Automate Back Office Activity Monitoring to Drive Operational Excellence

Miao He[1], Tao Qin[1], Sai Zeng[2], Changrui Ren[1], and Lei Yuan[3]

[1] IBM Research, China
{hmhem,qintao,rencr}@cn.ibm.com
[2] IBM T.J. Watson Research Center
saizeng@us.ibm.com
[3] IBM China Development Lab
leiycdl@cn.ibm.com

Abstract. Business process outsourcing (BPO) is growing rapidly with intensive competition. BPO providers aim to deliver high quality services with low cost. One of the key drivers is to optimize human resource utilization. It is critical to monitor and measure the activities of the practitioners in order to identify inefficient workers, unnecessary waste in operations, and non-standardized operations. Today's practices to monitor and measure the human activities are manual and error-prone. Motivated by increasing the accuracy and eliminating manual efforts for monitoring and measuring human activities, in this paper we present our research work to automatically classify and time the daily activity of a practitioner. Even though human behavior variations and noises brings substantial deviations and randomness, the developed *activity classifier and timer* handles the variations and reduces the noise to a satisfactory extent. The pilot results demonstrate 98.18% accuracy to classify transactions into the activity taxonomy, and 91.54% accuracy to find out the transaction cycle time therefore to aggregate to the time spent on each activity. The results are highly valued by our business partners, and the tool is considered as a revolutionary solution for human activity monitoring and measurement.

1 Introduction

Business process outsourcing (BPO) refers to the contracting of the operations and responsibilities of specific, mostly "non-core", business functions (or processes) to a third-party service provider. The outsourced functions can be "human resources, information technology, indirect procurement, finance, and accounting, etc" [10]. BPO is a rapidly growing offshore market with a projected annual growth rate of 60 percent according to [17], but not every provider has the chance to thrive in the undergoing industry prosperity. For instance, although Gartner forecasts a 6.3% in 2011 and 5% growth in 2012 for the worldwide market [1], it meanwhile predicts that "one-quarter of the top BPO operatives will not exist as separate entities in 2012," due to "economic risks, loss-making contracts, and inability to adapt to standardized delivery models" [5].

Being aware that a wide variety of factors together drives a provider's success [3, 6, 18], this application-oriented research aims to help providers achieve *operational excellence* in their *back office* processes. More specifically, we monitor and capture the *desktop application usages* of practitioners, which account for more than 90% of their working time. Based on application usages, we further develop the *activity classifier and timer*. Let's explain two key terms, *activity* and *transaction*, which will be revisited many times in the following.

Activity	The most granular level in the process taxonomy. Each activity has a particular and unique succession of processing procedures, and different activities have different successions of processing procedures.
Transaction	A transaction is an instance of a particular activity type.

Note that "transaction" has a many-to-one relation against "activity" type. Suppose there is an activity called "hotel invoice processing," then we can have one transaction with invoice number 123, and another with invoice number 456 - same activity type, differed transaction IDs (invoice numbers).

Activity classifier and timer has dual objectives - identify the correct *activity type* for each transaction, and simultaneously find the *start* and *end time* of each transaction. In our solution, a training module first learns the sequential application usage patterns of each activity type. This module takes application events with class label (the activity type) as input and implement sequential pattern mining algorithms. Then a testing module classifies and times a practitioner's work. Handling various human behavior variations and noises in the testing phase is technically challenging, because variations and noises lead to imperfect matching between actual application usages with the patterns learnt from standard operations. Typical variations and noises include (partial) batch processing, random click on applications, incomplete processing procedures, combined or interleaved processing and so on. If not handled, these variations and noises impose significantly negative impact on accuracy of classification and timing. In our approach, we incorporate our knowledge into human behaviors to handle variations and noises, and the details can be found in Section 3.

Our approach was implemented in a tool (the activity classifier and timer) and tested in a pilot with a few rounds of result validation. Our business shareholder from a leading BPO provider, endorses four major business benefits which were never thought to be feasible in the past. The foremost important benefit is that this tool can automate today's manual activity monitoring with high accuracy. One common practice of today's activity monitoring relies on practitioners to report their time spent on daily activities, plus team leads conduct floor auditing. Our solution eliminates the manual effort of "self reporting" and "floor auditing." The second benefit is that the solution can accurately count the number of transactions processed for each activity. Thirdly, we have observed an obvious behavior change of practitioners after the tool has been deployed - they become more efficient with less waste in operations just by awareness of "being watched" by the tool. Lastly, the tool can also discover the non-standard or exceptional operations, some of which should, but not yet, be documented in the desktop procedures.

In the remaining part of this paper, we will review related work in Section 2. Section 3 describes our approach in details, especially how we handle human behavior variations and noises featured in the back office service delivery. Section 4 illustrates the results, findings and business benefits after implementing our tool to the production environment. In Section 5 we conclude our work.

2 Related Work

To the best knowledge of the authors, there are no literatures relevant to automatic timer on service cycles based on knowledge discovery or data mining. But classification, the task of assigning an input object to one of several predefined categories, has many diverse applications. To name just a few, Bozorgi et al. train linear support vector machines (SVMs) on high dimension feature vectors to classify software vulnerabilities [4]; [7] develop a two-phase classifier that caters to large-scale file categorization; and [9] train a rule-based classifier for fingerprint classification. Some of the classification works focus on service process delivery like us. For example, Tang et al. develop a classifier to label the recorded conversations into a hierarchical taxonomy of the call types for a call center [16]. [11] categorize incoming emails to the contact centers based upon their contents. The resulted classifier identifies root, inner and leaf messages to track the progress of the email interactions. Note that a root message and a leaf message is the initiation and close of an interaction respectively, while an inner message is in between.

The most common classification algorithms include, but not limited to, decision tree, rule-based classifiers, nearest neighbor classifiers, Bayesian classifiers, artificial neural network, support vector machine, etc [15]. The mentioned methods depend on features rather than *feature sequences* to construct the corresponding classifiers. However, sequential patterns in application usage does matter much in our problem. Agrawal and Srikant first introduce *sequential pattern mining* in 1995 [2], which is trying to find if there exist any specific order of the occurrence of events. Notable applications area include “customer purchase behavior, Web access patterns, scientific experiments, disease treatments, natural disasters, DNA sequences, and so on” [12]. There are two classical sets of algorithms and derivatives in this area. One thought is to base the learning process upon the “Apriori” property in association rule mining, including AprioriAll, AprioriSome, DynamicSome in [2] , GSP in [14] and SPADE in [19], etc. The other series of algorithms proposed rely on recursively projecting the data sets into mutually exclusive subsets to speed up mining by avoiding scanning the entire database, see FreeSpan [8] and PrefixSpan [12] for more details.

3 Activity Classifier and Timer

We have developed a tool called *system timer* capable of capturing the starting and ending of every *application event*, *machine idle* and *keyboard idle*. An event refers to a non-switching stay on one application page. For example, system timer will record the starting time as soon as one opens Facebook in IE. Upon switching to Twitter after 10 seconds stay on Facebook, the tool promptly records the ending time of the Facebook event. Obviously, the ending time of Facebook is the starting time of Twitter. Machine idle refers to a special event type when a desktop is locked and keyboard idle is also a special event type when one let the computer on with neither mouse movement nor any key striking for a while. In this way, we can capture application events at very fine granularity which serves as the basis of the activity classifier and timer.

Before delving into the approach, we would like to walk the audience through most common non-standard human behaviors as summarized in Table 1. Figure 1 provides graphical illustrations for normal processing and variations, where each distinct shape represents a distinct application page (a processing step), and shapes with the same filling forms an end-to-end transaction. Transactions are of different activity types if their shape sequences are not identical. Omitted processing steps are colored gray. We define end-to-end transaction processing that exactly follows documented desktop procedures to be normal. Patternized deviations from the normal cases are called variations, while noises are unpredictable fluctuations around the normal cases without pattern governance. Our pilot involves eight activity types, each of which has only one succession of standard processing procedures. In other words, the service delivery center defines one standard pattern for each activity type. However, it turns out to have 40 variational sequential patterns after learning.

Understanding the variations and noises we are facing, we design an effective approach as shown in Figure 2. The approach makes integrated use of knowledge

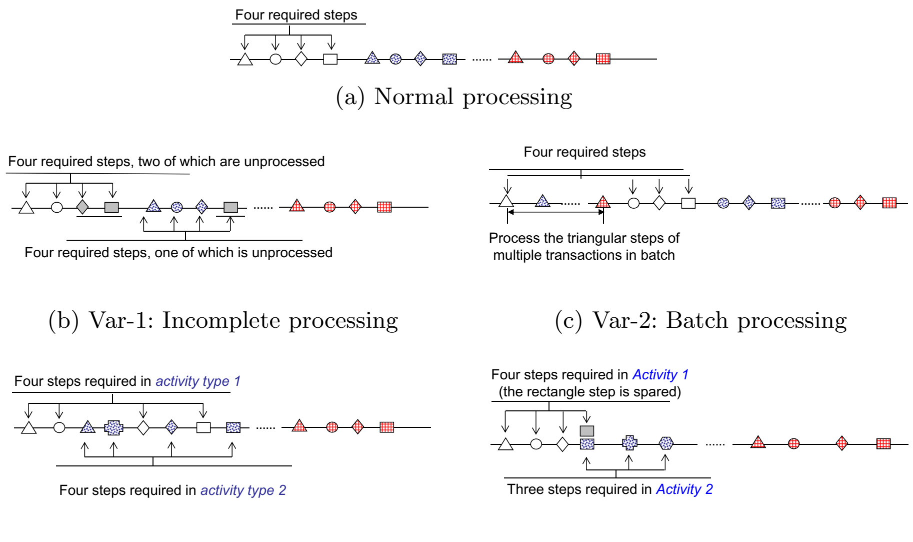
Table 1. Non-standard Human Behaviors

Human Behavior Variations

Index	Variation type	Description
Var-1	Incomplete processing	Only early or later steps of a transaction are spotted.
Var-2	Batch processing	Multiple transactions of the same activity type form a work unit, where some of the steps (featured application pages) are triggered in batch, for example, open ten invoices one after another before processing them one by one.
Var-3	Interleaved processing	Multiple transactions of different activity types form a work unit, where the required steps (featured application pages) of each transaction appear in an interleaving way.
Var-4	Combined processing	Multiple transactions of different activity types form a work unit, where not all the required steps (featured application pages) can be found for at least one of the activities.

Human Behavior Noises

Index	Noise type	Description
Noi-1	Repetitive visits	It is hard for a practitioner to complete everything on an application page without switches, and therefore non-deterministic number of visits on the same pages could occur.
Noi-2	Inadvertent clicks	It is possible to click some featured application pages of other activity types irrelevant to the transaction under processing. The inadvertent clicks put our approach in trouble to correctly label and time the actual activity being carried out.


Four required steps
Four required steps, two of which are unprocessed
Four required steps, one of which is unprocessed
Four required steps
Process the triangular steps of multiple transactions in batch
Four steps required in activity type 1
Four steps required in activity type 2
Four steps required in Activity 1 (the rectangle step is spared)
Three steps required in Activity 2

(a) Normal processing

(b) Var-1: Incomplete processing

(c) Var-2: Batch processing

(d) Var-3: Interleaved processing

(e) Var-4: Combined processing

Fig. 1. Graphical illustration of variations

discovery techniques including sequential pattern mining, scoring function, confidence interval. It follows the generic "training-testing" scheme, but attentive audience may notice that the testing phase, starting from *work unit partition*, is much more complicated than conventional approaches. The complexities result from *simultaneous classification and timing* of each transaction cycle, as well as the addressing of variations and noises as indicated in Table 1.

We also highlight that this approach embeds two sequential pattern mining modules for normal and deviated cases, respectively. Henceforth, we will use *deviated cases* to refer to *cases with human behavior variations* described in Table 1. The normal sequential pattern mining is conducted only once, with training data captured during the time veteran staffs, who strictly follow the desktop procedures, are doing their work. The training for deviated cases, on the contrary, requires iterative efforts, since it is impossible for practitioners to intentionally provide the full set of variations. We therefore incrementally expand the training data set for deviated cases with application events going through the tool but remaining unclassified. Note that practitioners usually will not diverge from the standard procedures unless shortcuts are discovered or un-documented exceptional requests enter. In the following, we are going to describe the key steps of the activity classifier and timer depicted in Figure 2.

Step 0 Initialization: The initialization includes *signature page identification* and *sequential pattern mining for normal cases*. A signature page meets two conditions. First it is a must that cannot be spared for a particular activity type, and secondly it can differentiate different transactions of the same activity type. In this work, we use the application page containing the unique transaction ID, or transaction ID page in short, as the signature page. Here we take advantage of the business process nature which is primarily transaction based. In sequential pattern mining, we always included the signature page as the first element in a pattern. We will not elaborate how we implement the "Apriopri-like" algorithm here. Interesting readers could refer to [13]. The initial sequential pattern mining extracts frequent sequential application patterns for normal transaction cycles. We initiate a toy example in the below, which will be used to demonstrate the flow of our approach, where each letter represents a different application page and letters with subscript (transaction ID) are the signature pages.

The problem setting of toy example

Normal pattern (learnt by mining)	$\{ABCD_xEFG\}$, $\{HI_xJ\}$, $\{ECD_xE\}$
Interleaved pattern (learnt by mining)	$\{ABCD_xHI_xEFGJ\}$, enclosed activities are $\{ABCD_xEFG\}$ and $\{HI_xJ\}$
Input sequence	$\{ECD_1EABCD_3HI_8EFGJ\}$

Step 1 Work unit partition: We partition the whole day application events by signature pages or signature page combinations, where the events around a signature page are considered plausible to belong to the same transaction. In our example, the three signature pages could partition the input sequence into three work units, i.e., $\{ECD_1EABC\}$, $\{EABCD_3H\}$, $\{HI_8EFGJ\}$. Alternatively, we can partition the sequence into two work units, i.e., $\{ECD_1EABC\}$ and $\{EABCD_3HI_8EFGJ\}$ since the "$D-I$" signature combination may form an interleaved work pattern learnt beforehand. Note that the applications between two signature pages (or signature page combinations) are considered possible to belong to both transactions at this phase. For instance, the $EABC$ between signature pages D_1 and D_3 are temporarily put into both work units. We point out that this step generates partitions in all the possible ways and feeds them into the following steps. In this example, we will generate partitions in two ways where one include three isolated work units, and the other include one isolated plus one interleaved work unit.

Step 2 Pattern testing and scoring: This critical step uses frequent patterns of normal and deviated cases to test if they occur around the signature pages or signature page combinations *appropriately.* The appropriateness means featured pages occur, and occur in the expected order. It is worth mentioning that we allow "partial matching," *which does not require a work unit containing 100% application pages in a pattern or arranging featured application events exactly as the pattern sequence.* We apply this principle because incomplete processing cases (Var-1) cannot have all the featured application events, while batch processing (Var-2) cannot guarantee that all the transactions contain all the featured events. In addition, this principle hedges against the risk of high generation error as a result of overfitting, which means that patterns learnt in the training phase might only work well for the training set, but are not superior for testing set.

However, "Partial matching" principle on the other hand causes the problem of having multiple candidates. It is common that activities processed by the same team (back office) are related with each other. The inherent relevance usu-

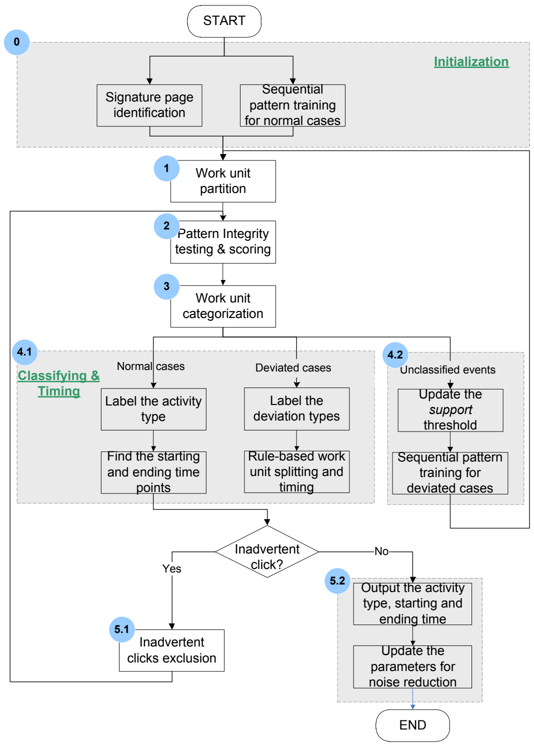

START
0
Initialization
Signature page identification
Sequential pattern training for normal cases
1
Work unit partition
2
Pattern Integrity testing & scoring
3
Work unit categorization
4.1
Classifying & Timing
Normal cases
Label the activity type
Find the starting and ending time points
Deviated cases
Label the deviation types
Rule-based work unit splitting and timing
4.2
Unclassified events
Update the *support* threshold
Sequential pattern training for deviated cases
Inadvertent click?
Yes
No
5.1
Inadvertent clicks exclusion
5.2
Output the activity type, starting and ending time
Update the parameters for noise reduction
END

Fig. 2. The work flow

ally leads to application sharing to some extent. For example, activity "invoice auditing" is a prerequisite of activity "invoice payment," and both activities visit the same application to retrieve invoices. Therefore, Application sharing plus "partial matching" principle implies multiple candidate patterns. We design score function

$$score = \frac{\text{number of appropriately matched pages}}{\text{number of pattern pages}} \tag{1}$$

to choose the best from the candidate patterns. Returning to the toy example, we test and score each candidate pattern as shown in Table 2.

Theorem 1. *When a transaction can either be extracted from an isolated work unit or part of an interleaved (combined) work unit obtained in the partition phase, the appropriately matched application pages in the former case will always be a subset of the later.*

The way we partition work units makes the Theorem 1 self-evident. In our example, the transaction $\{\underline{HI_8J}\}$ can be either extracted from the isolated work unit $\{HI_8EFGJ\}$ or the interleaved work unit $\{EABCD_3HI_8EFGJ\}$. And $\{HI_8EFGJ\}$ is a subset of $\{EABCD_3HI_8EFGJ\}$ as Theorem 1 states.

Theorem 2. *When a transaction can either be extracted from an isolated work unit or part of an interleaved (combined) work unit, it can always be split from the interleaved (combined) work unit with score not lower than score of the isolated case.*

Theorem 1 establishes that the number of appropriately matched pages split from the interleaved (combined) work unit must be equal to or more than the number of appropriately matched pages in the isolated page. Given the appropriately matched page count is the numerator in Equation (1), and the denominators are the same for both cases, we would know that Theorem 2 is valid.

Table 2. The example: pattern testing & scoring

Work unit	Pattern to test	Score
$\{ECD_1EABC\}$	$\{AB\underline{CD_xE}FG\}$	$3/7 = 42.9\%$
	$\{\underline{ECD_xE}\}$	$4/4 = 100.0\%$
$\{EABCD_3H\}$	$\{\underline{ABCD_x}EFG\}$	$4/7 = 57.1\%$
	$\{E\underline{CD_x}E\}$	$2/4 = 50.0\%$
$\{HI_8EFGJ\}$	$\{\underline{HI_xJ}\}$	$3/3 = 100.0\%$
$\{EABCD_3HI_8EFGJ\}$	$\{\underline{ABCD_xHI_xEFGJ}\}$	$10/10 = 100.0\%$

Step 3 Work unit categorization: We categorize work units by applying the below rules sequentially to compare across all the candidate patterns. Step 1 decides whether a work unit is a normal or a deviated case. Steps 2-4 defines the rules to eventually choose only one pattern which matches the work unit under concern to the best extent.

1. When a series of application events can be both a normal work unit or part of a deviated work unit, the "deviated type" will be prioritized because of Theorem 2.
2. Choose the pattern that has most number of appropriately matched application pages appear in the work unit under concern.
3. To break a tie in rule 2, select the pattern which has the highest matching score calculated via Equation (1).
4. To break a tie in rule 3, randomly select a candidate pattern.

For a normal transaction which has complete cycle, this step labels it with the activity type. For a deviated case, it points out the variation type and enclosed activities. For example, the work unit $\{EABCD_3HI_8EFGJ\}$ has variation type "interleaved work unit" and contains two interweaving activity types, $\{ABCD_xEFG\}$ and $\{HI_xJ\}$.

Let us still use the toy example for illustration. We go to rule 2 for the work unit $\{ECD_1EABC\}$ and associate pattern $\{ECD_xE\}$ with this work unit. Prioritizing $\{ABCD_xHI_xEFGJ\}$ over $\{ABCD_xEFG\}$ plus $\{HI_xJ\}$ according to rule 1, we label the work unit $\{EABCD_3HI_8EFGJ\}$ as an interleaved case.

Step 4.1 Classifying and timing for normal and deviated work units: For a normal case, we assign it with the activity type binding with the pattern selected in the step 3. To find the starting and ending time, we employ the *"farthest boundary" principle* which adopts the earliest starting page and latest ending page in the work unit to delimit the transaction cycle. This principle is designed to offset the noise brought by the fact that practitioners cannot always finish all the working on the same page without switching, or Noi-1 in Table 1. After splitting a deviated case into sub units according to signature and pattern pages, we similarly apply the "farthest boundary" principle to identify the tentative transaction starting and ending time.

Step 4.2 Label unclassified events and sequential pattern mining for deviated cases: This step invites domain experts or practitioners to look into the unclassified events to label them for training. The threshold of the support for the iterative mining is often adjusted by observing the variation frequencies. To illustrate this, let the complete cycle for activity 1 be $ABCDEFG$ and that for activity 2 be $HIJKLMN$. Assume there exist two common interleaved working styles, say, $ABCD|HIJ|EFG|KLMN$ and $ABCDE|HIJ|FG|KLMN$ which approximately make up for 75.0% and 12.3% of the interleaved cases, respectively. Then we could set the threshold of support no larger than 12.3% if the second variations deserve identification, otherwise we should raise the threshold to narrow the search space.

Table 3. Inadvertent click exclusion

```
For the transaction under concern, get its cycle time $cycleTime$.
  If ($cycleTime > maxSpan$)
    // Forward check
    Set $curPage = startPage =$ the signature page.
    While (A directly antecedent featured page relevant to the current activity type
    before $curPage$ exists)
      Set it to be $startPage$.
      Set $curGap$ to be time interval between $startPage$ and $curPage$
      If ($curGap < maxGap$)
        Set $curPage = startPage$
      Else
        Set $startPage = curPage$ and break while
      End If
    End While
    // Backward check is similar to the forward check, except for that we are trying to find the
    // $endPage$ instead of $startPage$. We omit the details for length constraint.
  End If
```

Step 5.1 Inadvertent click exclusion: We introduce two adaptive parameters, *maxSpan* and *maxGap*, to deal with the inadvertent clicking (Noi-2), given Noi-1 has been handled in Step 4.1 with the "farthest boundary" principle. Parameter *maxSpan* restricts the cycle time of a particular activity type, while *maxGap* confines the interval between two featured application pages under a certain threshold. A transaction cycle time exceeding *maxSpan* calls for a double check as a result of its abnormally longer processing time, implying that we probably incorrectly delimit or even misclassify the activity. We actuate the mechanism in Table 3 to reduce the noises. *maxSpan* strikes a balance between solution time against accuracy. An extremely large *maxSpan* is incapable to detect and exclude inadvertent clicks for more accurate timing, while a too small *maxSpan* leads to case-by-case checks, which are time-demanding.

Note that excluding inadvertent clicks might change the activity type, which is rare, though. Take two activity types with sequential patterns ABC_xD and BC_xD as an example. At first we classify the working time into type 1 because of rule 2, but the noise reduction phase excludes A, for it takes 15 min to arrive at B from A while $maxGap$ = 3 minutes. We thereof send the trimmed work unit to pattern testing and scoring (Step 2) for reclassification.

Step 5.2 Output and parameter update: This step outputs the class label, starting and ending time points of each transaction. Also, we update the *maxSpan* and *maxGap* by constructing one-sided confidence intervals and incorporating the newly identified transactions. Briefly speaking, we calculate, for each activity type i, the average cycle time $\bar{X}_i$ and the standard deviation σ_i. Let $Z = \frac{(\bar{X}_i - \mu_i)}{\sigma_i/\sqrt{n_i}}$, then $Z \sim N(0, 1)$, and

$$maxSpan_i = \bar{X}_i + z_\alpha \sigma_i/\sqrt{n_i},$$

where α is the significance level, n_i is the sample size, z_α is the value that makes $P\{Z \leq z_\alpha\} = 1 - \alpha$.

For *maxGap*, we need to compose it in real time in Step 5.1, since time interval between two featured pages differs among activities and applications. We calculate the upper bound for each application page similar to *maxSpan*, and sum the upper bounds of all the applications between two featured application pages to be *maxGap* in Table 3.

4 Performance Evaluation and Business Benefits

Through a pilot deployment of the activity classifier and timer to the production environment, we evaluate the tool performance in terms of *accuracy*. Further more, we share multifaceted business benefits realized by the tool, some of which are pleasant surprises beyond expectation.

4.1 Pilot Introduction

The pilot took place at a service delivery center dedicating to F&A (Finance & Accounting) processes. Out of dozens of activities, we only focus test our tool with eight activity types. It last 14 working days on four desktops (four practitioners) with 768 in-scope transactions processed.

In order to monitor the working time of a practitioner, the current practice of center is to enforce the self-report by practitioners, which encompasses *volume report* and *time report*. For volume report, practitioners upload the number of processed transactions by activity type and on daily basis as shown in Table

Table 4. Sample Volume and Time Reports

Sample Volume Report

Date	Activity Type	Volume
Nov 11, 2011	Invoice processing	20
Nov 11, 2011	Invoice payment	29

Sample Time Report

Activity Type	Start Time	End Time
Invoice processing	08:12:13, Nov 11, 2011	08:49:56, Nov 11, 2011
Invoice payment	08:49:56, Nov 11, 2011	09:12:00, Nov 11, 2011
Break	09:12:00, Nov 11, 2011	09:20:04, Nov 11, 2011
Invoice processing	09:20:04, Nov 11, 2011	09:51:17, Nov 11, 2011

4 (upper part). Upon switches between activities, the practitioners will manually log the beginning of "switching-to" activity and the ending time of the "switching-from" activity. Table 4 (lower part) shows what a time report looks like. We emphasize that practitioners trigger the manual timing upon activity type changeovers rather than transaction changeovers, which means multiple "invoice payment" transactions may be processed from 08:49:56, Nov 11, 2011 to 09:12:00, Nov 11, 2011 (the second row in the Sample Time Report).

To provide the baseline for us to compare the tool performance with, our business partner offered to audit the volume and time self-reports for three working days of the four practitioners. The auditing covers 276 transactions or 518 minutes working time. 518 minutes is far less than 12 working days (three days / practitioner * four practitioners) because we only have eight activities in scope and practitioners spent time processing other out-of-scope activities, too.

4.2 Performance Evaluation

We evaluate the performance of the tool on volume count and time capture, in correspondence with the volume and time self-report. The activity classifier and timer supports automatic volume report generation by counting the distinct transaction ID of each activity type. Table 5 (left part) shows the pilot results in terms of volume count. The tool faces two sources of errors, unclassified and misclassified transactions. For transactions which were indeed processed, but no learnt pattern matches this work unit with a satisfactory degree, we will have *unclassified transactions*. When a transaction is assigned with an incorrect label, we will have *misclassified transaction*. Misclassifications further breaks into two types - *misclassification (in scope)* refers to a incorrectly labeled transaction with its true class among one of the eight activities involved in the pilot; *misclassification (beyond scope)* refers to one with its true class out of the eight activities.

We measure the performance with *accuracy*, which is defined as

$$\text{accuracy} = \frac{\text{number of objects correctly classified}}{\text{total number of objects}}.$$

Table 5. Performance of automatic volume report

Items	Volume		Time	
	Self-report	Auto-report	Self-report	Auto-report
Correctly classified	274	271	407 min	498 min
Unclassified	2	4	91 min	15 min
Misclassified (in scope)	0	1	20 min	5 min
Misclassified (beyond scope)	0	0	85 min	26 min
Accuracy	**99.28%**	**98.18%**	**67.50%**	**91.54%**
Audited volume, time	276 transactions		518 min	

Our tool achieves 98.18% accuracy, which is slightly lower than the self-report accuracy 99.27%. The automatic report accuracy drops from 100% with four unclassified and one misclassified transactions. A deep dive into the four unclassified transactions reveals that they are exceptional cases, where no signature page such as the transaction ID page will appear. The signature-page-based nature of our approach brings about the failure. This discovery uncovers the problem with the process itself. Our business partner is happy that we expose the unnoticed non-standard processing. They will follow to redesign the process and make sure each transaction to have a signature page for monitoring purpose.

Next, we share the timing performance of our tool in Table 5 (right part). The correctly classified time is 498 min while total time under concern is 544 min. We can calculate that the accuracy is 91.54%, which is a significant improvement comparing with the self-report accuracy, 67.50%. We have discussed the root causes of low accuracy of self volume reports with our business partner, who believes too frequent manual time logging distracts practitioners from their normal work flow and practitioners are reluctant to do so. We see it as an opportunity where the automatic tool helps, since the charge model of our business partner depends on working time on different activities.

The above numeric results illustrate that the activity classifier and timer performs well in terms of both automatic volume and time reports. Next, we are going to demonstrate the effectiveness of our approach design (Figure 2) with *"what-if" scenarios.*

Require complete pattern (RCP)	If a work unit do not contain all the application pages appropriately of at least one pattern, it remains unclassified.
Neglect variations (NV)	Do not handle the variations.
Neglect inadvertent clicks (NIC)	Do not address the inadvertent clicks (Nor-2).
Neglect repetitive visits (NRV)	Do not address the repetitive visits (Nor-1).

Table 6 (left part) shows the results of the above four scenarios in terms of transaction count accuracy.

We can observe that the requirement of complete pattern matching leads to more unclassified transactions, which meets our intuition. The variation neglect often results in fewer classified transactions because we fail to split the interleaved or combined work units. Very seldom misclassified cases can occur due to

Table 6. What-if analysis of automatic volume report

Items	Volume				Time			
	RCP	NV	NIC	NRV	RCP	NV	NIC	NRV
Correctly classified	215	243	271	271	422	466	503	446
Unclassified	61	32	4	4	88	40	12	68
Misclassified (in scope)	0	1	1	1	8	12	3	4
Misclassified (beyond scope)	0	0	0	0	24	28	1345	10
Accuracy	**77.9%**	**88.4%**	**98.2%**	**98.2%**	**77.9%**	**89.0%**	**27.0%**	**84.5%**
Audited volume, time	276				518 min			

unaddressed inadvertent clicks, and we do not encounter it in our pilot. Repetitive visits handling cannot improve the automatic volume report accuracy as we can expect. Hence, the performance of NIC and NRV are identical to the activity classifier and timer. With regard to timing accuracy, we also analyze the four what-if scenarios to validate our algorithm effectiveness and the indispensability of each step in Table 6 (right part).

We could observe that we have the largest amount of unclassified time due to unclassified transactions when complete pattern matching required (RCP bar). When we do not handle the variations, the unclassified time also has a slight rise which indicates the practitioners did some combined or interleaved processing during our pilot. It is interesting that if we do not exclude the advertent clicks, we end in extremely low accuracy since the tool include too many out-of-scope time because of inadvertent clicks on some featured pages. If we do not consider the possibility of visiting the same page repetitively, we will have more unclassified time because we only start from the latest starting application page and end with the earliest ending application page.

4.3 Business Benefits

Section 4.2 demonstrates the high performance of the tool in classification and timing, and how the performance will be degraded without critical steps. This section shares the business benefits we obtained. A straightforward reap by enabling the automatic tool is the time saving in manual time and volume report, which on average costs a practitioner 13.29 min per day (2.7%) assuming the total working time is 8 hours per day. We can also save the auditing time of team leads.

Next, we observe an obvious behavioral change of the practitioners who has the tool deployed. The waste (consisting of machine idle and keyboard idle) when processing the eight monitored activities is much lower than that when processing out-of-scope activities. Table 7 shows our findings. If we extrapolate the saving to an eight-hour working time, the monitored practitioners should waste about 22 minutes but unmonitored ones has about 55 minutes to squander - 30 minutes to save per person day. Additionally, the efficiency has risen for the eight in-scope activities as compared to the pre-pilot period as shown in Table

Table 7. Business Benefit: Waste and Efficiency

Waste: monitored versus unmonitored activities

	Monitored activities	Unmonitored activities
Waste (%)	4.73%	11.48%

Efficiency: pre-pilot versus pilot period

	Pre-pilot period	Pilot period
Efficiency (%)	90.61%	109.94%

7. The waste reduction and efficiency improvement, we believe, attributes to the practitioners' awareness of the "being watched."

Finally, the iterative training mechanism in our approach helps our business partner to unveil non-standard or exceptional processing procedures. Non-standard processing, such as processing in batch, interleaved and combined processing, tend to have strong reasons in behind. The practitioners learn to do their work faster by combining or interleaving two related activity types together to omit some steps, which we call *shortcut-driven*. It is also quite common to do batch processing, since repetition creates efficiency with familiarity to particular working contents and savings of changeover costs, which we call *familiarity-driven*. The shortcuts discovered by practitioners may motivate the business to redesign and document more efficient processing procedures. Moreover, the unclassified time implies *exceptional cases and should rouse the attention of the team leads for further investigation.*

5 Conclusions

In this paper, we present an approach and tool that automatically classifies and times the transactions processed by practitioners, based on the fine-granular application usages. This work contributes to the existing literature with a streamlined approach which comprehensively consolidates knowledge discovery & data mining techniques, and furthermore handles the typical human variations and noises in the service delivery process with domain knowledge.

A pilot with a world-class BPO provider showed a success in both technology and business. Our approach results in high accuracy in both classification and timing. The critical part is to be able to address human behavior variations and noises for service delivery. Business wise, the tool can not only eliminate the self-report efforts, it also discipline practitioners' behavior to improve efficiency and reduce waste. Lastly but not least, it can discover non-standard or exceptional operations to enforce the process standardization and business control.

References

1. Computer Business Review (2011), `http://outsourcingbpo.cbronline.com/news/worldwide-bpo-market-to-grow-by-63-in-2011-gartner-230811`

2. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the 11th International Conference on Data Engineering, pp. 3–14 (1995)
3. Bharadwaj, S.S., Saxena, K.B.C., Halemane, M.D.: Building a successful relationship in business process outsourcing: An exploratory study. European Journal of Information Systems 19(2), 168–180 (2010)
4. Bozorgi, M., Saul, L.K., Savage, S., Voelker, G.M.: Beyond heuristics: Learning to classifify vulnerabilities and predict exploits. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 105–113 (2010)
5. Brown, R.H.: Business process outsourcing vendor consolidations: Is your contracts at risk? Gartner (2009)
6. Du, Z., Liao, X.: Well-defined processes and their effects on business process outsourcing vendor's success: An integrated framework. In: The 2010 International Conference on E-Business Intelligence, pp. 27–36 (2010)
7. Forman, G., Rajaram, S.: Scaling up text classification for large file systems. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 239–246 (2008)
8. Han, J., Pei, J., Mortazavi-AsI, B., Chen, Q., Dayal, U., Hsu, M.: FreeSpan: Frequent pattern-projected sequential pattern mining. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 355–359 (2000)
9. Karu, K., Jain, A.K.: Fingerprint classification. Pattern Recognition 29(3), 389–404 (1996)
10. Lacity, M.C., Willcocks, L.P., Rottman, J.W.: Global outsourcing of back office services: lessons, trends, and enduring challenges. Strategic Outsourcing 1(1), 13–34 (2008)
11. Nenkova, A., Bagga, A.: Email classification for contact centers. In: Proceedings of the 2003 ACM Symposium on Applied Computing, pp. 789–792 (2003)
12. Pei, J., Han, J., Mortazavi-AsI, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.: PrefixSpan: Mining sequential patterns efficiently by prefixed-projected pattern growth. In: Proceedings of the 17th International Conference on Data Engineering, pp. 215–224 (2001)
13. Qin, T., He, M., Zeng, S., Ren, C., Dong, J.: An effective pattern mining algorithm to support automatic process classification in contact center back office. In: Proceedings of the 2012 IEEE SOLI (2012)
14. Srikant, R., Agrawal, R.: Mining sequential patterns: Generalizations and performance improvements. In: Proceedings of the 5th International Conference on Extending Database Technology, pp. 3–17 (1996)
15. Tan, P.-N., Steinbach, M., Kumar, V.: Introduction to Data Mining. China Machine Press (2010)
16. Tang, M., Pellom, B., Hacioglu, K.: Call-type classification and unsupervised training for the call center domain. In: Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding, pp. 204–208 (2003)
17. Tapper, D.: Worldwide and U.S. IT outsourcing services 2004-2008 forecast: A potential perfect storm. IDC # 31089 (2004)
18. Tapper, D.: U.S. customers select IBM, HP-EDS, Unisys, Accenture, Infosys, ADP, and Fedelity as top 5 ranked BPO vendors for transformation, integration, innovation and cost optimization - excerpt from IDC # 216191. IDC # 216191 (2009)
19. Zaki, M.J.: SPADE: An efficient algorithm for mining frequent sequences. Machine Learning 42(1-2), 31–60 (2001)