed indicators derived from related work and assessed their potential as information sources for feedback analysis. This assessment showed that, from the known sources of information, only the label based similarity of activities can reliably be applied to decide whether an activity pair corresponds or not. In a next step, we designed a mixed-initiative approach that adapts the word similarity scores based on user feedback. We evaluated our approach with regard to established benchmarking samples and showed that user feedback can substantially improve the matching quality. Furthermore, we investigated strategies to reduce the user workload while maximizing its benefit.

In future research, we plan to investigate further strategies for decreasing the user workload while maximizing the matching quality. This comprises guidelines for choosing model pairs (or activity pairs) the user needs to provide feedback on. Another direction we plan to pursue is the extension of our approach to better account for semantic relations and co-occurrences of words within labels.

References

1. Z. Bellahense, A. Bonifati, and E. Rahm. *Schema Matching and Mapping*. Springer, Heidelberg, 2011.
2. M. C. Branco, J. Troya, K. Czarnecki, J. M. Küster, and H. Völzer. Matching business process workflows across abstraction levels. In *MoDELS*, pages 626–641, 2012.
3. U. Cayoglu, R. Dijkman, M. Dumas, P. Fettke, L. García-Bañuelos, P. Hake, C. Klinkmüller, H. Leopold, A. Ludwig, P. Loos, J. Mendling, A. Oberweis, A. Schoknecht, E. Sheetrit, T. Thaler, M. Ullrich, I. Weber, and M. Weidlich. The process model matching contest 2013. In *PMC-MR*, 2013.
4. R. Dijkman, M. Dumas, and L. García-Bañuelos. Graph matching algorithms for business process model similarity search. In *BPM*, pages 48–63, 2009.
5. R. Dijkman, M. Dumas, B. van Dongen, R. Käärik, and J. Mendling. Similarity of business process models: Metrics and evaluation. *Inf. Syst.*, 36(2):498–516, 2011.

6. M. Dumas, L. García-Bañuelos, and R. M. Dijkman. Similarity search of business process models. *IEEE Data Eng. Bull.*, 32(3):23–28, 2009.
7. C. C. Ekanayake, M. Dumas, L. García-Bañuelos, M. L. Rosa, and A. H. M. ter Hofstede. Approximate clone detection in repositories of business process models. In *BPM*, pages 302–318, 2012.
8. J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer-Verlag, Berlin, 2013.
9. D. Grigori, J. C. Corrales, and M. Bouzeghoub. Behavioral Matchmaking for Service Retrieval. In *IEEE ICWS*, pages 145–152, 2006.
10. T. Jin, J. Wang, M. L. Rosa, A. H. ter Hofstede, and L. Wen. Efficient querying of large process model repositories. *Computers in Industry*, 64(1):41–49, 2013.
11. 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, 2013.
12. A. Koschmider and E. Blanchard. User assistance for business process model decomposition. In *IEEE RCIS*, pages 445–454, 2007.
13. M. Kunze, M. Weidlich, and M. Weske. Behavioral similarity - a proper metric. In *BPM*, pages 166–181, 2011.
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. V. I. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady.*, 10(8):707–710, 1966.
16. D. Lin. An information-theoretic definition of similarity. In *ICML*, pages 296–304, 1998.
17. F. J. Massey. The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951.
18. R. Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, 2009.
19. P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, 2005.
20. J. Vanhatalo, H. Völzer, and J. Koehler. The refined process structure tree. *Data Knowl. Eng.*, 68(9):793–818, 2009.
21. M. Weidlich, R. M. Dijkman, and J. Mendling. The ICoP framework: Identification of correspondences between process models. In *CAiSE*, pages 483–498, 2010.
22. M. Weidlich, J. Mendling, and M. Weske. Efficient consistency measurement based on behavioral profiles of process models. *IEEE Trans. Softw. Eng.*, 37(3):410–429, 2011.
23. M. Weidlich, T. Sagi, H. Leopold, A. Gal, and J. Mendling. Predicting the quality of process model matching. In *BPM*, pages 203–210, 2013.
24. M. Weidlich, E. Sheetrit, M. C. Branco, and A. Gal. Matching business process models using positional passage-based language models. In *ER*, pages 130–137, 2013.
25. C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, 2000.
26. H. Zha, J. Wang, L. Wen, C. Wang, and J. Sun. A workflow net similarity measure based on transition adjacency relations. *Computers in Industry*, 61(5):463 – 471, 2010.