Efficient Selection of Process Mining Algorithms

Jianmin Wang, Raymond K. Wong, Jianwei Ding, Qinlong Guo and Lijie Wen

Abstract—While many process mining algorithms have been proposed recently, there does not exist a widely-accepted benchmark to evaluate and compare these process mining algorithms. As a result, it can be difficult to choose a suitable process mining algorithm for a given enterprise or application domain. Some recent benchmark systems have been developed and proposed to address this issue. However, evaluating available process mining algorithms against a large set of business models (e.g., in a large enterprise) can be computationally expensive, tedious and time-consuming. This paper investigates a scalable solution that can evaluate, compare and rank these process mining algorithms efficiently, and hence proposes a novel framework that can efficiently select the process mining algorithms that are most suitable for a given model set. In particular, using our framework, only a portion of process models need empirical evaluation and others can be recommended directly via a regression model. As a further optimization, this paper also proposes a metric and technique to select high quality reference models to derive an effective regression model. Experiments using artificial and real datasets show that our approach is practical and outperforms the traditional approach.

Keywords—Business process mining, evaluation, benchmark.

1 Introduction

Process mining is a process management technique analyses business processes based on event logs. The basic idea is to extract knowledge from event logs recorded by an information system. This knowledge, in the form of business process models, can be extracted by process mining algorithms. These mined models can be used for the purpose of process compliance, and improving current business models. While many process mining algorithms have been proposed recently, there does not exist a widely-accepted benchmark to evaluate and compare these process mining algorithms. As a result, it can be difficult to choose a suitable process mining algorithm for a given enterprise or application domain.

For instance, process mining algorithms are used to mine business process models using process logs. The mined models will then be compared against the original process models of the enterprise for conformance checking, or to discover more efficient, streamlined business process models (which are still semantically equivalent to the original process models). However, since different process mining algorithms have different properties (e.g., some perform better on models with invisible tasks, while others do not), being able to select the most appropriate process mining algorithm (i.e. the one that produces mined models that are semantically similar to the original models and structurally equal to or better than the original models) for the models from a given enterprise is a big advantage.

While recent research and software prototypes have attempted to provide such an evaluation framework (e.g. [20, 28]), empirically evaluating all available process mining algorithms against the business models provided by a given enterprise is usually computationally expensive and time consuming.

Worst still, business process models are usually evolving. Therefore, these process model mining algorithms will need to be regularly re-evaluated against these changing models for conformance checking, re-engineering or discovery of more streamlined, improved models.

It is the primary aim of this paper to investigate a scalable solution that can evaluate, compare and rank these process mining algorithms efficiently. In particular, it attempts to investigate how we can choose an effective process mining algorithm for an enterprise without evaluating different process mining algorithms extensively, and also how to avoid re-evaluating all the algorithms whenever the business processes of the enterprise change.

More formally, consider a set of process models $M_1, M_2, ..., M_m$ for a given enterprise. Assume we use a process mining algorithm p to mine the corresponding process models $M_1^p, M_2^p, ..., M_m^p$ from the event logs of an enterprise. By analyzing the difference between M_i and M_i^p, we can discover any compliance issues, potential improvements to the original model M_i, or problems with the original model M_i. For example, M_i^p could be more streamlined (e.g., have fewer state transitions) than M_i. One way to measure the differences between the mined model M_i^p, with the original model M_i, is to use a similarity measure. Different similarity measures have been proposed in the process mining community for different measurements. For example, some are more focused on structural measurements (i.e., how similar are the two models based on the graph structure); while others are based on the behavioral similarity (i.e., mea-

- *J. Wang, J. Ding, Q. Guo and L. Wen are with School of Software, Tsinghua University, Beijing 100084, China*
  E-mail: jimwang@tsinghua.edu.cn
- *R.K. Wong is with School of Computer Science and Engineering, University of New South Wales, Sydney, NSW 2012, Australia*
  E-mail: wong@cse.unsw.edu.au

suring the similarity based on the behavioral semantics, because process models define a part of the behavior of an organization). A good overview of structural and behavioral similarities can be found in [7, 9].

Furthermore, depending on the complexity and types of constructs of the process models to be mined, different process mining algorithms have different performance and characteristics (as shown in the motivating examples in Section 3). Therefore, traditionally, using process evaluation frameworks such as [20, 28, 29] and given a set of available process mining algorithms $p_1, p_2, ..., p_q$, we need to run each of the algorithms, p_j, on the event logs of a given organization to produce $M_1^{p_j}, M_2^{p_j}, ..., M_m^{p_j}$, in order to determine which mining algorithm is the most appropriate for the organization, i.e., produce the best score by a similarity measure. If multiple similarity measures are considered, different similarity measures between $M_i^{p_j}$ and M_i for all $i \in [1..m]$ and for each algorithm p_j (for $j \in [1..q]$) need to be calculated. This whole process can be computationally very expensive, as it involves $m \times q$ computations of mining a process model.

In our proposed approach, we select a portion of the models, namely $M_1, M_2, ..., M_n$ where $n < m$, to evaluate and to construct a regression model such that it can be used to estimate the similarities of other models $M_{n+1}, ..., M_m$ without any empirical evaluation. We also propose a metric and a selection method to select models that are effective to differentiate the performance and capabilities of the given process mining algorithms. As a result, our approach only involves $n \times q$, where $n < m$, computations of mining a process model. Furthermore, we can use the regression model directly to estimate the similarities without any empirical evaluation when the process models of an organization have been changed but not significantly. In this case, the entire benchmarking process can be saved. Even if there is a major change, in the worst case, evaluating n models instead of m models is still a significant speedup, as shown in the experiments of this paper.

In summary, this paper will firstly summarize our previous efforts in the empirical evaluation of process mining algorithms using reference models [28]. In this approach, the quality of a discovered model is measured by the behavioral and structural similarities with its reference model. Although this approach provides a reasonably accurate benchmark for process mining algorithms, it takes a long time to run, even for process models from a medium-size enterprise. This is because it needs to evaluate each process mining algorithm on each process model of an organization.

We then propose to only use a fraction of the process models from the given enterprise as reference models. Next, we apply regression analysis (or most of the common supervised learning methods) to obtain two trained models - one for behavioral measure and the other one for structural measure. We can then apply these trained models to any process models from the enterprise and obtain the estimated similarities for the available mining algorithms. Hence, we do not need to empirically evaluate the process mining algorithms on each process model in order to determine the most suitable one for the enterprise.

To achieve good accuracy, we have considered an excessive number of parameters (48 features as listed in Appendix A) extracted from the process models. We extend the features defined in [15] to 48 features so that we have more than enough features that (hopefully) capture all important characteristics of these models. We can then find out a significant subset of these features empirically, using a dimension reduction technique. Therefore, for efficiency, we apply a dimension reduction technique to reduce these features to just enough parameters to appropriately rank the given mining algorithms.

Furthermore, as an optimization step and to ensure good quality of our approach, we propose a novel metric to select reference models with properties that can clearly differentiate the performance of the available process mining algorithms (a.k.a. significant reference models). For example, most mining algorithms will have a very high similarity score for simple process models. However these simple process models are not very useful as reference models for our training purpose, as they cannot be used to differentiate the performance of the algorithms. With this metric, only significant reference models will be used for training. This optimization step will enhance both the accuracy and efficiency during the training phase of our approach. Moreover, when the models of an organization change, we can use this optimization step to detect if those previously selected, significant reference models have been majorly changed. This can be used to determine if a new training is necessary when there may be too many changes made to the process models.

Next, we apply these two trained regression models to real-life processes obtained from a major boiler manufacturer and a high-speed train manufacturer in China. The estimated similarities and runtime are recorded. Based on the estimated similarities, our framework ranks the business process mining algorithms and recommends the algorithm with the highest similarity for each of these two enterprises. Finally, to evaluate the speed improvement and accuracy, the results obtained from the trained regression models will be compared with those obtained from the empirical evaluation.

The organization of the rest of this paper is as follows. Section 2 discusses the related work. Section 3 lists two example process models from real-life datasets to illustrate the problem that we are addressing in this paper. Section 4 provides an overview of our proposed framework, while Section 5 describes the training / learning phase of our framework. Section 6 presents the optimization step that is based on a metric for selecting significant reference models. Section 7 presents the experiments and discusses the results. Finally, Section 8 concludes the paper, and Appendix A lists the 48 features of process

models that are used by our framework.

2 Related Work

Many process mining algorithms have been proposed to date, e.g. [24, 31]. However, as highlighted by [11], a widely-accepted benchmark to evaluate and compare different process mining algorithms still has not been identified. In particular, creating representative benchmarks is still a challenge for the process mining community [11]. For example, there are many process discovery techniques and different vendors offer different types of products. However, there is no consensus on the quality of these techniques.

The term process mining refers to methods for distilling a structured process description from a set of real executions [23]. As pointed out by [23], the challenge of workflow mining is to derive *good* workflow models with as little information as possible. An important and natural criterion for the quality of a mined workflow model is the consistency between the mined model and the traces in the workflow log.

Quality criteria (including: fitness, simplicity, precision and generalization), which are based on event logs, have been widely discussed in the community. However, striking a balance between these quality criterion is a major challenge [11]. This is becuase these four dimensions are competing with each other. Furthermore, event logs are often very large and diificult to understand.

There has been research and discussions on the quality measurement of business process models and what contributes to "good" process models (e.g., [14, 17, 26]). For the purposes of our paper, we are more interested in how good the business process models are, in order to differentiate the performance of the given business process mining algorithms. For example, some business process models (including many models from the WF-Nets [22] of the SAP reference model dataset [16, 18]) are too simple to differentiate the performance of those commonly used business process mining algorithms. Therefore, in this paper, we will examine and define the criteria on how to select good models to analyze these mining algorithms.

To the best of our knowledge, the first effort towards a common evaluation framework for process mining algorithms was done by Rozinat et al. [20, 21]. Their evaluation framework consists of four components, namely, event log and reference model repository; log generation module; modification and verification tools; evaluation and comparison module. Their proposal focuses on the compliance evaluation between the event log and the mined process model. Most importantly, as pointed out by the authors in [20], a comprehensive set of benchmarks (ideally containing both artificial and real-life datasets) is still lacking.

This paper adapts and extends the concept from [20] by empirically evaluating process mining algorithms based on the behavioral similarity and the structural similarity between reference process models (user knowledge) and mined models (discovered knowledge). Therefore, throughout this paper, we assume that we know the business process model that is used to generate the process event log.

While various similarity measurements for business process models have been proposed recently, e.g., [8, 9, 32], in this paper, we adopt the approach from [1] for structural similarity measure. This is due to its good performance in determining the relative distance between the original models and mined models. Behavioral similarity is measured by the Principle Transition Sequence (PTS) metric [27], since it can deal with many complex structures, such as none-free choice, arbitrary loop, even the unbounded Petri net, etc, and can be calculated efficiently, as mentioned in [2]. However, our evaluation framework can easily be incorporated into other measurement schemes, to suit any specific requirements.

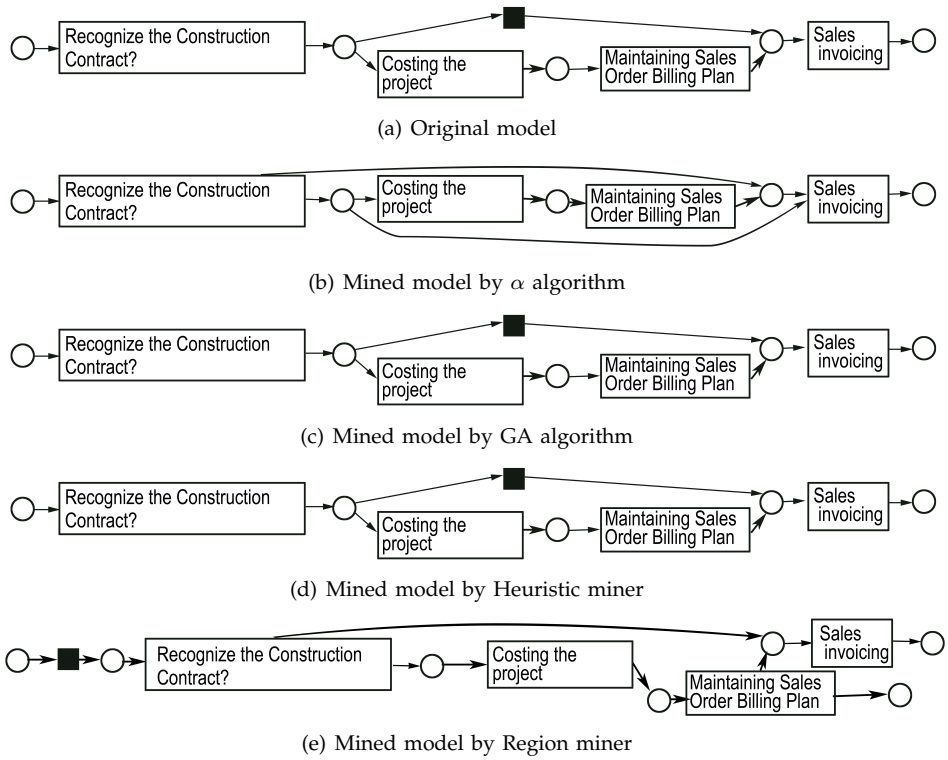
The process mining algorithms being evaluated in this paper are: α algorithm [24], genetic algorithm [5], heuristics miner [30] and region miner [4]. Without loss of generality, we believe that these four algorithms represent the four most popular classes of business process mining algorithms.

Finally, our idea of using regression to estimate the process mining algorithms on business process models was first reported in [29]. During the training phase in [29], a random number (instead of all models) of process models have been selected and used. The notion of significant reference models, which will clearly affect the effectiveness of the chosen reference models for training, has not been investigated. This results in uncertainty in terms of the quality of the proposed approach, i.e., unstable accurarcy and performance. For example, there is no way to detect and select good reference models for training out of a huge variety in the process models (such that two process models differ significantly). Furthermore, without the mechanism to select significant reference models, the proposal in [29] cannot cope with changes in process models. For instance, it will not be able to determine when and whether a new training phase is required, when there may be too many changes of the models such that the regression model for recommendation may not longer be valid.

3 Motivating Examples

Many business process mining algorithms have been proposed. However, each of them have different performance for different business processes. This section illustrates this situation by drawing examples from two real-life datasets (one from a major boiler manufacturer in China, and one from a high-speed train project in China).

For example, consider the model SD050 in Figure 1, which we have extracted from a major boiler manufacturer in China. SD050 is a sales invoicing subprocess in



(a) Original model

(b) Mined model by α algorithm

(c) Mined model by GA algorithm

(d) Mined model by Heuristic miner

(e) Mined model by Region miner

Fig. 1. Process model SD050 and its mined models

the sales module of the company. After benchmarking this model against the four process mining algorithms using the framework proposed in [28], both their behavioral and structural similarities are shown in Table 1. Table 1 shows that Genetic and Heuristic algorithms produce models with excellent structural similarity, while Alpha and Region algorithms produce models with very poor structural similarity. On the other hand, Alpha algorithm produces a model with reasonable behavioral similarity, while all three other algorithms produce models with excellent behavioral similarity. All of these can be easily observed in Figure 1.

TABLE 1
Similarities between SD050 and its mined models

Similarity	Alpha	Genetic	Heuristic	Region
Structural	0.1800	1	1	0.1000
Behavioral	0.7500	1	1	1

Consider another example: process model MM240 extracted from a high-speed train dataset as shown in Figure 2. MM240 is a material management process in the material supply module of the company. After benchmarking this model against the four process mining algorithms, both their behavioral and structural similarities are shown in Table 2. Table 2 shows that Alpha and Heuristic algorithms produce models with excellent structural similarity, while Genetic algorithm produces a model with reasonably good structural similarity. However, after running the region algorithm for 30mins on BeehiveZ [12] using a machine with 2GB RAM (1GB Java heap size), Intel CPU 2.53GHz, no result was produced (hence both structural and behavioral similarities cannot be measured for the Region algorithm). On the other hand, other than the Region algorithm, all other algorithms produce models with excellent behavioral similarity for MM240. All of these can be observed in Figure 2.

TABLE 2
Similarities between MM240 and its mined models

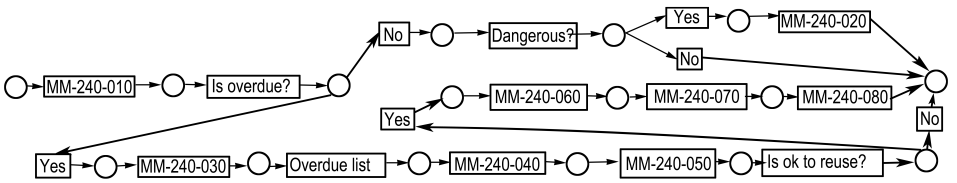
Similarity	Alpha	Genetic	Heuristic	Region
Structural	1	0.8600	1	NA
Behavioral	1	1	1	NA

The above two examples (based on real-life datasets) have illustrated that different process mining algorithms perform differently on different models. As a result, enterprises need to benchmark different process mining algorithms to select appropriate algorithms for their models. However, as shown in a later section of this paper, benchmarking these algorithms against the existing models of a given enterprise (especially for a large enterprise) can be computationally expensive and take a long time.

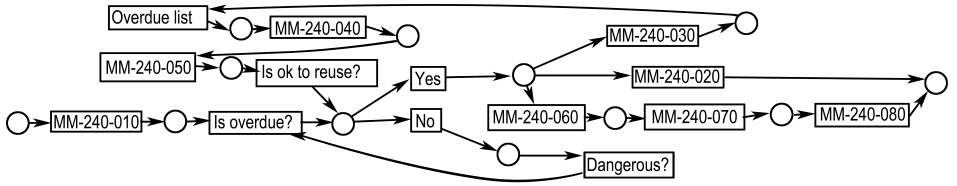
4 Overview of The Framework

4.1 Framework Components

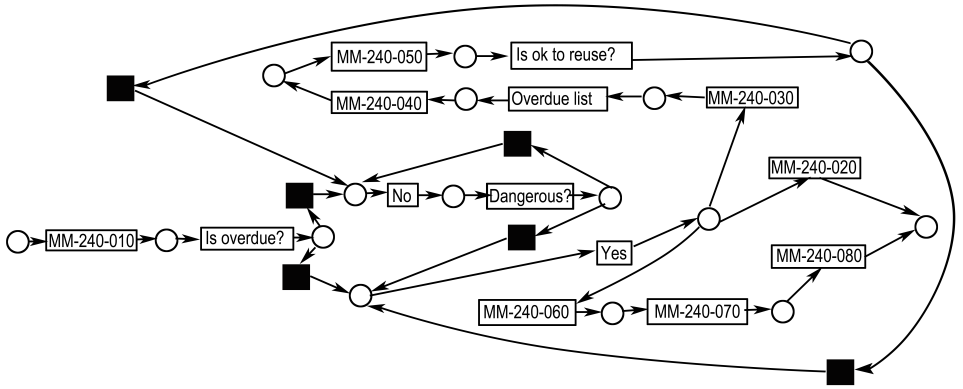
Our framework is built by extending the work from Rozinat et al. [20]. The overall architecture of the framework is shown in Figure 3.



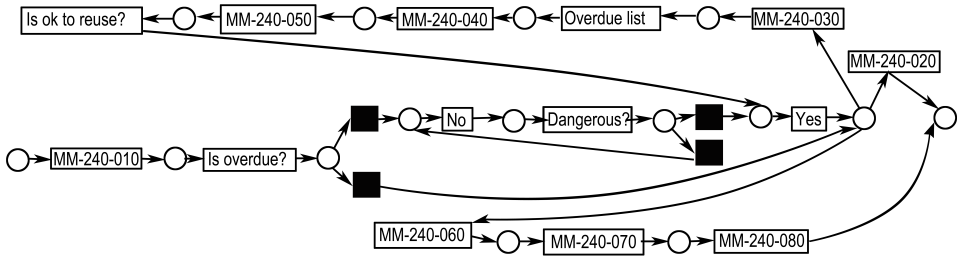
(a) Original model



(b) Mined model by α algorithm



(c) Mined model by the genetic algorithm



(d) Mined model by the heuristics algorithm

Fig. 2. Process model MM240 and its mined models

In Figure 3, the original work from [20] assumes the original reference models or the process log (either generated from the original models, or extracted from the enterprise log, which conforms to these models). The process mining algorithms can then be run on the log to produce the mined models. Next, the similarity measure can then be performed on the mined models and the original models. This is the central idea behind [20].

Our previous work ([28, 29]) empirically benchmarks various process mining algorithms using structural and behavioral similarities. Although the results are good, the approach may take a long time to compute, especially when the model set is large and complex (since it needs to evaluate each of the process mining algorithms that are available to the organization on each process model).

As shown in Figure 3, we extend the idea in [20] to a full framework that includes a learning/training phase and a recommendation phase.

During the learning phase, the feature extraction module extracts key features (as listed in the Appendix A of this paper), to build a model that captures the similarity measure. Since we do not know which features affect the performance of the individual mining algorithms, an unnecessary number of features may have been extracted. Feature reduction (i.e. dimension reduction) is required to analyze these extracted features and reduce them to only those that will significantly affect our final results. Finally, during the learning phase, regression analysis produces a regression model that captures the correlation

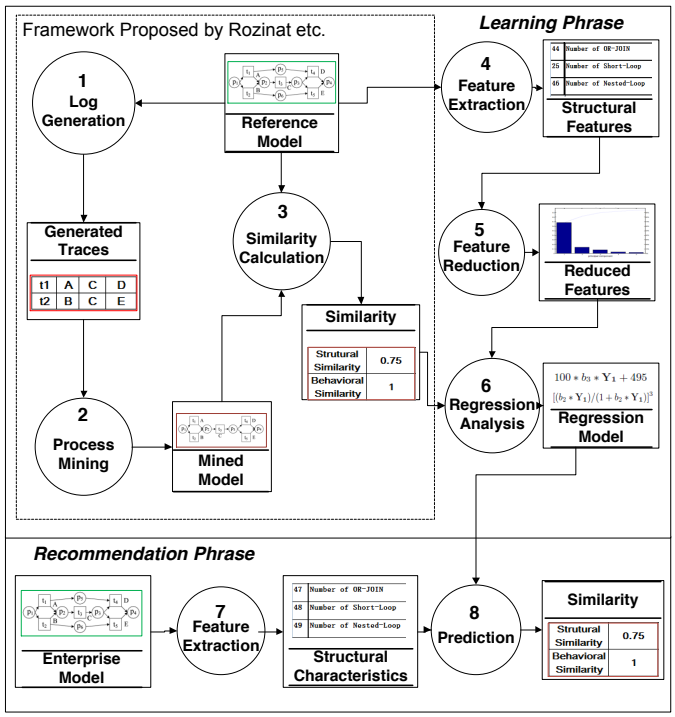


Fig. 3. Overall architecture of our framework

between the reduced features and the similarity values.

After the regression model is obtained, during the recommendation phase, we can recommend the most suitable (i.e. best performing) mining algorithm without performing the actual empirical benchmarking. Since applying the regression model to the model features to obtain the estimated similarities is very fast, the only cost during the recommendation phase is feature extraction.

Most importantly, rather than using any given models as reference models (which may contain models that cannot clearly differentiate the performance of the mining algorithms, e.g., models that are too small and simple), we propose a new metric for selecting models that are good enough to differentiate the performance of the given mining algorithms based on similarity measure. Therefore, the Reference Model used in Figure 3 will only contain models that are useful in the learning (which will enhance both the accuracy and speed of our framework). This is quite different from the original idea in [20].

The whole framework has been implemented in our prototype system: BeehiveZ [12], which is publicly available[1]. Note that while the prototype system itself supports many different process mining algorithms, for clarity and conciseness, we have chosen to benchmark the most common 4 classes of process mining algorithms in this paper (namely, Alpha, Genetic, Heuristics and Region). Our prototype system also integrates the process mining plugins and model transformation support of ProM [25].

1. http://code.google.com/p/beehivez/

4.2 Similarity metrics

The similarity between the reference model and the mined model can be used to measure the quality of a process mining algorithm in most cases (except when the reference model has some defects leading to malformed event logs). The reference models are selected as exemplary sets. So, these models can be used as 'good examples'. We measure the process mining algorithm based on both structural and behavioral aspects.

In our framework, we use labeled Petri net [6] as the modeling language. In some sense, labeled Petri nets can be seen as a foundation theory for process modeling. Structural similarity can be measured by the notion of dependency graph as described in [1]. In our system, the incidence matrix of labeled Petri net is used as the Process Matrix of a dependency graph. We calculate the trace of the dependency difference matrix between the two process models as the distance (d). Then we use the formula, $1/(1 + d)$, to calculate the structural similarity of the two process models.

Behavioral similarity is measured by the Principle Transition Sequence (PTS) metric [27]. The notion of *coverability tree* is one of the fundamental methods for behavioral analysis of Petri nets [19]. As the set of traces of a labeled Petri net may be infinite, PTS can provide a good behavior approximation of the essence of a process model. PTS metric can deal with many complex structures, such as none-free choice, arbitrary loop, even the unbounded Petri net, etc, and can be calculated efficiently, as mentioned in [2]. Therefore, we chose PTS as the behavioral similarity measurement in this paper.

5 The Learning Phase

5.1 Feature Extraction

We extend the features defined in [15] to 48 features that characterize the business process models. These features include important information such as the number of transitions, number of places, number of and-joins, number of and-splits, number of xor-joins, etc.; and other information such as edge density, number of invisible tasks of the model, etc. Augmented with examples, all these 48 features are listed in Appendix A of this paper.

5.2 Feature Reduction

Since some of the 48 chosen features may be unnecessary or contain information that is not very useful towards process mining algorithm selection, it is desirable (both in terms of speed and accuracy) to apply some dimension reduction technique to reduce the data dimension for regression. There will be correlation between the features (since we choose many features - 48 features!). In this paper, we use PCA (Principal Component Analysis) [13] to produce several principal components that are linearly independent to each other. By using PCA, we can then identify which features are more significant

to determine the performance of process mining algorithms.

First of all, we employ the concept of *entropy* to measure the amount of information. Suppose we represent the features using a random variable vector $\vec{x} = (x_1, x_2, ..., x_p)^T$ where $p = 48$ in our case, and the given process model set as a matrix $\mathbf{X}$ of $m * p$ dimensions, where m is the number of process models in the model set.

Definition 1: (Entropy)[3] The amount of information of random variable vector $\vec{x} = (x_1, x_2, ..., x_p)^T$ can be obtained by:

$$\mathcal{H}[\vec{x}] = -\sum_{\vec{x}} p(\vec{x}) \log(p(\vec{x}))$$

where $p(\vec{x})$ is the probability distribution of $\vec{x}$.

Definition 2: (Principal Component Analysis) A linear transformation called PCA is defined as

$$\vec{x'} = PCA(\vec{x}) = \mathbf{A} * \vec{x}$$

where $\vec{x'}$ is a new vector with dimension $p * 1$, and $\mathbf{A}$ is a transformation matrix with dimension $p * p$, where

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1p} \\ a_{21} & a_{22} & \dots & a_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ a_{p1} & a_{p2} & \dots & a_{pp} \end{bmatrix}$$

(1) If $x'_1 = a_{11}x_1 + a_{12}x_2 + ... + a_{1p}x_p$, and we arrange the columns of the matrix such that the variance of x'_1 $Var(x'_1)$ is the largest. Then we call x'_1 the First Principal Component;
(2) If $x'_2 = a_{21}x_1 + a_{22}x_2 + ... + a_{2p}x_p$ and $(a_{21}, a_{22}, ..., a_{2p})$ is orthogonal to $(a_{11}, a_{12}, ..., a_{1p})$, and we arrange the columns of the matrix such that $Var(x'_2)$ is the second largest. Then we call x'_2 the Second Principal Component;
(3) Similarly, we get the Third Principal Component, the Fourth Principal Component, etc. We can also get the p-th Principal Component easily.

From the Definitions 1 and 2, we have two properties:
Property 1: $Var(x'_1) > Var(x'_2) > ... > Var(x'_p)$
Property 2: $\mathcal{H}[\vec{x'}] = \mathcal{H}[\vec{x}]$
Proof of Property 1: Trivial from Definition 2.
Proof of Property 2:

$$\begin{aligned} \mathcal{H}[\vec{x'}] &= -\sum_{\vec{x'}} p(\vec{x'}) \log(p(\vec{x'})) \\ \mathcal{H}[\vec{x'}] &= -\sum_{\vec{x}} p(\mathbf{A} * \vec{x}) \log(p(\mathbf{A} * \vec{x})) \end{aligned}$$

The transformation matrix $\mathbf{A}$ is an orthogonal matrix, i.e., its inverse matrix $\mathbf{A}^{-1}$ of $\mathbf{A}$ exists and $det(\mathbf{A}^{-1})$ equals to 1 or -1. Therefore, $p(\mathbf{A} * \vec{x}) = p(\vec{x}) * det(\mathbf{A}^{-1})$. As a result:

$$\begin{aligned} \mathcal{H}[\vec{x'}] &= -\sum_{\vec{x}} p(\vec{x}) * |det(\mathbf{A}^{-1})| \log(p(\vec{x}) * |det(\mathbf{A}^{-1})|) \\ \mathcal{H}[\vec{x'}] &= -\sum_{\vec{x}} p(\vec{x}) \log(p(\vec{x})) = \mathcal{H}[\vec{x}] \end{aligned}$$

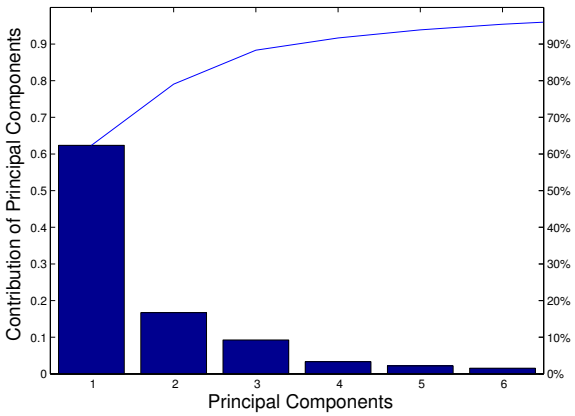


Fig. 4. The effect of PCA to the artificial dataset

□

By probability, we know that the model set $\mathbf{X}$ is a subset of sample space of the random variable vector $\vec{x}$. We can also define the *entropy* of the model set $\mathbf{X}$ and analyze its principal components. $\mathbf{X}$ can be represented as $\mathbf{X} = (\vec{x_1}, \vec{x_2}, ..., \vec{x_m})^T$. We will then get $\mathbf{X}' = PCA(\mathbf{X})$ and $\mathcal{H}[\mathbf{X}] = \mathcal{H}[\mathbf{X}']$. So we can have a new representation for the model set $\mathbf{X}'$ with the same dimensions, where $\mathbf{X}' = (\vec{x'_1}, \vec{x'_2}, ..., \vec{x'_m})^T$. We will then have:

$$\begin{aligned} \mathcal{H}[\mathbf{X}'] &= -\sum_{i=1}^{m}\sum_{j=1}^{p} p(x'_{i,j}) \log(p(x'_{i,j})) \\ &= -\sum_{j=1}^{p}\sum_{i=1}^{m} p(x'_{i,j}) \log(p(x'_{i,j})) \\ &= \sum_{j=1}^{p} \mathbf{H}[PC_j] \end{aligned}$$

where principal component PC_j is the j^{th} column of $\mathbf{X}'$. With PCA, we can reduce the features drastically, and only lose a small amount of information. For example, when applying PCA to 90 models from the artificial dataset used in this paper, even just choosing 6 principal components would already contain more than 99% of the information from the original 90 models, as illustrated in Figure 4.

From Figure 4, we also notice that the first principal component, namely PC1, contains most of the information. Figure 5 shows the relations between the coefficients of the 1st principal component and the model features. The X-axis indicates the features and the Y-axis is the coefficient value of PC1. From PC1, we can observe that the features *number of transitions, number of places, number of arcs, number of tars, number of state in state-space* (those long bars in the figure) are more significant than the others.

5.3 Regression Analysis

With PCA, now we have a matrix of the model set, say $\mathbf{X}''$, of dimension $m * q$ smaller than that of the

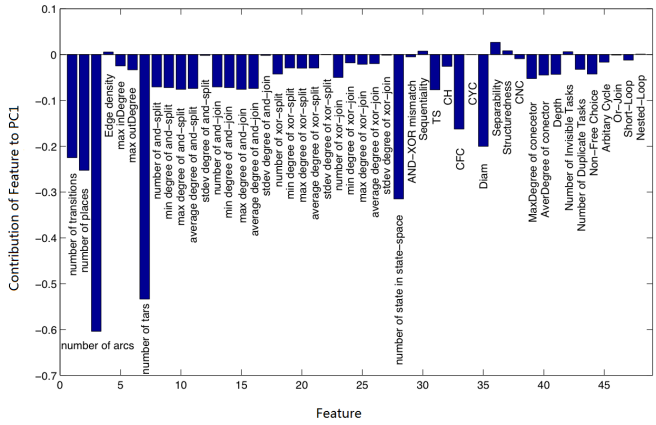


Fig. 5. Relationship between the coefficients of PC1 and the model features

original model set (of dimension $m*p$) to consider during regression analysis, where $q \leq p$. i.e., We will have $\mathbf{X''}$ as independent variable and the similarity score as dependent variable for regression. Let $x''_1, x''_2, ..., x''_q$ be the principal components and y be the dependent variable. As mentioned earlier, we chose regression to capture the correlation between the models and similarity scores in this paper, but we believe that most supervised learning techniques would work well too.

Now we have m given process models. Each model can be represented as a linear equation formed by its independent and dependent variables. i.e.,

$$y = a_1 x''_1 + a_2 x''_2 + ... + a_q x''_q + a_{q+1} \quad (1)$$

where the extra term a_{q+1} is an error term. The purpose of our regression analysis is to compute the coefficients $a_1, a_2, ..., a_q$. According to the generalized least squares(GLS) we can transform Formula 1 into:

$$z = \sum_{k=1}^{m} (z_k - (a_1 x''_{k1} + a_2 x''_{k2} + ... + a_q x''_{kq} + a_{q+1}))^2 \quad (2)$$

such that we need to determine $a_1, a_2, ..., a_q$ to minimize z. Hence, we need to compute the partial derivatives of $a_1, a_2, ..., a_q$ in Formula 2, i.e.,

$$(\mathbf{CC^T}) \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_q \end{bmatrix} = \mathbf{C} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_q \end{bmatrix}$$

where matrix $\mathbf{C}$ satisfies:

$$\mathbf{C} = \begin{bmatrix} x''_{11} & x''_{21} & \dots & x''_{m1} \\ x''_{12} & x''_{22} & \dots & x''_{m2} \\ \vdots & \vdots & \vdots & \vdots \\ x''_{1q} & x''_{2q} & \dots & x''_{mq} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Finally, by Cholesky decomposition[10], we can derive the regression coefficients $a_1, a_2, ..., a_q$. As a result, we have obtained a regression model for our approach.

6 Further Optimization: Reference Model Selection

As described previously, the given reference model set may contain models that are useful to differentiate the performance of the process mining algorithms. This section presents a new metric that defines which reference models are significant enough to be used to differentiate the mining algorithm performance.

Since we will deal with each of these two types of similarities (structural and behavioral) in the same way, for the rest of this section, we assume structural similarity for conciseness. We define a similarity matrix ss of dimension $m * 4$ (since there are m models and 4 similarity scores that correspond to 4 mining algorithms) as:

$$\mathbf{ss} = \begin{bmatrix} ss_{11} & ss_{12} & ss_{13} & ss_{14} \\ ss_{21} & xs_{22} & ss_{23} & ss_{24} \\ \vdots & \vdots & \vdots & \vdots \\ ss_{m1} & ss_{m2} & ss_{m3} & ss_{m4} \end{bmatrix}$$

6.1 Significant Model Determination

We define two rankings, namely variance ranking and distance ranking, to help determining if a reference model is *significant*.

Definition 3: (Variance Ranking) We rank the rows of the similarity matrix ss in decreasing order according to the variances of the values in each row, and call this new matrix with ranked rows ss′.

Each row in the similarity matrix ss contains four similarity scores obtained from the four mining algorithms. We re-arrange these scores such that they are sorted in decreasing order and form a new similarity matrix ss′:

$$\mathbf{ss'} = \begin{bmatrix} ss'_{11} & ss'_{12} & ss'_{13} & ss'_{14} \\ ss'_{21} & xs'_{22} & ss'_{23} & ss'_{24} \\ \vdots & \vdots & \vdots & \vdots \\ ss'_{m1} & ss'_{m2} & ss'_{m3} & ss'_{m4} \end{bmatrix}$$

Note that $ss'_{i1} \geq ss'_{i2} \geq ss'_{i3} \geq ss'_{i4}$. We can then calculate three distances $dis_{i1} = |ss'_{i1} - ss'_{i2}|$,$dis_{i2} = |ss'_{i2} - ss'_{i3}|$ and $dis_{i3} = |ss'_{i3} - ss'_{i4}|$; and also rank the three distances dis_{i1} dis_{i2} dis_{i3} by their values in increasing order. Next, we define a distance matrix **dis** such that the i_{th} row of the distance matrix **dis** is dis'_{i1},dis'_{i2} and dis'_{i3}, where $dis'_{i3} \geq dis'_{i2} \geq dis'_{i1}$.

Definition 4: (Distance Ranking) In the i_{th} row of the distance matrix, we call **dis**, dis'_{i1},dis'_{i2} and dis'_{i3} the *First Distance*, the *Second Distance* and the *Third Distance* of the similarity matrix ss, respectively. We can then rank the rows of the similarity matrix ss in decreasing order firstly by their *First Distance*, then by the *Second Distance* and finally by the *Third Distance*. We also select the top x number of rows as significant models for each of these 3 rankings (as a result, the significant models will be more selective after each of the above 3 rankings.)

With the two rankings, we can now determine which reference models are significant. Firstly the rows of the

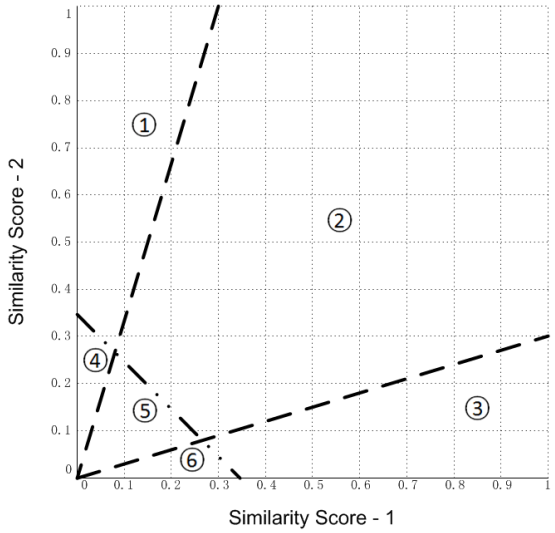


Fig. 6. A two-dimensional similarity space (i.e., both X- and Y-axes are similarity scores).

similarity matrix ss are ranked by *Variance Ranking*. Then we choose the top x number of rows as significant reference models, because the variances of their similarity scores are larger than the rest of the models, i.e., they have larger variation of similarity scores to differentiate the mining algorithms. The amount of x can be decided according to the number of models and degree of optimization desired by the enterprise. For example, if speed is more important than accuracy, then a smaller x is preferred. Without loss of generality, x can also be decided or specified as a percentage of the models to be kept as significant models.

Next, we get a new similarity matrix $\mathbf{ss_1}$ containing only those significant reference models, i.e., its dimension is $x * 4$. We then rank the rows of $\mathbf{ss_1}$ using *Distance Ranking* three times, and each time we keep only a smaller number of rows from the top as significant reference models. These final, chosen models will be more significant (with more distinct, differentiating similarity scores) than those models that have not been chosen. Hence, they are more effective to be used as reference model for training purpose.

To illustrate these 2 rankings, consider the two dimensional similarity space (i.e., consider 2 mining algorithms instead of 4) as shown in Figure 6. In Figure 6, Areas 4, 5 and 6 contain models that have not been selected as significant models after *Variance Ranking*, because these areas contain models with similarities of smaller variance than those in Area 1, 2 and 3. With *Distance Ranking*, areas 1 and 3 contain models more significant than those in area 2 because of their more distinct similarity scores.

6.2 Classification of Significant Reference Models

Now we can determine if a reference model is significant or not using the two rankings described above. To speed up the determination of reference models, we further propose using SVM (support vector machine) to learn classifying the significant reference models from the insignificant ones. To further optimize the SVM time, we consider the reduced features after PCA, instead of the original 48 features for each process model.

6.2.1 Classification in Artificial Models

We choose 90 models from the artificial dataset to train the SVM and 90 models to test its accuracy. We set the x for *Variance Ranking* and *Distance Ranking* to 65%. The number of significant models determined is 17.

For structural similarity, the SVM training time is 23.9 seconds, and the accuracy is 82.22%. For behavioral similarity, the training time is 42.5 seconds and the accuracy is 72.22%.

6.2.2 Classification in Boiler Models

We choose 36 models from the boiler dataset to train the SVM and 36 models to test its accuracy. We set the x for *Variance Ranking* and *Distance Ranking* to 85%. The number of significant models determined is 18.

For structural similarity, the SVM training time is 14.2 seconds, and the accuracy is 72.22%. For behavioral similarity, the training time is 9.8 seconds and the accuracy is 83.33%.

6.2.3 Classification in High-speed Train Models

We chose 81 models from the high-speed train dataset to train the SVM and 81 models to test its accuracy. We set the x for *Variance Ranking* and *Distance Ranking* to 55%. The number of significant models determined is 8.

For structural similarity, the SVM training time is 18.4 seconds, and the accuracy is 70.37%. For behavioral similarity, the training time is 18.5 seconds and the accuracy is 74.07%.

7 The Experiments

Experiments were performed on BeehiveZ using a machine with 2GB RAM (1GB Java heap size), Intel CPU 2.53GHz. We collected both artificial and real-life datasets for benchmarking purpose. Real-life data are valuable in evaluating and benchmarking process mining algorithms, as different process mining algorithms can perform very differently with different applications and requirements. For example, α algorithm discovers all SWF-nets only if complete event logs are presented [24]. It also has limitations on discovering processes beyond SWF-nets. However, the algorithm is still good enough for an organization whose process models are mostly structural. Therefore, it is always desirable to consider real-life datasets (especially the datasets from the same organization, or at least from the same industry) when evaluating process mining algorithms for an organization.

TABLE 3
Dataset Properties

Dataset	Size	Average			Minimum			Maximum		
		#transitions	#places	#arcs	#transitions	#places	#arcs	#transitions	#places	#arcs
Artificial	270	4.038889	4.1	11.37222	1	1	1	13	14	30
Boiler	108	7.375	7.291667	15.08333	3	3	6	14	11	28
Trains	243	15.91975	14.54321	32.53416	5	5	10	35	32	70

7.1 The Datasets

As noted in [11], while many process mining algorithms have been proposed recently, there exists no widely-accepted benchmark to evaluate these process mining algorithms. As a result, it can be difficult to compare different process mining algorithms especially over different application domains. This section describes the empirical benchmarking component (as described in [28]) for evaluating business process mining algorithms that we have added to our previously built Business Process Model and Instance Management System called BeehiveZ.

To construct a comprehensive benchmark for process mining algorithm evaluation, both artificial and real-life reference models are included in our empirical study. The basic properties of these models are summarized in Table 3.

To determine which business process mining algorithm is the most appropriate for the given dataset, one can perform empirical benchmarking on each of the available algorithms using the models from this dataset. The following subsections briefly introduce the sources of the datasets and summarizes the benchmark results.

7.1.0.1 The Artificial Dataset: Artificial process models (total 270 models) are collected from other papers, the SAP reference models, and also manually created.

7.1.0.2 The Boiler Manufacturer Dataset: We collected 108 real-life process models from Dongfang Boiler Group Co., Ltd. Dongfang Boiler is a major supplier of thermal power equipment, nuclear power equipment, power plant auxiliaries, chemical vessels, environmental protection equipment and coal gasification equipment in China (with more than 33% market share).

7.1.0.3 The High Speed Trains Dataset: We collected 243 real-life process models from Tangshan Railway Vehicle Co., Ltd. (TRC). TRC is the oldest enterprise in China's railway transportation equipments manufacturing industry, with its predecessor constructing China's first railway line from Tangshan to Xugezhuang in 1881 during the westernization movement at the end of the Qing Dynasty. The company has also built a train that has achieved the highest speed of Chinese railways - 394.3km/h.

7.2 Empirical Evaluation

Figure 7 shows the benchmark results (as histograms) of both the structural and behavioral similarities between the mined models and the original models for the 3 datasets using the empirical evaluation framework described in [28]. In each sub-figure of Figure 7, the X-axis is the similarity score (between 0 and 1) and Y-axis is the number of models that have the same similarity score. We can clearly observe that some algorithms perform better on some models but poorly on some others. For example, Genetic produced more models with good structural similarities for the artificial and boilder datasets than the other 3 mining algorithms, and Alpha produced more models with good structural similarities for the trains dataset.

TABLE 4
Feature extraction cost (in seconds)

Dataset	Size	Total time
Artificial	270	1.9
Boiler	108	0.8
Trains	243	4.8

TABLE 5
Mining time taken (in seconds). It includes the time taken by components 1, 2 and 3 from Figure 3.

Dataset	Size	Alpha	Genetic	Heuristic	Region
Artificial	270	166	3480	174	895
Boiler	108	161	1625	157	291
Trains	243	393	13248	409	307795

7.2.1 Feature Extraction Cost

Table 4 shows the time taken by feature extraction (extracting the 48 features) for each dataset. The time taken is very short, with the high-speed train dataset taking the longest time due to its larger and more complex models.

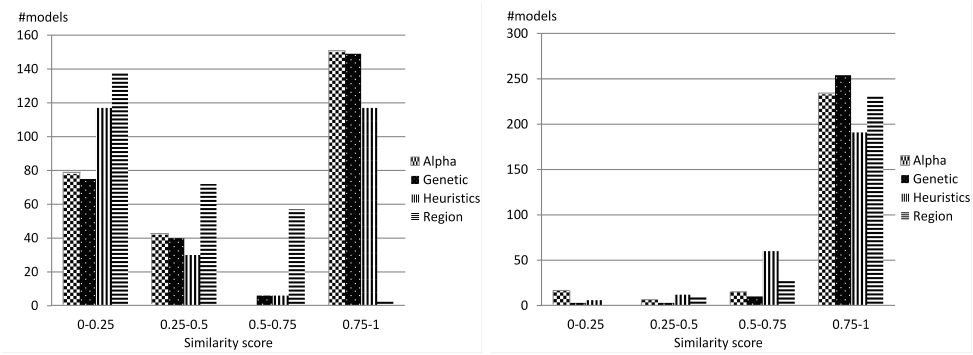
7.2.2 Process mining cost

Table 5 shows the time taken by each algorithm for mining all the models from each dataset. Note that we have set a timeout of 30mins for mining each model. The region algorithm failed to produce mined models for some models in the high-speed train dataset.

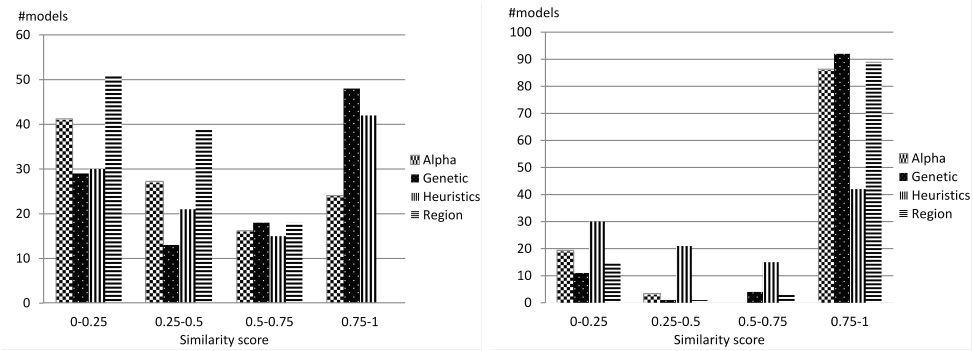
7.3 Using Our Recommendation Framework

7.3.1 Speed

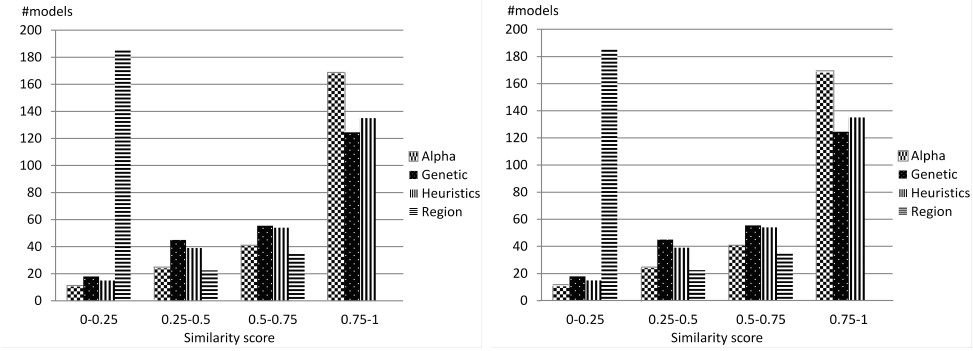
The total time taken for performing the empirical evaluation on one third of the models in each dataset is shown as Mining time in Table 6. The total training time (including the time for selecting the reference models using SVM and the time for deriving the regression



(a) Artificial dataset: Structural similarity (Left), Behavioral similarity (Right)



(b) Boiler dataset: Structural similarity (Left), Behavioral similarity (Right)



(c) Trains dataset: Structural similarity (Left), Behavioral similarity (Right)

Fig. 7. Similarity distribution for the 3 datasets

model) for these training models is shown as Training time in Table 6.

TABLE 6
Speed Comparisons (in seconds). Training time includes the time taken by components 5 and 6 from Figure 3; and also the time taken by the SVM for selecting significant reference models.

Models	Mining time	Training time
Artificial	1572	66
Boiler	745	24
Trains	107282	36

In our experiments, we use one third of the models of each dataset for the learning phase, and the rest of the two third of the models to test the recommendation accuracy. Therefore, if we empirically evaluate the mining algorithms, we would have spent 3 $\times$ Mining Time. On the other hand, the total time taken (including the time for calculating the similarities) in the learning phase is the Mining Time + the Training Time.

Furthermore, during the recommendation phase, the time taken for applying the regression model is negligible. The only significant time taken during recommendation is caused by the feature extraction. The actual feature extraction time is only a few seconds for each entire dataset, as shown in Table 4.

The results of the experiments are summarized in Table 7. In Table 7, the "Traditional approach" column shows the time needed to empirically evaluate the mining algorithms. Since we have selected one third of the models for the learning phase of our approach, the time taken for learning, as stated in the "Our approach - learning" column, is approximately one third of the time

in the "Traditional approach" column. After the learning phase is done, our approach can make recommendation of mining algorithm for the rest of the models really fast, relatively negligible when compared with the time taken for learning, as stated in the "Our approach - recommendation" column.

TABLE 7
Overall time comparison (in seconds) of our proposed method against traditional approach

Models	Traditional approach (in s)	Our approach - learning (in s)	Our approach - recommendation (in s)
Artificial	4715	1572	67
Boiler	2234	745	25
Trains	312845	107282	41

In fact, when the models of an enterprise are changed, using the traditional approach, the empirical evaluation (which will take a similar amount of time again as shown in the "Traditional approach" column of Table 7) need to be done again in order to choose the most appropriate mining algorithm for the updated models. Using our approach, we may not need to perform the learning phase again and can still recommend an effective mining algorithm accurately. If this is the case, the difference in time taken by the traditional approach and our approach will be very large, e.g., 312845 seconds versus 41 seconds. In order to determine whether re-learning is needed, we can check if the newly selected significant reference models are similar to the previously selected significant reference models or not. If most of the selected significant reference models are different, this implies that the learnt model for recommendation may be outdated and needs to be re-learnt. As shown in Table 7), even if re-learning is needed, the total time of our approach (i.e., learning + recommendation) is still faster than the traditional approach.

7.3.2 Accuracy

To evaluate the accuracy of our recommendation framework, we first evaluate the test models empirically and for each model, determine the algorithm that produces the mined model with maximum similarity (a.k.a. the actual best algorithm). Then we apply our regression model to each of these test models without performing any empirical evaluation nor calculating the similarities. For each test model, we also determine the algorithm that has the highest estimated similarity (from the regression model), a.k.a. the estimated best algorithm. If the actual and estimated best algorithms are the same, then we consider this as correct.

Tables 8 and 9 show the accuracy of the regression model for structural similarity and behavioral similarity, respectively.

Overall, the results show that our approach can produce very accurate estimated results, at a fraction of time needed for empirical evaluation.

TABLE 8
Accuracy For Structural Similarity

Models	Size	#Correct	Accuracy
Artificial	180	170	94.44%
Boiler	72	70	97.22%
Trains	162	156	96.30%

TABLE 9
Accuracy For behavioral Similarity

Models	Size	#Correct	Accuracy
Artificial	180	174	96.67%
Boiler	72	66	91.67%
Trains	162	154	95.06%

7.3.3 Examples

Consider the examples shown in Section 3 again, Tables 10 and 11 show the structural and behavioral similarities estimated from our regression model (R) for models SD050 and MM240, without performing the empirical evaluation on these two models, respectively.

TABLE 10
SD050: Similarities from empirical evaluation vs similarities estimated from the regression model

Similarity	Alpha	Genetic	Heuristic	Region
Structural (E)	0.1800	1	1	0.1000
Structural (R)	0.0627	0.4504	0.3633	0.0301
Behavioral (E)	0.7500	1	1	1
Behavioral (R)	0.8593	1	1	0.8853

Although the estimated similarity values are different from the similarity values obtained from the empirical evaluation (E), they have the same ranking order. For example, in Table 10, according to both the empirical evaluation and the regression model, Genetic and Heuristic algorithms produce the models with much better structural similarity than the other 2 algorithms. On the other hand, all algorithms produce the models for SD050 with very good behavioral similarity.

TABLE 11
MM240: Similarities from empirical evaluation vs similarities estimated from the regression model

Similarity	Alpha	Genetic	Heuristic	Region
Structural (E)	1	0.8600	1	NA
Structural (R)	1	0.8862	0.9503	0.0567
Behavioral (E)	1	1	1	NA
Behavioral (R)	0.9576	0.7002	0.8793	0.1940

8 Conclusions and Future Work

There is no widely-accepted benchmark to evaluate and compare process mining algorithms. We have proposed a new framework that can efficiently select the most suitable process mining algorithm (according to its performance on the provided process models) for a given enterprise. Moreover, to the best of our knowledge, we are the first ones to propose a method to identify and select reference models that are the most appropriate

for comparing the given business process mining algorithms. After the appropriate reference models are identified, features from these models are extracted and reduced to efficiently derive a regression model. The regression model can then be used to evaluate, compare and rank the available process mining algorithms based on the given business processes from the enterprise, without the need of tedious, computational expensive empirical evaluation.

Using an artificial dataset and two real-life datasets, we found that

- The estimated similarities obtained from the trained models are consistent with the ones obtained from empirical evaluation. The estimated best mining algorithm recommended by our framework is almost the same as the one found by empirical evaluation i.e., with more than 90% accuracy.
- We have proposed a novel mechanism to select reference models with properties that can clearly differentiate the performance of the available process mining algorithms, i.e., only significant reference models will be used for training. This optimization step will enhance both the accuracy and efficiency during the training phase of our approach. Moreover, when the models of an organization change, we can use this mechanism to detect if those previously selected, significant reference models have been changed. A new training phase may be necessary when significant reference models are changed.
- The time taken by our recommendation framework is mostly due to the learning phase. In the learning phase, the major time overhead is due to the computation of similarities (performed by running the corresponding process mining algorithms). Therefore, if we only use a small number of models for training, and apply our recommendation system for a much larger set of models. The speed improvement is dramatic (as shown in our experiment results).
- When our recommendation framework is trained for process models from a particular enterprise, unless substantial changes have been made, it is not necessary to re-learn for any subsequent changed process models or new models. Applying the regression model to obtain a recommendation for a model set only takes a few seconds, instead of the hours of benchmarking usually required to determine which algorithm will produce better results.

In summary, different from traditional process mining evaluation and benchmarking, using our approach, only a portion of process models need empirical evaluation and other process models can be recomended directly via a regression model.

Our ongoing work includes using the proposed metric for reference model selection to analyze the algorithmic behavior of each of the common business process mining algorithms. As a result, we hope to derive a more versatile and powerful business process mining algorithm.

REFERENCES

[1] J. Bae, L. Liu, J. Caverlee, L-J. Zhang, and H. Bae. Development of distance measures for process mining, discovery and integration. *Int. J. Web Service Res.*, 4(4):1–17, 2007.

[2] M. Becker and R. Laue. Analysing differences between business process similarity measures. In *International Workshop on Process Model Collections, Clermont-Ferrand, France*, 2011.

[3] M. Bishop and F.R. Eng. *Pattern Recognition and Machine Learning*. Lecture Notes in Machine Learning. Springer, 2006.

[4] J. Carmona, J. Cortadella, and M. Kishinevsky. A region-based algorithm for discovering Petri nets from event logs. In *BPM'08*, pages 358–373, 2008.

[5] A.K.A. de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst. Genetic process mining: an experimental evaluation. *Data & Knowledge Engineering*, 14:245–304, 2007.

[6] J. Desel and G. Juhas. What is a Petri net? In *Unifying Petri Nets, LNCS 2128*, pages 1–25. Springer-Verlag, 2001.

[7] R. Dijkman, M. Dumas, B. van Dongen, R. Käärik, and J. Mendling. Similarity of business process models: Metrics and evaluation. *Information Systems*, 36(2):498–516, April 2011.

[8] R.M. Dijkman, M. Dumas, and L. García-Ba nuelos. Graph matching algorithms for business process model similarity search. In U. Dayal et al., editor, *BPM 2009*, volume 5701 of *Lecture Notes in Computer Science*, pages 48–63. Springer-Verlag, Berlin, 2009.

[9] M. Dumas, L. García-Ba nuelos, and R.M. Dijkman. Similarity search of business process models. *Bulletin of the IEEE Technical Committee on Data Engineering*, 32(3):25–30, 2009.

[10] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Series in Mathematics. Cambridge University Press, 1985.

[11] IEEE Task Force on Process Mining. Process Mining Manifesto. In F. Daniel, S. Dustdar, and K. Barkaoui, editors, *BPM 2011 Workshops*, volume 99 of *Lecture Notes in Business Information Processing*, pages 169–194. Springer-Verlag, Berlin, 2011.

[12] T. Jin, J. Wang, and L. Wen. Efficiently querying business process models with BeehiveZ (demo paper). In *BPM 2011*, 2011.

[13] I.T. Jolliffe. *Principal Component Analysis*. Series in Mathematics. Springer-Verlag, 1986.

[14] J. Mendling. Testing density as a complexity metric for EPCs. In *German EPC Workshop on Density of Process Models*, 2006.

[15] J. Mendling. *Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness*, volume 6 of *Lecture Notes in Business Information Processing*. Springer, 2008.

[16] J. Mendling, M. Moser, G. Neumann, H.M.W. Verbeek, B.F. van Dongen, and W.M.P. van der Aalst. Faulty EPCs in the SAP reference model. In *BPM'06*, pages 451–457, 2006.

[17] J. Mendling, H.A. Reijers, and J. Cardoso. What makes process models understandable? In *Proceedings of the 5th international conference on Business process management*, BPM'07, pages 48–63. Springer-Verlag, 2007.

[18] J. Mendling, H.M.W. Verbeek, B.F. van Dongen, W.M.P. van der Aalst, and G. Neumann. Detection and prediction of errors in EPCs of the SAP reference model. *Data and Knowledge Engineering*, 64(1):312–329, 2008.

[19] T. Murata. Petri nets: properties, analysis and applications. *Proceedings of the IEEE*, 77:541–580, 1989.

[20] A. Rozinat, A.K.A. de Medeiros, C.W. Günther, A.J.M.M. Weijters, and W.M.P. van der Aalst. Towards an evaluation framework for process mining algorithms. In *BPM Center Report BPM-07-06, BPMcenter.org*, 2007.

[21] A. Rozinat, A.K.A. de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst. The need for a process mining evaluation framework in research and practice. In A. ter Hofstede, B. Benatallah, and H.Y. Paik, editor, *BPM 2007 Workshops*, volume 4928 of *Lecture Notes in Computer Science*, pages 84–89. Springer-Verlag, Berlin, 2008.

[22] W.M.P. van der Aalst. The application of Petri nets to workflow management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.

[23] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow mining: A survey of issues and approaches. *Data and Knowledge Engineering*, pages 237–267, 2003.

[24] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow mining: discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.

[25] B.F. van Dongen, A.K.A. de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W.M.P. van der Aalst. The ProM framework: a new era in process mining tool support. In G. Ciardo and P. Darondeau, editors, *International Conference on Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer-Verlag, Berlin, 2005.

[26] I. Vanderfeesten, J. Cardoso, J. Mendling, H.A. Reijers, and W.M.P. van der Aalst. Quality metrics for business process models. In *2007 BPM and workflow handbook*, pages 179–190, 2007.

[27] J. Wang, T. He, L. Wen, N. Wu, A.H.M. ter Hofstede, and J. Su. A behavioral similarity measure between labeled Petri nets based on principal transition sequences - (short paper). In *OTM Conferences (1)*, pages 394–401, 2010.

[28] J. Wang, S. Tan, L. Wen, R.K. Wong, and Q. Guo. An empirical evaluation of process mining algorithms based on structural and behavioral similarities. In *ACM SAC, Trento*, 2012.

[29] J. Wang, R.K. Wong, J. Ding, Q. Guo, and L. Wen. On recommendation of process mining algorithms. In *IEEE ICWS, To appear*, 2012.

[30] A.J.M.M. Weijters, W.M.P. van der Aalst, and A.K.A. de Medeiros. Process mining with HeuristicsMiner algorithm. In *BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven*, 2006.

[31] L. Wen, W.M.P. van der Aalst, J. Wang, and J. Sun. Mining process models with non-free-choice constructs. *Data Mining and Knowledge Discovery*, 15(2):145–180, 2007.

[32] 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.

Jianmin Wang graduated from Peking University, China, in 1990, and got his M.E. and Ph.D. in Computer Software from Tsinghua University, China, in 1992 and 1995, respectively. He is now a full professor at the School of Software, Tsinghua University. His research interests include: unstructured data management, workflow & BPM technology, benchmarking database system, software watermarking, and mobile digital right management.

Raymond Wong is an Associate Professor at School of Computer Science and Engineering, University of New South Wales and CTO of Cohesive Data Inc. During 2005-2010, he founded and led the database research group at National ICT Australia Limited (NICTA). He has published more than 100 research papers and supervised more than 12 PhD graduates to completion. He received his BSc from The Australian National University, MPhil and PhD from Hong Kong University of Science and Technology.

Jianwei Ding (born in 1986) studied mathematics at Sichuan University, China. From then to now, he is a PhD student in computer science at Tsinghua University. His reasearch interets include business process mining and data mining.

Qinlong Guo is an undergraduate in School of Software, Tsinghua University, China ,and will receive his B.S in 2012. His research interests includes workflow technology.

Lijie Wen is now an assistant professor in School of Software, Tsinghua University. His research interests are workflow technology, process mining and process model management. He received his B.S. and Ph.D. degrees in Department of Computer Science and Technology, Tsinghua University in 2000 and 2007 respectively. He has published more than 30 academic papers on international conferences and journals (such as Data Mining and Knowledge Discovery, Journal of Intelligent Information Systems, Data & Knowledge Engineering, Computers in Industry etc.).