Business Process Deviance Mining: Review and Evaluation

HOANG NGUYEN, Queensland University of Technology, Australia
MARLON DUMAS, University of Tartu, Estonia
MARCELLO LA ROSA, Queensland University of Technology, Australia
FABRIZIO MARIA MAGGI, University of Tartu, Estonia
SURIADI SURIADI, Queensland University of Technology, Australia

Business process deviance refers to the phenomenon whereby a subset of the executions of a business process deviate, in a negative or positive way, with respect to its expected or desirable outcomes. Deviant executions of a business process include those that violate compliance rules, or executions that undershoot or exceed performance targets. Deviance mining is concerned with uncovering the reasons for deviant executions by analyzing business process event logs. This article provides a systematic review and comparative evaluation of deviance mining approaches based on a family of data mining techniques known as sequence classification. Using real-life logs from multiple domains, we evaluate a range of feature types and classification methods in terms of their ability to accurately discriminate between normal and deviant executions of a process. We also analyze the interestingness of the rule sets extracted using different methods. We observe that feature sets extracted using pattern mining techniques only slightly outperform simpler feature sets based on counts of individual activity occurrences in a trace.

General Terms: Process Mining, Business Process Deviance, Sequence Classification

1. INTRODUCTION

Process mining is a family of techniques to extract knowledge of business processes from event logs [van der Aalst 2011]. It encompasses, among others, techniques for automated discovery of process models from event logs, techniques for checking conformance between a given process model and an event log, as well as techniques for analyzing and predicting performance of business processes based on event logs.

This paper deals with *business process deviance mining* – a family of process mining techniques aimed at analyzing event logs in order to explain the reasons why a business process deviates from its normal or expected execution. Such deviations may be of a negative or of a positive nature – cf. theory of positive deviance [Spreitzer and Sonenshein 2004]. Positive deviance corresponds to executions that lead to high process performance, such as achieving positive outcomes with low execution times, low resource usage or low costs. Negative deviance refers to the executions of the process with low process performance or with negative outcomes or compliance violations.

The input of deviance mining is an event log consisting of a set of labelled traces (the so-called *training set*). Each trace represents the execution of one case of the business process under analysis. It consists of a sequence of events, where an event corresponds to the execution of an activity. Each trace is associated with a label indicating whether the trace is "normal" or "deviant".

Given this input, the problem of deviance mining is that of calculating a function (called a *classifier*) that takes as input a trace and outputs a class for this trace (normal or deviant). Such function must produce accurate labels, i.e. it should guess the correct class of a trace both for traces in the training set but also for other unseen traces. Furthermore, since the purpose of deviance mining techniques is to explain deviance, their output should be expressed in terms of patterns or rules that allow an analyst to extract useful insights regarding the sources of the observed deviance.

Since traces consist of sequences of events, one family of techniques applicable for deviance mining is *sequence classification* [Xing et al. 2010]. The goal of sequence classification is to construct classifiers that discriminate between two or more classes of sequences. One key issue in sequence classification is that of extracting features from

the sequences that can be given as input to standard classification techniques such as decision trees. Such features can be extracted for example using sequence mining techniques. Various such techniques have been studied in the context of business process deviance mining as discussed later. However, no comparative study has been conducted to assert which techniques are more suitable in this setting.

This article presents a comparative evaluation of sequence classification techniques for business process deviance mining. Based on a review of the state of the art, the article identifies two families of techniques that have been employed for deviance mining – those based on frequent pattern mining and those based on discriminative pattern mining. Representatives of these two families are benchmarked using a battery of real-life event logs, covering situations where deviance is frequent (balanced datasets) and others where deviance is rare (unbalanced). These representative techniques are assessed in terms of *accuracy* and *interestingness* measures. The former measures serve to quantify the extent to which the extracted patterns correctly reflect the classification of cases into normal and deviance, whereas the latter measures serve as a proxy for the usefulness of the extracted patterns.

This article is an extended version of a conference paper [Nguyen et al. 2014]. With respect to the conference paper, this article adds: (i) a systematic classification and review of deviance mining techniques; (ii) a refinement of the evaluated techniques using Fisher score for feature selection; (iii) an extended evaluation with three additional event logs; and (iv) an assessment of the interestingness of the extracted classifiers in addition to their accuracy.

The paper is structured as follows. Section 2 discusses existing methods for business process deviance mining. Section 3 outlines the methods for feature extraction and for classification evaluated in this study. Next, Section 4 presents the experimental results in terms of classification accuracy while Section 5 presents the results in terms of rules interestingness. Section 6 summarizes the contribution and discusses directions for future work.

2. STATE OF THE ART

To analyze the state of the art in the field of sequence mining for business process deviance mining, we started by conducting a literature search using the systematic review principles of [Kitchenham 2004]. The search process started by submitting queries to a well-known literature database, Google Scholar.[1] with keywords associated with the scope of the paper. We constructed queries by combining the keyword "business process" with the following keywords: "deviance mining" (referring to the problem under study), "sequence mining" (the broad class of solutions of interest) and "discriminative patterns" (the specific class of solutions of interest). Since the term "workflow" is sometimes used as a quasi-synonym of "business process", we also included queries combining "workflow" with the above keywords. For each query, we gathered the first 20 hits in Google Scholar. As we conducted the search, we noted additional terms appearing in the titles of relevant papers, namely "sequential patterns" and "signature patterns" and also used such terms in the search. The search was performed in September 2015. It resulted in 200 hits. Based on title, we filtered out papers that were clearly out of scope and removed duplicates. This filtering reduced the number of candidate papers to 34. We then proceeded with an inspection of the references of each paper in order to find other papers related to the topic. This activity increased the number of candidate papers to 48.

From these publications we could classify them into three main groups: (1) pattern mining papers which focus on discovery of certain interesting patterns from datasets,

[1] http://scholar.google.com

(2) classification papers which propose new techniques and algorithms to improve classification quality with regards to a target class variable, and (3) pattern-based classification papers which involve mining patterns and evaluating the predictive power of the patterns with regards to a target class variable. Note that papers in all groups span related work within and outside the domain of business processes. Our review was then concentrated on the last group of 9 papers since we wanted to survey related techniques which have been evaluated in or can be applied for business processes. From this group, we selected two representative papers of automated pattern mining and classification evaluation. The first one is [Lo et al. 2009] which examined discriminative patterns in the field of software engineering, and the second one is [Bose and van der Aalst 2013] which examined different types of sequential patterns in business processes. In addition, based on our knowledge, we also added two case study papers which proposed manual but closely related approaches: [Suriadi et al. 2013] examined business process activities and their effects on the process duration, and [Swinnen et al. 2012] examined the predictive power of sets of activities. Consequently, our survey could identify a taxonomy (see Fig. 1) represented by four papers and consisted of three categories of techniques: (1) based on individual activities, (2) set-based, and (3) sequence-based. In addition sequence-based techniques could be further distinguished into sequential and discriminative.

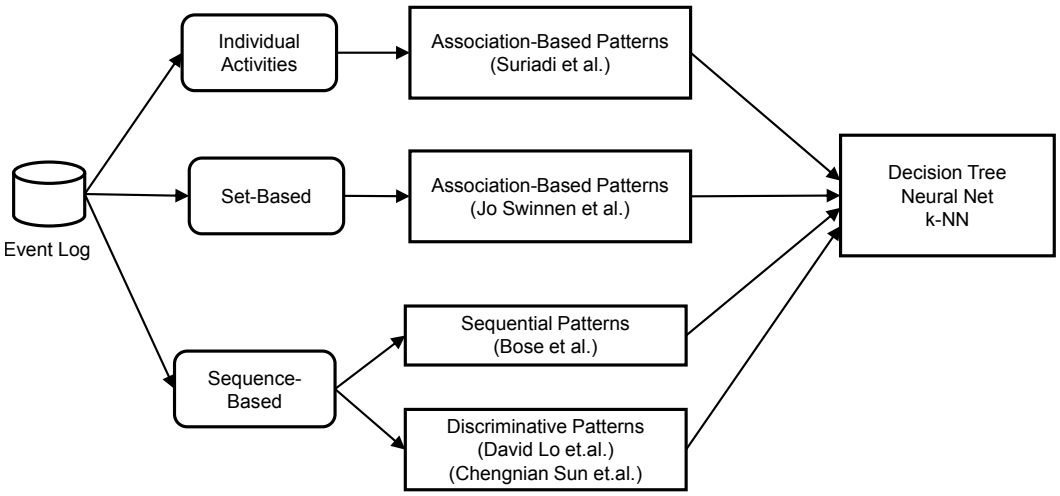

Fig. 1: Taxonomy of Business Process Deviance Mining Approaches

All automated techniques for deviance mining (i.e. excluding manual delta analysis), involve a pattern extraction phase where patterns are extracted from traces seen as sequences of simple symbols (tokens) representing activity occurrences, followed by a classifier construction phase, where each trace is abstracted as a vector of features, and these vectors are given as input to a classification method such as decision tree miner, which produces a *classifier*. The resulting classifier is then analyzed in order to extract individual patterns or rules combining multiple patterns, that explain a high percentage of the observed deviance. These patterns or rules are given as input to analysts to help them understand the sources of observed deviance in the process.

Different techniques differ depending on the abstraction of the event logs used to define the features. The three ways of abstraction as shown on the taxonomy are described as follows.

(1) Occurrence count of *individual activities* in a trace as in [Suriadi et al. 2013]. In this feature extraction method, each activity type appearing in the log (e.g. "Receive Purchase Order Change", "Issue Invoice", "Invoice Paid") is treated as a numerical feature. For a given trace, the value of an activity feature is the number of times the activity in question occurs in the trace. For example if in a given trace, activity "Receive Purchase Order Change" appears 3 times, this is the value of the corresponding activity feature. This approach is quite straightforward with consideration that single or multiple activities may have significant influence on the case outcome.
(2) Frequent *set of trace attributes* as in [Swinnen et al. 2012]. In this feature extraction method, frequent sets of activities are taken as a feature regardless of the order of their occurrence in the trace. A trace in this view follows the market basket data model in data mining. Frequent item set mining algorithms, such as Apriori [Agrawal et al. 1994], can then be applied to mine patterns that meet a support and/or confidence threshold. This approach takes interest in the association among activities *regardless of their ordering*.
(3) *Sequence of events* that occurs multiple times in a trace or across traces. These features represent different way of event coordination such as loop, sub-processes or synchronization. We select two representative types of sequence-based features: one from [Bose and van der Aalst 2009] (called Sequential Patterns) and one from [Lo et al. 2009] (called Discriminative Patterns).
    (a) [Bose and van der Aalst 2009] proposes sequential patterns including *Tandem Repeats*, *Alphabet Tandem Repeats*, *Maximal Repeats* and *Alphabet Maximal Repeats*. These patterns capture different types of control-flow relations in a business process (loops, parallelism and subprocesses) and have been shown to be potentially suitable for analyzing business process deviance [Bose and van der Aalst 2013]. More specifically, tandem repeats mining searches for recurrent sequences of events within a trace which represent loops, whereas maximal repeats mining searches for any repetitive sequences in the whole log which represent sub-processes. An alphabet-type feature represents multiple tandem repeats or maximal repeats, respectively, if they share the same constituent activities (alphabet is the set of unique activities constituting a sequential pattern). Alphabet-based patterns, thus, capture variations in tandem repeats or maximal repeats due to parallelism.
    (b) [Lo et al. 2009] proposes iterative features consisting of consecutive or non-consecutive events that have multiple occurrence within a trace or across traces. This technique is different to tandem/maximal repeats in the sense that the atomic events in an iterative feature do not necessarily occur close together, which captures synchronized behaviors among events. In addition, [Lo et al. 2009] uses feature selection technique to return only features that are strongly associated with the outcome classes (so called "Discriminative Patterns"). This association is measured by a weight value, such as *Fisher score* [Duda et al. 2012].

The first method (*individual activities*) is in essence a baseline. Indeed, the occurrence of a single activity in a trace is the simplest form of feature one can extract from a sequence. In this work, we aim at evaluating how much added-value other feature extraction methods (i.e. set-based and sequence-based variations) could contribute on top of the *individual activities*. Thus, we study seven feature sets in the sequel: (i) individual activities (IA); (ii) maximal repeats (MR); (iii) alphabet maximal repeats (MRA); (iv) tandem repeats (TR); (v) alphabet tandem repeats (TRA); (vi) iterative patterns (IP); and (vii) set-based patterns (SET).

Deviance mining is partially related to predictive monitoring of business processes [Maggi et al. 2013; Metzger et al. 2015]. Whereas deviance mining aims at explaining the reasons for deviant cases in *post mortem* manner, predictive monitoring aims at predicting whether or not an ongoing case will end up in a normal or a deviant category upon its completion – for example predicting whether or not an ongoing case will lead to a customer complaint or a deadline violation. Deviance mining and predictive monitoring techniques have overlapping inputs but distinct outputs. Deviance mining techniques take as input a set of traces classified into normal and deviant, and returns a set of rules that explain the deviance. Predictive process monitoring techniques additionally take as input the incomplete trace of a running case of the process, and they return a prediction of the outcome of this case.

Given their overlapping inputs, predictive monitoring and deviance mining techniques have common concerns. Specifically, they both have to deal with the question of how to extract features from the set of historical traces in order to construct a classifier – with the difference that in predictive monitoring, one has to take into account the fact that the classification is done with respect to the trace prefix of an incomplete case, and thus the "state" of execution of the current case needs to be taken into account. Not surprisingly, some of the feature extraction methods mentioned above can also be found in the context of predictive monitoring. For example, [Maggi et al. 2013] considers both individual activity features as well as frequent sequential patterns, whereas [Leontjeva et al. 2015] extends these features with others, including the index in the trace where a given activity occurs (e.g. does a given activity occur in the first or second position in the trace?) as well as other features extracted from hidden markov models constructed from the historical traces. [de Leoni et al. 2016] provides a framework for feature extraction that additionally considers the counts of individual activity occurrences and their index as possible features, as well as a range of other features related to the data-flow and the resource perspective (e.g. current value of a given variable of the process, and current workload of a given resource). Similar features are also employed in [Conforti et al. 2013; Conforti et al. 2015] to predict process risks. With respect to these latter two works, the present article focuses on studying the predictive power of control-flow features. On the other hand, it goes deeper by considering not only individual activity features but also set-based and sequence-based patterns as discussed above.

3. EVALUATION SETUP

This section presents the evaluation setup. First, it describes the datasets used; next it outlines the methods used for feature extraction and for classification.

3.1. Datasets

We used six datasets derived from five real-life event logs. We selected these datasets in order to cover classifications of "normal" and "deviant" cases based on *temporal* and *non-temporal* criteria. Note that *case* and *trace* are used interchangeably in this paper, as well as *feature* and *pattern*. In the former criterion, the difference between deviant and normal cases is made on the basis of the process duration w.r.t. a duration threshold (i.e. slow vs. normal processes); in the latter criterion, the difference between types of cases depend on an outcome attribute of the case (e.g. failing or successful outcome).

The six logs fall into two sets. The first set (including the first two logs in Table I and II) is recorded software behaviors [Lo et al. 2009]. These two logs contain traces in which every event is a software command. Every trace can be viewed as a series of commands executed while using the software. The trace outcome is marked as "successful" when the execution has no errors, and "failing" when the execution results in

Dataset	Deviance criterion	Deviant class distribution
Schedule	non-temporal	50%
MySQL	non-temporal	50%
Hospital1	temporal	45%
Hospital2	temporal	44%
Insurance1	temporal	62%
Insurance2	non-temporal	71%

Table I: Labeling of the six datasets used for the evaluation.

Dataset	Normal cases	Deviant cases	Total cases	Total event classes	Mean event classes per case	Mean events per case
Schedule	2,140	2,140	4,280	19	15	124
MySQL	51	51	102	16	16	24
Hospital1	448	363	811	26	14	18
Hospital2	15,929	12,847	28,776	32	8	10
Insurance1	1,921	3,195	5,116	13	7	20
Insurance2	79	197	276	19	6	16

Table II: Descriptive statistics for the six datasets.

an error (deviant case). The Schedule log was obtained from a Siemens system and the MySQL log was obtained from the a MySQL database engine. These processes are machine-based and highly repeatable.

By contrast, the second set (including the four remaining logs in Tables I and II), contains weakly structured processes with intensive human involvement. The Hospital1 log pertains to a chest pain patient flow in an emergency department of an Australian hospital. Traces are labelled as “quick” for cases that complete within a 180 minutes, and“slow” (or deviant) otherwise.

The Hospital2 log is provided by another Australian hospital pertaining to the assessment process of mental health patients from the time the patients are referred to the hospital (as outpatients) to the time they are first assessed by qualified medical personnel. A case is labelled as “quick” if the assessment process is completed within 5 days, or “slow” (or deviant) otherwise.

The Insurance1 log comes from an Australian insurance company and records an extract of the instances of a commercial insurance claims handling process. Also for this log, we used a temporal deviance criterion for classification, with “quick” being the label for those cases that complete within 30 days, and “slow” otherwise. Also in this case, slow cases are deviant cases.

The Insurance2 log is extracted from the Insurance1 dataset to evaluate the relationship between individual activities and non-temporal outcome. The two outcome labels, “mistake” and “rejected”, are non-temporal that reflect the final decision on cases (the latter is deviant). The two activities “Authorise Decline” and “Confirm Decline” are also deliberately filtered out.

3.2. Model construction

The model construction has two stages: pattern mining and model training/testing. Given a labeled event log (i.e. a log where each trace is labeled as “normal” or “deviant”), the first stage will mine and select features. The latter stage will then construct a classifier by transforming the training traces into a vector space and using standard classification techniques. In this context, a trace t is converted to a vector of features $(\langle f_1, ...f_n \rangle, l)$, where $f_i (i = 1, 2, ..., n)$ is the value of the i^{th} feature for trace t

and l is the label (normal or deviant). The value is the frequency count of the feature in the trace.

We use a feature selection technique as used in [Cheng et al. 2008] and [Lo et al. 2009]. This technique selects a feature based on a weight of its statistical correlation with the outcome classes. The weight can be, for example, *information gain* or *Fisher score*. Fisher score is used in this paper in the same way as [Lo et al. 2009]. Basically, in the first place, this technique selects an initial set of frequent features based on a support (frequency) level. Then, Fisher scores are computed for all features. Finally, it selects a final set of features by sequentially checking each feature from a high to low Fisher score ranking and selecting one that covers at least one trace from the trace database.

In this technique, the number of features selected is controlled by two parameters: *minimum support level* and *coverage threshold*. According to [Cheng et al. 2008], the support level has an impact on the discriminative power. A minimum support level of 0.25 is used for all feature types. The coverage threshold θ is used in the final selection step, in the way that any trace covered by more than θ features will be excluded from being checked against the next features for coverage. The aim is the final feature set will contain high-score features that can cover as many traces as possible, without being overly concentrated on a small proportion of traces. A common coverage threshold of 5 is used for all feature types.

In the second stage, to construct classifiers from the labeled samples, we can use a range of methods. Given that we seek an explainable classifier, a natural choice is decision tree learning, which produces trees from which human-readable rules can be extracted. This is the approach employed for example in [Suriadi et al. 2013; Bose and van der Aalst 2013; Sun et al. 2013]. Hence, we include decision trees (C4.5 method implemented in RapidMiner) as a baseline in the evaluation. We also include a k-NN (k-Nearest Neighbors) classification method in the evaluation as an example of a simple classification technique that does not construct an explicit classification model, but instead classifies a given sample based on its most similar samples. Specifically, given a sample, the k nearest neighbors are found and the class (normal or deviant) that is most common among neighbors in the training set is used to classify the given sample. We set the value of parameter k to 8, after initial trial-and-error to find a value yielding higher accuracy. Although no rules can be extracted from a k-NN classifier, the output of k-NN is explainable in the sense that given a trace t, one can show which "similar" traces have been used to classify trace t as normal or deviant. Finally, we included neural networks in the evaluation as representative of a method that can adjust itself to the data and handle large feature sets [Zhang 2000] even though it does not produce explicit (understandable) rules as decision trees do.

Different tools are used for feature mining and classification. Tandem/maximal repeats and their alphabet variants are mined using the Signature Discovery plug-in in ProM [Bose and van der Aalst 2013], while iterative pattern mining is based on the implementation in [Lo et al. 2009]. Set-based pattern mining is implemented using the FP-Growth algorithm for frequent item set mining [Han et al. 2000]. Tools for classification model training and testing are built on libraries and workflows of RapidMiner v6.0.

Five-fold validation is used consistently from pattern mining to model training and testing. A log is randomly divided into five training and five testing datasets with 80/20 proportion of original traces, respectively. The class distribution in the training and testing data remains the same as that in the original data. The training dataset is *only* used for mining patterns and producing the classification model, and the testing dataset is *only* used for testing the trained model. This is, indeed, aligned with pattern-

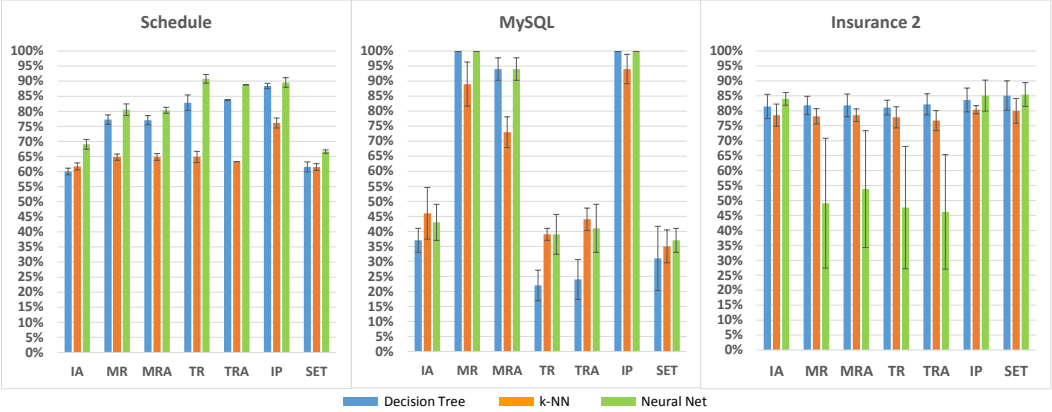

Fig. 2: Predictive Accuracy(1).

based comparative methods used in [Cheng et al. 2008], [Lo et al. 2009], [Deshpande et al. 2005] and [Lee et al. 2011].

The deviant class is usually rare in many datasets as compared to the non-deviant class. This skewed class distribution is unexpected because we want to discover deviant features and test their discriminative power. However, support-based frequent pattern mining tend to return only features of the dominant class; as a consequence, the training model built would be biased towards the dominant class. In this case, the prediction on the skewed data could be highly accurate but of little use in practice. To address this challenge, oversampling is used for training data to increase the population of the deviant class by randomly duplicating its traces. In this way, the training data of Schedule, MySQL and Hospital2 dataset have been balanced.

4. ACCURACY EVALUATION

We measured classification accuracy in terms of the standard notion of *accuracy* defined as $\frac{tp+tn}{tp+tn+fp+fn}$, where tp is the number of traces correctly classified as deviant (true positives), tn is the number of traces correctly classified as normal (true negatives), fp is the number of false positives and fn is the No. of false negatives. Additionally, we also report on the *Area Under the ROC Curve (AUC)* of each classifier. The AUC corresponds to the probability that a random negative sample is ranked higher than a random positive sample in the list of samples ranked from most likely to least likely to belong to the deviant class. AUC has been shown to be suitable for evaluating the accuracy of classification techniques both for balanced and unbalanced datasets [He and Garcia 2009]. This choice makes the results more comparable across the datasets.

The prediction result is shown in Figure 2 - 5. For the Schedule dataset, the IP, TR and TRA patterns provide the highest result (from 88 to 90%). For the MySQL dataset, the MR and IP patterns score the best (100%). For the Insurance2 dataset, the IP and SET patterns provide a slight improvement over the IA baseline (85.45% vs. 84% and 0.937 vs. 0.916). However, for other datasets (Hospital1, Hospital2 and Insurance1), we observed that the composite patterns have virtually no improvement over the IA.

Table III shows the average Fisher scores computed on the list of selected features for each feature type. Clearly, the features with better predictive power has higher average Fisher score than others. This is evident in the case of IP/Schedule and MR &IP /MySQL (X/Y refers to a pair where X is the feature type and Y is the dataset). Likewise, IP/Insurance2 is highlighted with higher average Fisher score than the others.

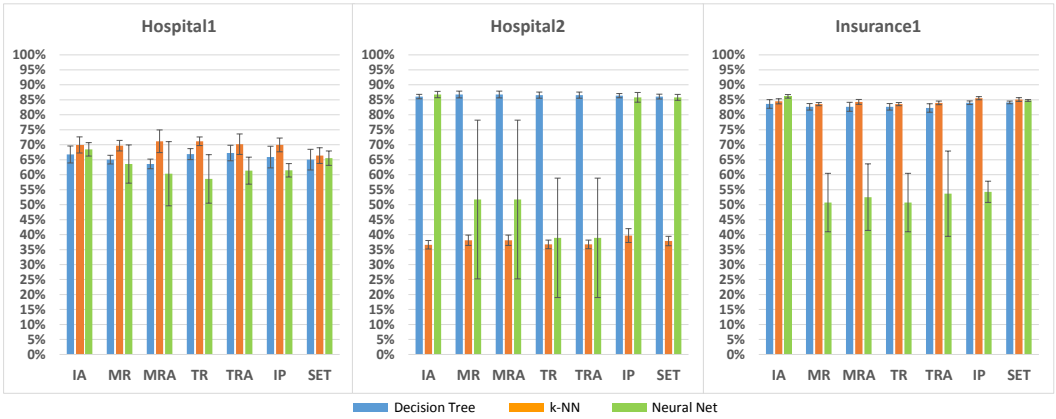

Fig. 3: Predictive Accuracy(2).

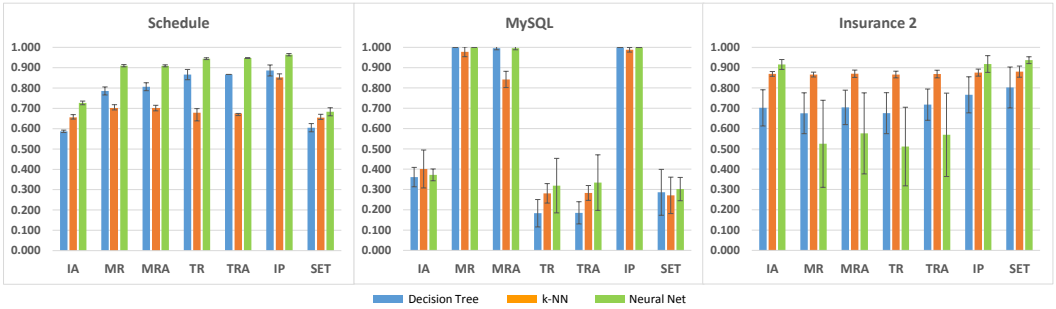

Fig. 4: AUC(1).

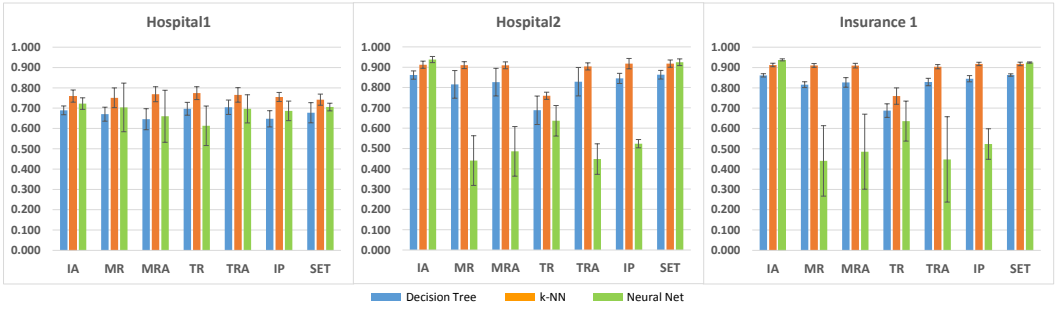

Fig. 5: AUC(2).

The list of selected features ranked by Fisher scores indeed supports that feature types with higher predictive accuracy do provide a number of highest scored features. For example, IP/Schedule has quite many patterns with high scores (see Figure 6) whereas IA/Hospital1 dominates the ranking (see Figure 7&7).

Another observation is the composite patterns, i.e. non-IA, achieve higher accuracy than IA only when the case outcome is not temporal, e.g. Schedule and MySQL datasets. In other words, if the outcome is temporal, IA patterns likely carry most of the outcome-related signals since case duration is usually susceptible to the activity

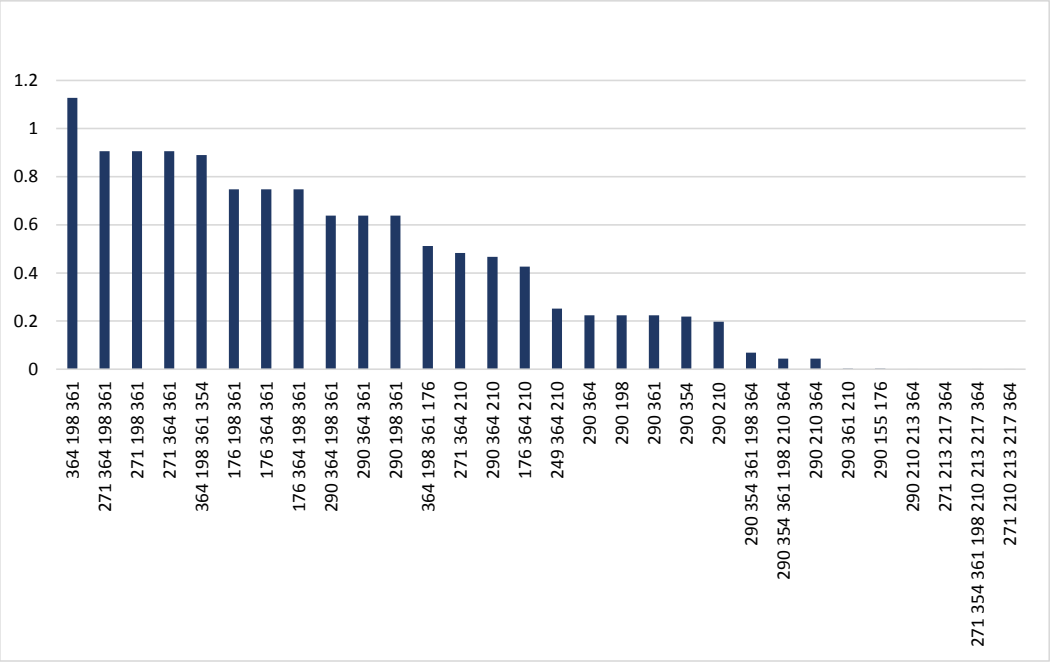

Fig. 6: Selected Features and Fisher scores (IP/Schedule). Every number represents one activity, e.g. '364' is one activity.

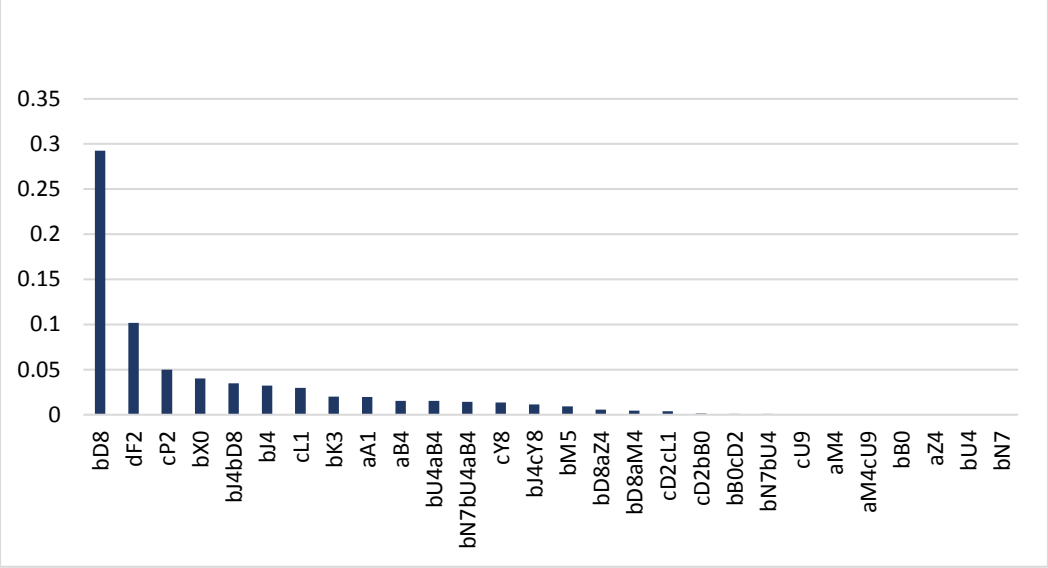

Fig. 7: Selected Features and Fisher scores (MR/Hospital1). The top-ranking encoded features such as bD8, cF2, cP2, and bx0 represent individual activities.

Dataset		IA	MR	MRA	TR	TRA	IP	SET
Schedule	Fisher Score	0.0381	0.056	0.061	0.022	0.026	**0.376**	0.107
	No. of Features	17	45	39	46	28	31	93
MySQL	Fisher Score	0.002	∞	0.616	0.0016	0.0018	**152,985**	0.003
	No. of Features	5	9	7	4	3	13	9
Hospital1	Fisher Score	0.03	0.026	0.025	0.053	0.055	0.024	0.032
	No. of Features	21	26	28	11	11	66	100
Hospital2	Fisher Score	0.019	0.053	0.053	0.03	0.03	0.065	0.067
	No. of Features	32	9	9	2	2	21	22
Insurance1	Fisher Score	0.052	0.049	0.039	0.049	0.044	0.06	0.039
	No. of Features	13	27	33	27	29	76	80
Insurance2	Fisher Score	0.044	0.065	0.075	0.065	0.066	0.1	**0.164**
	No. of Features	18	21	22	19	18	31	35

Table III: Average Fisher scores by feature types.

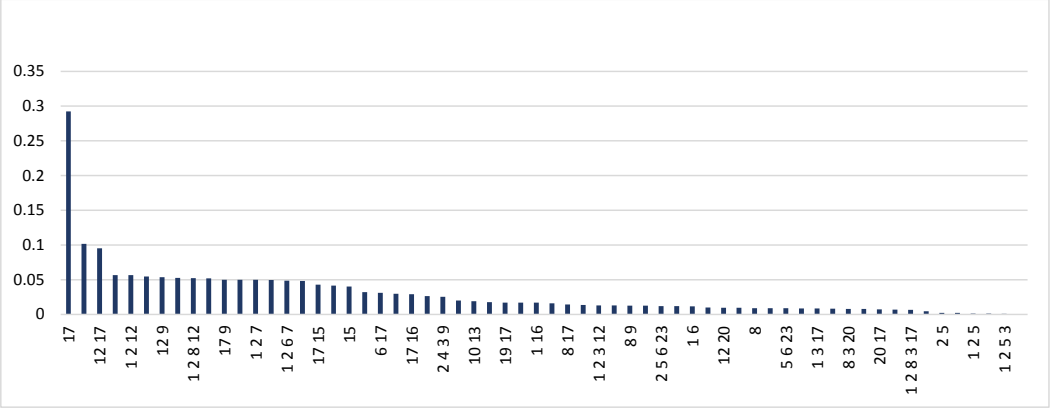

Fig. 8: Selected Features and Fisher scores (IP/Hospital1). Every numeric code represents one activity, e.g. '17' is one activity.

execution, i.e. the longer a case is, the more activities it has as observed in Hospital1, Hospital2 and Insurance1. Similarly, in the case of Insurance2, we deliberately use non-temporal outcome criterion and filter our related activities. As a result, the highest accurate feature type is SET.

In conclusion, three observations arise from our experiments. First, IA captures most of the signals if the deviance outcome is temporal. Second, composite patterns are highly predictive in case of structured processes, e.g. Schedule and MySQL datasets. Finally, in case of less structured processes, composite patterns yield negligible improvement over IA, possibly because most of outcome-related signals have been captured by IAs or non-activity features, e.g. resource and data.

Execution times. All experiments were executed on a laptop with 2.5GHz Intel CPU Dual Core, 8GB internal memory, and 64-bit Windows operating system. Fig. 9 shows the average feature mining runtime from five-fold iterations. In relation to the dataset characteristics presented in Table II, factors affecting mining performance are likely involved with the number of events per case, the number of event classes per case and the data size (total number of cases). This is observed in case of the Schedule dataset with exceptionally long cases, Hospital2 with shortest average cases but largest data size, Insurance1 with quite large data size but low number of event classes per case. Mining techniques respond to different degrees to data characteristics. For example, it

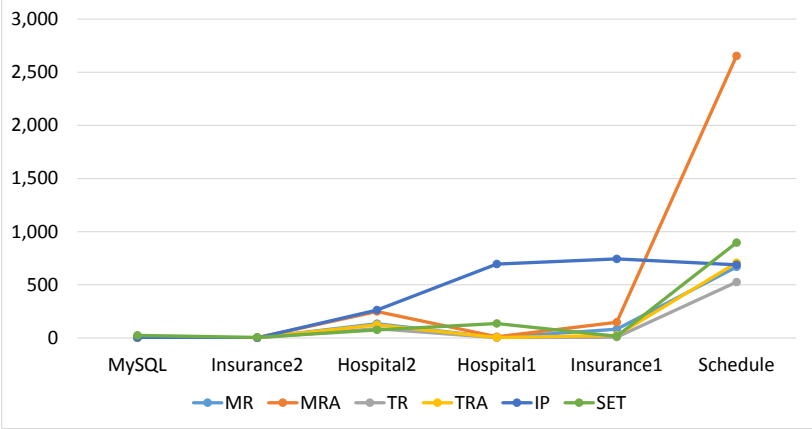

Fig. 9: Mining Runtime (in seconds)

can be observed that MRA is highly sensitive to the number of events per case while IP is not; and SET is more susceptible to the number of event classes per case.

5. RULES INTERESTINGNESS EVALUATION

We extracted rules from the six datasets across seven feature types using decision tree classifier. This technique takes the path from a leaf node of the tree to the top as a rule. All decision trees are trained with uniform input parameters. As a result, there are seven rulesets for each dataset, one per feature type. The rule has the form of $A => B$, where the antecedent A is a conjunction of multiple selectors in the form of *attribute OP value* with *OP* as relational operator, and the consequent B is the class label.

For each dataset, we compare the quality of these rulesets on a number of selected measures and observe the effect of pattern types on the rules quality. The measures used in our evaluation include:

- Rule Length: the number of operators in the antecedent part of the rule. This is averaged from all rules of a ruleset.
- %Generalization [Gallion et al. 1993]: the generalizability based on the number of training examples and the number of rules.

$$\%Generalization = (1 - \frac{\#ofRules}{\#ofTrainingExamples}) * 100 \quad (1)$$

- Probability-based Objective Rule Interestingness Measures. [Geng and Hamilton 2006] evaluates 38 interestingness measures with reference to 12 desired properties. The authors ranked these measures in terms of how much they hold a property (Table V in [Geng and Hamilton 2006]). Based on this evaluation, we were able to select six top measures as listed below (noted that Yule's Q and Yule's Y are normalized variants of Odd Ratio [Tan et al. 2004]). Readers should refer to [Geng and Hamilton 2006] for a detailed discussion on these measures while a brief description is given below.
  - Collective Strength (CS): proposed in [Aggarwal and Yu 1998] to replace the support-based paradigm in frequent itemset mining. The original idea is to measure the collective strength of an itemset I as $CS(I) = \frac{GoodEvents}{E[GoodEvents]} * \frac{E[BadEvents]}{BadEvents]}$, where *Good Events* mean the items of the itemset are present in most of trans-

actions and *Bad Events* are when they are not. In the context of rule evaluation, *Good Events* are when A and B are strongly correlated, and the opposite are *Bad Events*. The collective strength CS ranges from 0 to $+\infty$. If $CS = 0$, A and B are mutually exclusive (perfect negative correlation). If $CS = 1$, A and B are totally independent. If CS increases to $+\infty$, A and B are increasingly positively correlated.

$$CS = \frac{P(AB) + P(\neg B|\neg A)}{P(A)P(B) + P(\neg A)P(\neg B)} * \frac{1 - P(A)P(B) - P(\neg A)P(\neg B)}{1 - P(AB) - P(\neg B|\neg A)} \tag{2}$$

— Two-Way Support (TWS): proposed in [Yao and Liu 1997] to search for interesting knowledge in multiple databases. This measure ranges from $-\infty$ to $+\infty$. If it approaches $+\infty$, the relation $A => B$ increases and vice versa, the relation $B => A$ increases when it approaches $-\infty$.

$$TWS = P(AB) * \log_2 \frac{P(AB)}{P(A)P(B)} \tag{3}$$

— $\phi - Coefficient$: this measure is based on Pearson correlation coefficient which ranges between -1 and +1. When it is greater than 0, A and B are positively correlated. When it equals 0, A and B are independent. When it is less than 0, A and B are negatively correlated.

$$\phi - Coefficient = \frac{P(AB) - P(A)P(B)}{\sqrt{P(A)P(B)P(\neg A)P(\neg B)}} \tag{4}$$

— Piatetsky-Shapiro (PS): originally proposed in [Piatetsky-Shapiro 1991], this measure reflects the variance of probability when A and B are correlated and independent, with values ranging from -0.25 and 0.25. If PS is zero, A and B are totally independent. If PS is greater than 0, A and B are positively correlated, and vice versa, less than 0 when A and B are negatively correlated.

$$PS = P(AB) - P(A)P(B) \tag{5}$$

— Yule's Q ($YuleQ$): this is a normalized version of Odds Ratio (OR) [Mosteller 1968], and based on the contingency table involving A and B. Yule's Q ranges from -1 (perfect negative correlation) to +1 (perfect positive correlation).

$$OR = \frac{P(AB)P(\neg A \neg B}{P(A\neg B)P(\neg AB)} \tag{6}$$

$$YuleQ = \frac{OR - 1}{OR + 1} \tag{7}$$

— Information Gain (IG): this measure is related to the Lift measure $\frac{P(AB)}{P(A)P(B)}$, i.e. how many times it needs to lift the confidence from the case when A and B are totally independent $P(A)P(B)$ to the case when A and B are statistically correlated $P(AB)$. IG ranges from $-\infty$ (negatively correlated) to $+\infty$ (positively correlated).

$$IG = log \frac{P(AB)}{P(A)P(B)} \tag{8}$$

5.1. Patterns and Rules

Across all datasets and feature types, we observe that patterns with high Fisher score are often chosen to form rules with high coverage based on the number of cases covered. This is justifiable because the Decision Tree would pick up these patterns as preferable splitting points based on their strong discriminative power (Gini index is used for Decision Tree in our experiment). The higher Fisher score a pattern has, the more rules they take part in and the higher coverage those rules are. Several representative examples are reviewed in details as follows.

In particular, IP/Schedule provides a remarkable pattern: "364 198 361" (every number represents one activity). The related high-coverage rule is: *IF "364 198 361"* $\geq$ *0.500 THEN failing*, meaning if this pattern occurs, the trace would have a failing outcome. Conversely, no IA-based rules are formed based on these three events, meaning activities individually do not carry any outcome-related signals.

For the Hospital1, a remark is that most rules contain the "Nursing Progress Notes" activity which has the top Fisher score. In this process, "Nursing Progress Notes" seems to dominate the relationship with the duration-based outcome, i.e. if there are more "Nursing Progress Notes", the case would last longer.

For the Insurance1, the IA-based rules basically have similar forms as the IP- and MR-based if composite patterns are broken down into individual activities. Although some IP- and MR-based patterns have slightly higher scores than the best of IAs, for example "1 13 16" (0.344 vs. 0.159) and "1 16" (0.292 vs. 0.159), but it is not substantial as in the case of the Schedule dataset. That's possibly the reason why IP has a better predictive result than other features but cannot surpass IA.

For Insurance2, notably some IP- and SET-based patterns have higher Fisher scores than IA and others. Specifically, there is one remarkable IP-based rule: *IF "Process Additional Information, Contact Customer"* $\geq$ *0.5 THEN Rejected Case* (51% coverage), meaning if "Process Additional Information" and "Contact Customer" occur together, the case would be rejected. We observed that the IA also provides rules relating to "Process Additional Information" and "Contact Customer" but its rules are longer with other activities included. This is likely related to the fact that though "Process Additional Information" and "Contact Customer" rank very high in IA feature list, they are lower than the IP-based patterns.

In this analysis, it is more evident that IAs likely carry most of the outcome-related signals in case of temporal outcome criterion (Hospital1 and Insurance1). Composite patterns, however, contribute some interesting rules (Schedule and Insurance2).

5.2. Rules Interestingness Measures

Different metrics are used to evaluate the quality of a ruleset. Rule Length is computed by averaging length of individual rules. %Generalization is computed according to formula (1). For probability-based measures, cumulative interestingness is computed by firstly sorting rules within a ruleset in decreasing order of the measure values, then accumulating these values from the highest to lowest. The cumulative measure for one feature type is visualized as an upward curve by the number of rules (Fig. 10,11,12).

Our observation is summarized as follows with selected graphs for illustration due to space limitation.

For the Schedule, the IP-based ruleset has consistently high interestingness over the others, except in ϕ-Coefficient where the TR and TRA are the best (Fig. 10). Notably, IP-based ruleset also achieves the highest predictive outcome, as well as TR and TRA. Rule Length and %Generalization are quite similar across feature types.

For the MySQL, similar to the Schedule dataset, IP-based and MR-based ruleset show the best cumulative interestingness, which accords with their high predictive

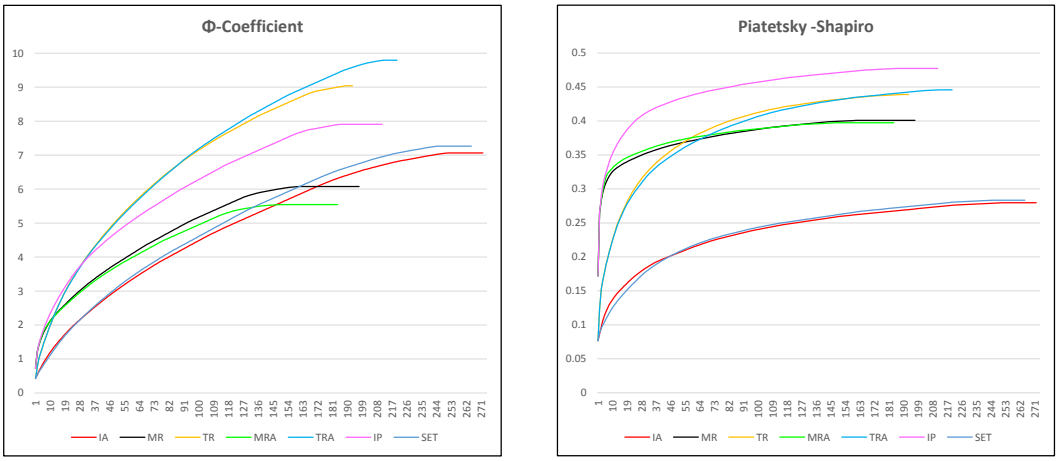

Fig. 10: Schedule dataset - ϕ-Coefficient and Piatetsky-Shapiro

strength (Fig. 11). They also have the shortest length and the greatest %Generalization (Rules were not generated for several feature types due to pruning parameters).

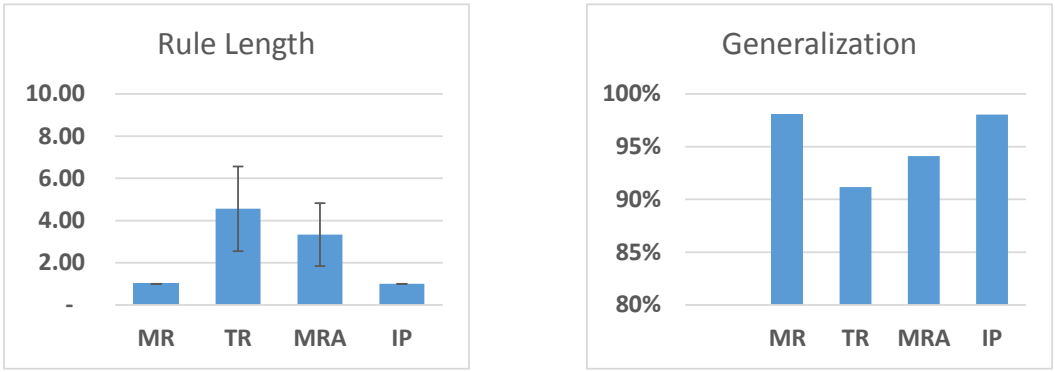

Fig. 11: MySQL dataset - Rule Length & %Generalization

For the Hospital1, the MRA-based and IP-based ruleset have higher cumulative interestingness than others in all measures (Fig. 12). They also have high %Generalization after the IA-based. All rulesets are approximately equal in rule length.

For the Hospital2 dataset, all rulesets are balanced across all measures, which corresponds to their equal predictive power and Fisher score as shown in section 4 and 5.1. For the Insurance1 dataset, similar to the Hospital1 dataset, the MRA-based and IP-based rulesets are higher than others in all interestingness measures. The IP-based and Set-based have slight improvement over others in Rule Length and %Generalization. For the Insurance2 dataset, the MRA-based cumulatively outperforms others in all measures except ϕ-Coefficient, Yule's Q and Information Gain where the Set-based is the best.

In summary, across all datasets, the IP-based, MRA-based and Set-based ruleset are often at the top of the cumulative interestingness measurement. The IP-based often results in shorter length and higher %Generalization. The IA-based ruleset is often

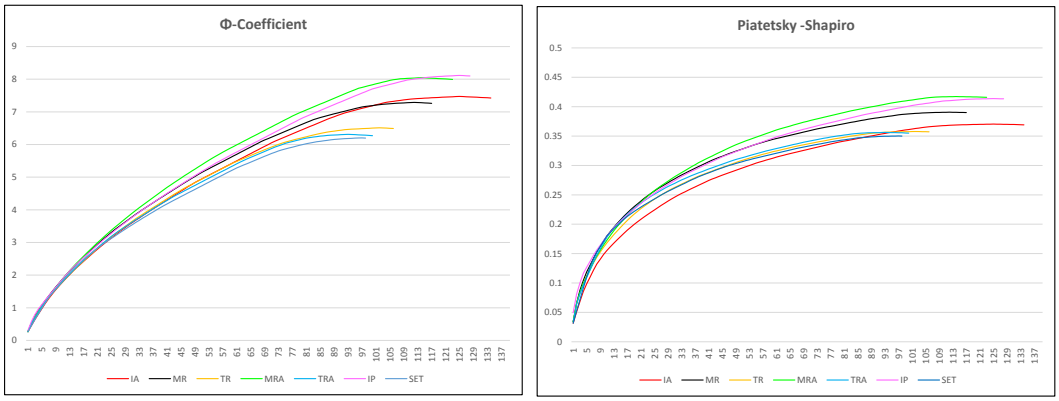

Fig. 12: Hospital1 dataset - ϕ-Coefficient and Piatetsky-Shapiro

below the others in most of the measures, except in Yule's Q and Information Gain where it improves after more rules are added to the ruleset. In short, rulesets based on composite patterns has higher interestingness than those based on IA.

6. CONCLUSION

Our review of the state of the art has revealed that existing techniques for business process deviance mining are based on the extraction of patterns from business process execution traces using either individual activity frequency or frequent or discriminative pattern mining approaches. The empirical evaluation has unveiled evidence that, in the context of structured (predictable) business processes, pattern mining approaches clearly outperform the basic approach based on individual activity frequency in terms of accuracy. On the other hand, in the case of business processes with high level of variability, the accuracy improvement obtained by using pattern mining approaches is negligible. The latter observation suggests that when business processes have high variability, no clear composite patterns emerge, and thus such patterns do not add predictive power over simple activity counts. However, we observed that the use of composite patterns (be them frequent or discriminative ones) generally has a positive effect on the interestingness of the mined rule sets.

In the case of processes with high variability, the maximum accuracy (AUC) achieved is around 80%, which might be insufficient in many practical applications. Underlying this limitation is the fact that the reviewed techniques for business process deviance mining treat the input as consisting of simple symbolic sequences, i.e. sequences of simple symbols (tokens) representing in our case activity or event occurrences. In some cases, including all six datasets used in this study, business process execution logs consist instead of *temporal complex* symbolic sequences, i.e. sequences of timestamped events, each event with a payload consisting of attribute-value pairs. Such logs can be extracted for example from appropriately instrumented information systems, such as patient management systems (in the case of healthcare processes) or claims handing systems (for insurance claims), or from mainstream Enterprise Resource Planning systems. It is likely that data payloads associated with events can convey significant information regarding possible deviances and thus cannot be ignored as in the techniques evaluated in this paper.

A direction for future work is thus to develop and apply techniques for extracting (discriminative) patterns from complex symbolic sequences – a non-trivial and open problem as noted in [Xing et al. 2010]. While tackling the problem of complex symbolic

sequence mining in the general case is challenging, it may be possible to reduce the problem of business process deviance mining to well-scoped subsets of this problem, for example by taking advantage of information contained in available process models, such as which activities read/update which data attributes, which data attributes are used to make a decision in the process, etc., in order to prune the pattern search space.

Acknowledgments This work is funded by the Estonian Research Council, ERDF via the Estonian Centre of Excellence Programme. This research is also supported by the Australian Centre for Health Services Innovation (#SG00009-000450) and by the Australian Research Council (#DP150103356).

REFERENCES

Charu C Aggarwal and Philip S Yu. 1998. A new framework for itemset generation. In *Proc. of PODS*. ACM, 18–24.

Rakesh Agrawal, Ramakrishnan Srikant, and others. 1994. Fast algorithms for mining association rules. In *Proc. of VLDB*. Morgan Kaufmann, 487–499.

R.P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. 2009. Abstractions in Process Mining: A Taxonomy of Patterns. In *Proc. of BPM (LNCS)*, Vol. 5701. Springer, 159–175.

R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. 2013. Discovering Signature Patterns from Event Logs. In *Proc. of CIDM*. IEEE, 111–118.

Hong Cheng, Xifeng Yan, Jiawei Han, and Philip S. Yu. 2008. Direct Discriminative Pattern Mining for Effective Classification. In *Proceedings of ICDE*. IEEE Computer Society, 169–178.

R. Conforti, M. de Leoni, M. La Rosa, and W.M.P. van der Aalst. 2013. Supporting Risk-Informed Decisions during Business Process Execution. In *Proc. of CAiSE (LNCS)*, C. Salinesi, M.C. Norrie, and O. Pastor (Eds.), Vol. 7908. Springer, 116–132.

R. Conforti, M. de Leoni, M. La Rosa, W.M.P. van der Aalst, and A.H.M. ter Hofstede. 2015. A recommendation system for predicting risks across multiple business process instances. *Decision Support Systems* 69 (2015), 1–19. DOI:http://dx.doi.org/10.1016/j.dss.2014.10.006

Massimiliano de Leoni, Wil M. P. van der Aalst, and Marcus Dees. 2016. A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs. *Inf. Syst.* 56 (2016), 235–257. DOI:http://dx.doi.org/10.1016/j.is.2015.07.003

Mukund Deshpande, Michihiro Kuramochi, Nikil Wale, and George Karypis. 2005. Frequent substructure-based approaches for classifying chemical compounds. *Knowledge and Data Engineering, IEEE Transactions on* 17, 8 (2005), 1036–1050.

Richard O Duda, Peter E Hart, and David G Stork. 2012. *Pattern classification*. John Wiley & Sons.

Roger Gallion, Chaman L Sabharwal, Daniel C St Clair, and William E Bond. 1993. Dynamic id3: A symbolic learning algorithm for many-valued attribute domains. In *Proceedings of the 1993 ACM/SIGAPP symposium on Applied computing: states of the art and practice*. ACM, 14–20.

Liqiang Geng and Howard J. Hamilton. 2006. Interestingness measures for data mining: A survey. *ACM Comput. Surv.* 38, 3 (2006), 9. DOI:http://dx.doi.org/10.1145/1132960.1132963

Jiawei Han, Jian Pei, and Yiwen Yin. 2000. Mining frequent patterns without candidate generation. In *ACM SIGMOD Record*, Vol. 29. ACM, 1–12.

Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* 21, 9 (2009), 1263–1284.

Barbara A. Kitchenham. 2004. *Procedures for Undertaking Systematic Reviews*. Technical Report. Computer Science Department, Keele University (TR/SE- 0401) and National ICT Australia Ltd. (0400011T.1).

Jae-Gil Lee, Jiawei Han, Xiaolei Li, and Hong Cheng. 2011. Mining discriminative patterns for classifying trajectories on road networks. *Knowledge and Data Engineering, IEEE Transactions on* 23, 5 (2011), 713–726.

Anna Leontjeva, Raffaele Conforti, Chiara Di Francescomarino, Marlon Dumas, and Fabrizio Maria Maggi. 2015. Complex Symbolic Sequence Encodings for Predictive Monitoring of Business Processes. In *Proc. of BPM*. Springer, 297–313. DOI:http://dx.doi.org/10.1007/978-3-319-23063-4

David Lo, Hong Cheng, Jiawei Han, Siau-Cheng Khoo, and Chengnian Sun. 2009. Classification of Software Behaviours for Failure Detection: A Discriminative Pattern Mining Approach. In *Knowledge Discovery and Data Mining*. ACM, 557–566.

Fabrizio Maggi, Chiara Di Francescomarino, Marlon Dumas, and Chiara Ghidini. 2013. Predictive Monitoring of Business Processes. In *Proc. of CAiSE*. Springer, 457–472.

Andreas Metzger, Philipp Leitner, Dragan Ivanovic, Eric Schmieders, Rod Franklin, Manuel Carro, Schahram Dustdar, and Klaus Pohl. 2015. Comparing and Combining Predictive Business Process Monitoring Techniques. *IEEE Trans. Systems, Man, and Cybernetics: Systems* 45, 2 (2015), 276–290.

Frederick Mosteller. 1968. Association and estimation in contingency tables. *J. Amer. Statist. Assoc.* 63, 321 (1968), 1–28.

Hoang Nguyen, Marlon Dumas, Marcello La Rosa, Fabrizio M. Maggi, and Suriadi Suriadi. 2014. Mining Business Process Deviance: A Quest for Accuracy. In *Proc. of OTM (LNCS)*, Vol. 8841. Springer, 436–445.

Gregory Piatetsky-Shapiro. 1991. Discovery, analysis, and presentation of strong rules. In *Knowledge discovery in databases*. AAI/MIT, 229–238.

Gretchen M. Spreitzer and Scott Sonenshein. 2004. Toward the Construct Definition of Positive Deviance. *American Behavioral Scientist* 47, 6 (2004), 828–847.

Chengnian Sun, Jing Du, Ning Chen, Siau-Cheng Khoo, and Ye Yang. 2013. Mining explicit rules for software process evaluation. In *Proc. of ICSSP*. ACM, 118–125.

Suriadi Suriadi, Moe Thandar Wynn, Chun Ouyang, Arthur H. M. ter Hofstede, and Nienke J. van Dijk. 2013. Understanding Process Behaviours in a Large Insurance Company in Australia: A Case Study. In *Proc. of CAiSE*. Springer, 449–464.

Jo Swinnen, Benoît Depaire, Mieke J. Jans, and Koen Vanhoof. 2012. A Process Deviation Analysis – A Case Study. In *Proc. of BPM'2011 Workshops*. Springer, 87–98.

Pang-Ning Tan, Vipin Kumar, and Jaideep Srivastava. 2004. Selecting the right objective measure for association analysis. *Information Systems* 29, 4 (2004), 293–313.

Wil M.P. van der Aalst. 2011. *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer.

Zhengzheng Xing, Jian Pei, and Eamonn J. Keogh. 2010. A brief survey on sequence classification. *SIGKDD Explorations* 12, 1 (2010), 40–48.

Jun Yao and HUAN Liu. 1997. Searching multiple databases for interesting complexes. *KDD: Techniques and Applications, World Scientific, Singapore* 482 (1997), 484–485.

G. P. Zhang. 2000. Neural Networks for Classification: A Survey. *IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 30, 4 (2000), 451–462.