Automatic Discovery of Service Name Replacements Using Ledger Data

Suppawong Tuarob, Conrad S. Tucker
Computer Science and Engineering
Industrial and Manufacturing Engineering
Pennsylvania State University, PA 16802 USA
{*suppawong,ctucker4*}*@psu.edu*

Ray Strong, Jeannette Blomberg, Anca Chandra,
Pawan Chowdhary, Sechan Oh
Services Research, IBM Almaden Research Center
San Jose, CA 95120 USA
{*hrstrong,blomberg,anca,chowdhar,seoh*}*@us.ibm.com*

***Abstract*—Recent studies have illustrated historical financial data could be used to predict future revenues and profits. Prediction models would be accurate when long-run data that traces back for multiple years is available. However, changes in service structures often result in alteration of the nomenclatures of the services, making the streams of financial transactions associated with affected services discontinue. Manually inquiring the history of changes can be tedious and unsuccessful especially in large companies. In this paper, we propose a machine learning based algorithm for automatically discovering service name replacements. The proposed methodology draws heterogeneous features from financial data available in most ledger databases, and hence is generalizable. Our proposed methodology is shown to be effective on ground-truth synthesized data generated from real-world IBM service delivery ledger database.**

***Keywords*-Service Name Replacement; Classification; Machine Learning;**

I. Introduction

A service delivery project consists of two phases: the sale engagement and the delivery phases. The former typically involves the service provider reviewing the requirements and offering the costs of the necessary services. After the deal is won, the contractually promised services are delivered. Performance data associated with service delivery is often recorded with various taxonomic classifications, which are referred to as *names* throughout the paper. In most cases, names are also used as *IDs* of the corresponding services or components. For example, a unique name associated with a database maintenance service (e.g. `<DATABASE,LABOR,MAINTENANCE>`) may be comprised of a category name (`DATABASE`), a service type (`LABOR`), and a service component (`MAINTENANCE`).

While the services are being delivered, associated costs and revenues are recorded in the ledger database (usually monthly). Each transaction (either cost or revenue) is usually accompanied with the associated service name. This ledger data not only serves the traditional accounting purposes, but also behaves as a valuable historical data for service and financial researchers in various applications such as profitability forecasting [1], stock market volatility prediction [2], and workload forecasting in business processes [3]. All such applications rely on long run historical data to build accurate prediction models.

However, business processes often result in changes in the service nomenclature systems. For example, a business may have used `<DATABASE,LABOR,MAINTENANCE>` to name an on-site maintenance service. Later, when an off-site maintenance is offered as an additional service, `<DATABASE,LABOR,MAINTENANCE>` may be changed to `<DATABASE,LABOR,ONSITE MAINTENANCE>` to better distinguish the traditional on-site service from the new off-site one. Furthermore, changes in internal business structure (department division/merging) could also result in alteration of the naming system. In order to construct accurate models of service delivery performance from long running historical data, these name replacements must be recognized so that data associated with a replacing name is viewed as a continuation rather than a beginning.

Other variants of the same core problem include tracking historical information of contracts. A single service delivery project may have been signed in multiple different contracts with different contract IDs due to various reasons such as changes in terms of agreement, contract renewals, or simply amends of required services. These different contracts would be viewed as separate service delivery projects. The ability to trace and link these distinct contracts that belong to the same projects together would allow more effective pricing schemes to be generated.

A proactive solution would be to have a dedicate database that keeps track of the history of name replacement events. However, not all companies follow this protocol. Another naive solution would be to directly consult experts in the accounting department to inquire the name replacement history. However, such approach can be unsuccessful due to:

- Histories of name changes may not be systematically kept track of.
- Most accounting practices only care about recording the correct amounts. This results in unaccountability of service name changes, especially the change history in multiple years back.
- The number of name replacements can be so large that consulting the accounting experts would become infeasible.

Known quantitative solutions involve comparing average

978-1-4673-7281-7/15 $31.00 © 2015 IEEE
DOI 10.1109/SCC.2015.90

performance before and after a possible event of a name replacement, where averages are taken over all relevant data for the two names, or over data from a fixed duration window. In this paper, we show that such a method could fail to capture the correct name replacements in service delivery settings where volatility is a norm, and client/contract dependency could often cause unpredictable movements in the cost/revenue associated with a particular service.

The problem of discovering name replacements is framed into a classification problem, where a pair of two names is classified whether it is a correct replacement or not. We propose a machine learning based classification methodology that draws continuity statistics from multiple aspects of the transaction time series data available in ledger databases. During our investigation, we notice that the financial data is only useful during a specific window surrounding the name replacement event. Hence, we also propose two approaches to dynamically determine the size (window) that limits the data to be processed. The proposed methodology is generalizable and scalable since the features could be extracted from common ledger data. Listed below are the main contributions of this paper:

1) We formalize the problem of identifying service name replacements into a classification problem.
2) We propose a novel set of features that characterize name replacements. These features are drawn based on financial information available in common ledger databases.
3) We validate our proposed methods using rigorous evaluation techniques on synthetic data generated from real-world service delivery transaction records.

II. Definitions and Problem Statement

To simplify the ledger structure, let:

- $\mathbb{T} = \{t_1, t_2, ..., t_{|\mathbb{T}|}\}$ be the time space, where t_i denotes an equal-length time interval, and $t_i < t_{i+1}$. Normally, t_i means the i^{th} month of the ledger entries.
- $\mathbb{C} = \{c_1, c_2, ..., c_{|\mathbb{C}|}\}$ be the set of contracts.
- $\mathbb{G} = \{g_1, g_2, ..., g_{|\mathbb{G}|}\}$ be the set of names. A name g is a triple of $\langle ID, Y, T\rangle$. ID is the textual representation of g's name. Y is the transaction stream data associated with g, and defined on each time $t \in T$. That is $Y = \{y(t) : t \in T\}$.

A window $w = \{t_{start}, t_{start+1}, ..., t_{end}\}$, where $t_{start} \leq t_{end}$, defines the time interval of $t_{end} - t_{start} + 1$ periods. A ledger entry $e(g, c, t)$ represents the $amount$ (in currency unit) associated with a service name g under the contract c at time period t. Alternatively, c is said to *cover* g at time t. Specifically, $CoveringContracts(n, w)$ returns a set of contracts $C' \subseteq \mathbb{C}$ each of which covers the service name g during the window period w.

Note that, though other useful pieces of information such as component descriptions and logs of name replacement events may be available, a majority of companies do not have such information. Consulting experts for such information is also infeasible since the amount of inquiries would be large, and they may not have complete knowledge about name replacement events that took place long time ago.

If a service name g_b stops being used at time period t^{crit} (t^{crit} is said to be the *critical period* of g_b), and shortly after (typically within the next few time periods) a new service name g_a comes to exist, a natural curiosity would be to ask whether g_a is a replacement of g_b. (Note that `b` stands for *before* and `a` stands for *after*). We formalize this question to a classification question. Specifically, given a name replacement candidate $p = \langle g_b, g_a\rangle$, our task is to build a classifier that classifies p as $positive$ if g_a is the correct replacement of g_b, and $negative$ otherwise.

III. Related Literature

To the best of our knowledge, we are the first to address the problem of automatic detection of name replacements in business processes. Though entity resolution, where pieces of information that belong to the same entity are linked together, would be a similar class of problems, relevant literature on such problems pertains to entities whose underlying information is not time series (i.e. persons, books, webpages, scholarly papers, etc.) [4]. Hence, existing methods for entity resolutions would be irrelevant to our research.

Since the name replacement detection problem is converted to a classification problem where the features are based on continuity of financial time series data, relevant literature on such continuity measurements is discussed. Multiple measurements for continuity or smoothness in financial time series data were used in the literature including autocorrelation and coefficient of determination (R^2). Autocorrelation is used to detect non-randomness in data, and has been applied to measure smoothness in time series data [5]. R^2 indicates how well an observed time series data fits a statistical model, and has been widely used to measure the fitness in financial time series models [6]. In this paper, we show that these measures alone could not be used directly to capture continuity in our dataset due to high level of noise and volatility characterizing service delivery transaction streams. A more advanced technique to quantify connectedness in financial time series data was introduced by Ganeshapillai et al. where a regressor is trained with active return, market sensitivity, and return connectedness features [7]. Though their method was reported successful, the proposed features are only specific to stock market data, which has different nature and properties from transactional data (i.e. cost and revenue) in ledgers.

Though extensive literature has devoted to financial time series analysis [8], most techniques assume that the time series data is whole–different time series curves belong to different entities. However, this paper takes into the account that pieces of time series data with different names may

belong to the same entity (service), and proposes a set of methods to link them.

IV. Feature Extraction

The name replacement detection is converted to a classification problem where a candidate replacement (i.e. $\langle g_b, g_a \rangle$) is classified whether it is positive or negative. From observation, financial time series can be unpredictable, and hence no single set of rules can be applied to determine their continuity. Literature has shown the use of data driven techniques (especially machine learning based ones) which are capable of learning representative *signals* from the training data to automatically build a model that captures the variations within such a dataset [9], [10], [11]. Here, we propose to train a machine learning based classifier with heterogeneous features drawn from the training data. These features are designed to determine whether two pieces of transaction data belong to the same service, and are discussed in the following subsections.

A. Time Series Continuity Statistics

Ledger data is a time series data where costs and revenues associated with each service are recorded every month. We observe that, whenever a name replacement occurs, the transactions associated with the new name start appearing within a few months. Based on this observation, if a candidate $p = \langle g_b, g_a \rangle$ is the correct name replacement, then the joint time series of the financial transactions of these two names must be continuous and exhibit reasonably the same trend (both volatility and amplitude). Given a candidate $p = \langle g_b, g_a \rangle$ and a window w which limits the scope of the financial data, three statistics are computed to measure the *continuity* of the two curves Y_b and Y_a (associated to g_b and g_a respectively): *Coefficient of Determination (R^2)*, *Average Similarity (AS)*, and *Adjusted Average Similarity (AAS)*. These statistics are based on basic intuition and are computed very fast (time complexity of $O(|w|)$. Since the system could be used to process millions of candidates, it is crucial that the continuity statistics that we develop here can be computed fast.

Let the function $aggregate(Y_b, Y_a) \rightarrow Y_{ba}$ returns a summed curve of Y_b and Y_a.

1) Coefficient of Determination (R^2): We make the case that if g_a replaces g_b, then the unified stream of transactions corresponding to g_b and g_a would appear seamlessly as the stream of a single service, and should not be visually distinguishable around the critical period. We notice that during a short period of time, the behavior of transaction stream associated with a service is linear with some degree of noise and volatility. The R^2 statistic determines how well $aggregate(Y_b, Y_a)$ exhibits a linear trend.

Let $Y = \{y(t) : t \in T\}$ be the set of observations, where $|Y| = n$ and $\overline{y}$ is the mean. Let $F = \{f(t) : t \in T\}$ represents the set of modelled values generated by fitting a linear regression model that minimizes the least square error. The R^2 statistic used in this research determines how well the time series data exhibits a linear trend once they are combined, and is defined as:

$$R^2(Y,F) = 1 - \frac{SS_{res}(Y,F)}{SS_{tot}(Y)} \quad (1)$$

$$SS_{res}(Y,F) = \sum_{t \in T} (y(t) - f(t))^2 \quad (2)$$

$$SS_{tot}(Y) = \sum_{t \in T} (y(t) - \overline{y})^2 \quad (3)$$

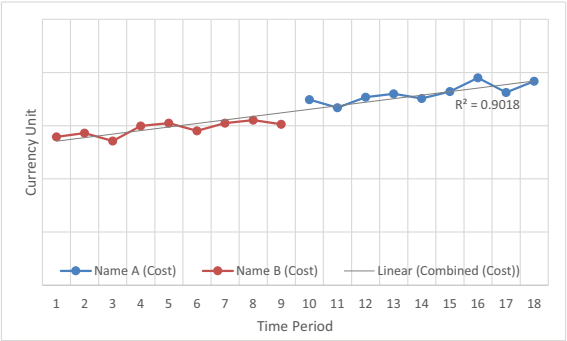

Figure 1. R^2 statistic measures how well the model (linear regression) fits the observations. Oftentimes, transaction streams exhibit linear trends. Hence, if two names are associated to the same service, then it would likely be possible to fit a linear regression model into the observations. R^2 statistic captures such a characteristic.

$SS_{res}(Y,F)$ is the sum of squares of residuals. $SS_{tot}(Y)$ is the total sum of square, proportional to the sample variance. $R^2(Y,F)$ returns a real number between 0 and 1, determining how well the linear regression model F fits the observations Y.

Figure 1 illustrates an example of the financial transaction streams associated to a *before* (B) and the corresponding *after* (A) names. The straight grey line is the fitted linear regression model. From the figure, a high R^2 of 0.90 is observed.

2) Average Similarity (AS): R^2 is a good statistic for steady streams with low volatility such as revenues from which a constant amount of money is collected from a contract every month. However, some financial streams are highly volatile as often reflected by costs associated with a particular service. However, provided that the foundations of a service (such as number of contracts/clients) remain the same, then the average transactions of the *before* and *after* names should not differ much. The *Average Similarity (AS)* is developed under this intuition.

Let $Y_b = \{y_b(t) : t \in T_b\}$ and $Y_a = \{y_a(t) : t \in T_a\}$ represent the two curves to be examined for continuity. The *AS* statistic is defined as:

$$AS(Y_b, Y_a) = \begin{cases} 0 & ;if\ \overline{Y_b} = \overline{Y_a} = 0 \\ 1 - \frac{|\overline{Y_b} - \overline{Y_a}|}{|\overline{Y_b}| + |\overline{Y_a}|} & ;otherwise \end{cases} \quad (4)$$

Where $\overline{Y_b}$ and $\overline{Y_a}$ represent the means of observations Y_b and Y_a respectively. $AS(Y_b, Y_a)$ returns a real number between 0 and 1, indicating the level of similarity between Y_b and Y_a.

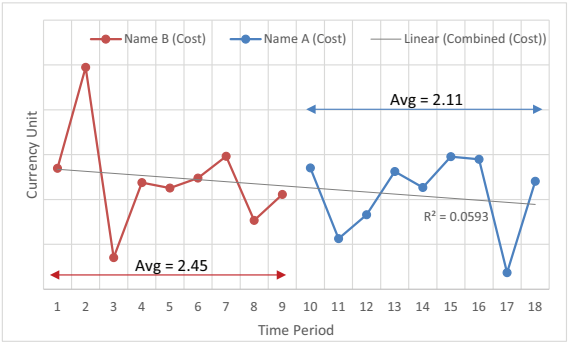

Figure 2. *Average Similarity (AS)* statistic quantifies the similarity in the overall transaction amounts of the *before* and the *after* names. This statistic is suitable for services with high volatility (low R^2), but steady trends (i.e. no sharply decreasing or increasing).

Figure 2 illustrates two transaction streams of a service that appear under two different names. For this particular example, the volatility of the data is high making R^2 (= 0.06) statistic infeasible to capture the continuity. However, the *AS* statistic for these two curves is 0.93, which is more suitable for services that exhibit high volatility.

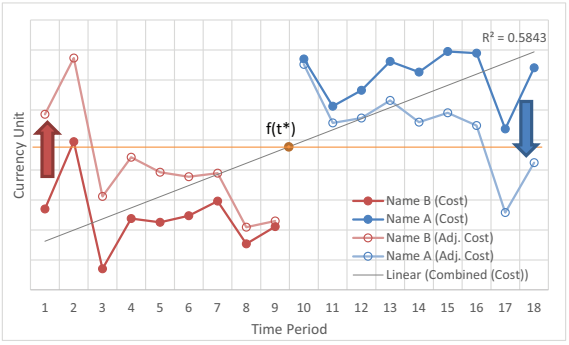

Figure 3. *Adjusted Average Similarity (AS)* statistic combines the benefits of both R^2 and *AS*, by rotating the transaction stream so that the slope of the fitted linear regression line is zero. This allows *AS* measure to capture better similarity for services with sharp rises or falls (due to contract terminations or initiations).

3) Adjusted Average Similarity (AAS): Oftentimes, a sudden change in the overall amount (increase or decrease) can be observed in a transaction stream. The causes of these changes include the initiations or terminations of contracts, which are usually a causal effect of business processes. New (possibly big) contracts are signed, or existing contracts are terminated. This will cause the aggregate transaction stream associated with a particular service to suddenly rise/fall over a short period of time.

The sudden changes in transaction streams result in low R^2 scores since the data cannot be described by a regression model. The changes also result in poor AS scores, since the difference in the amounts before and after the change point are too large.

To get around this problem, a novel measure called *Adjusted Average Similarity (AAS)* is developed. The idea is to adjust the transaction stream by rotating all the data points around the *pivot* such that the fitted regression model has a slope of zero. Figure 3 illustrates the idea. The solid red/blue lines are the actual transaction streams associated with the *before*/*after* names of the same service. The R^2 and AS computed from these actual streams are 0.5843 and 0.5125 respectively, which are too low to be conclusive. However, the AAS score is 0.8264, which is high enough for a decision maker to conclusively determine that these two curves belong to the same service.

B. Choosing the Right Window

Computing the continuity statistics from the entire data may violate the linearity assumption and could introduce bad knowledge to the classifiers [12], since transaction streams exhibit unpredictable behaviors in the long run, due to multiple factors such as business plans, market situations, etc. In this paper, we propose a set of methods for dynamically choosing a suitable window for a candidate so that the continuity statistics computed from the data limited to within such a window are meaningful and representative.

Mathematically, given a candidate p, we would like to find a window $w = \{t_{start}, t_{start+1}, t_{start+2}, ..., t_{end}\}$ where $t_{start}, t_{end} \in \mathbb{T}$ and $t_{start} \leq t_{end}$, such that the continuity statistics computed from $Y_b = \{y_b(t) : t \in T_b \cap w\}$ and $Y_a = \{y_a(t) : t \in T_a \cap w\}$ are high when p is actually a true name replacement candidate, and low otherwise. We propose to use two methods for selecting the windows: *Scanning Based Window Selection (SW)* and *Segmentation Based Window Selection (SG)*.

1) Scanning Based Window Selection (SW): The *SW* window selection algorithm assumes that financial time series exhibits a linear trend during a specific time period. The algorithm hence finds the window w^* wherein the unified transaction stream gives the best continuity statistics by exhaustively examining all possible window sizes.

Given a candidate $p = \langle g_b, g_a \rangle$ and an objective function H, the algorithm computes the optimal window w^* that maximizes H. For our research purposes, we tried all the continuity statistics for H, and found that the combination of R^2 and AS scoring functions works best.

2) Segmentation Based Window Selection (SG): A family of segmentation algorithms that are useful in modeling the time series data we work with was introduced in [1]. The algorithm family is indexed by a positive integer neighborhood size. The algorithm segments the time series based on the numeric valued associated with each element. A

time series element is a provisional peak if (1) its value is $\leqslant$ the values of each element of the neighborhood of preceding elements and each element of the neighborhood of succeeding elements, and (2) its value is $>$ the value of its immediate successor. Analogously, a time series element is a provisional trough if (1) its value is $\leqslant$ the values of each element of the neighborhood of preceding elements and each element of the neighborhood of succeeding elements, and (2) its value is $<$ the value of its immediate successor. These definitions are used in left to right processing of time series. The asymmetry selects the beginning (left end) of a set of equal values as the provisional peak or trough. In the general segmentation algorithms, a provisional peak (trough) becomes a peak (trough) if the next provisional peak or trough is a provisional trough (peak) respectively. The set of peaks and troughs form the boundary elements of a segmentation of the time series. These boundary elements alternate between peaks and troughs and each segment can thus be designated as either rising or falling. Moreover, the provisional peaks and troughs are very easy to compute.

The SW window selection method can be significantly time consuming, so we tried an alternative method for selecting a window based on finding at most two provisional peaks (or troughs). Starting from the change point, we find the first provisional peak or trough to the right. This becomes the right end point of the window. If no provisional peak or trough is found to the right of the change point, then the right end point of the window is the right end point of the time series. The left end point of the window is found in the exactly symmetric way, proceeding to the left. Note that this means the provisional peak or trough on the left will be the right most of a set of equal values or none of them. Note also that the window may be of the form of one rising or falling segment or a pair of segments with the change point as their shared boundary. The characteristic of a member of our algorithm family is the neighborhood size. Increasing the characteristic tends to increase the window size by ignoring minor fluctuations in the time series values. Characteristic 2 is reported in [1] as best for use in predicting future values of our financial time series; so we chose a neighborhood size of 2 for our experiments with this much faster way to select the relevant window of data.

The following example illustrates selecting a window based on the segmentation algorithm:

1	4	3	4	5	7	4	3	4	5
left end of window					change point		right end of window		

C. Using Contractual Information

A simplified model of service delivery involves contractual relationships between a supplier and a set of clients. Each client signs a set of contracts. A contract specifies a set of services that the supplier has to deliver to the client at each time period, along with the amount of money that the client is obligated to pay for each service (revenue). In each contract, the cost associated with a service is not specified and is determined when the requested services are delivered. This flexibility gives the supplier time to work on minimizing costs.

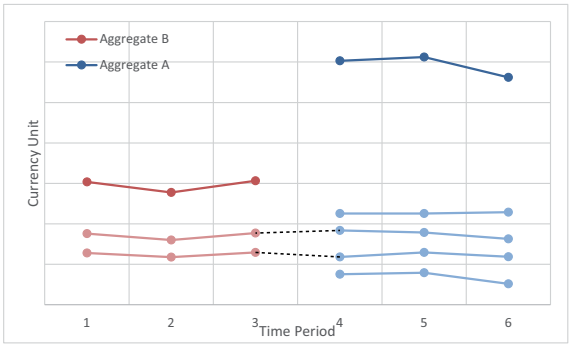

Figure 4. A name replacement event may co-occur with contract termination/initiation events. This would result in a sudden change in the level of amounts associated with a service. Contracts that cover a name change event usually provide useful information.

Using only the aggregated transaction streams of two names to determine whether they belong to the same service may not suffice, since some services may behave similarly to each other. This problem can become aggravate when the transaction amounts are very small and volatile, in which case it would be difficult to distinguish a service from others which behave similarly.

Hence, when the information from aggregate level is insufficient, the information from the contract level could act as supportive information. We make a hypothesis that if the name g_a is the correct replacement of the name g_b for a service s, then if a contract contains g_b and has not been terminated after g_b stops being used, then the transaction that appears under the name g_a must also be recorded under such a contract. Figure 4 illustrates an example of a set of transaction streams of a service which is recorded under two names, B and A. A pale line represents a stream associated with a contract. A dash line links two curves which belong to the same contract. The solid lines are the aggregate streams. It is shown that the aggregate amounts recorded under the two names are very different, resulted from new contracts being initiated. Hence, it would be hard to tell that A is the replacement of B if only the aggregate information were used. It is our conjecture that the information from the individual contracts could be critical and useful. Hence, the continuity statistics are computed not only at the aggregate level, but at the contract level as well.

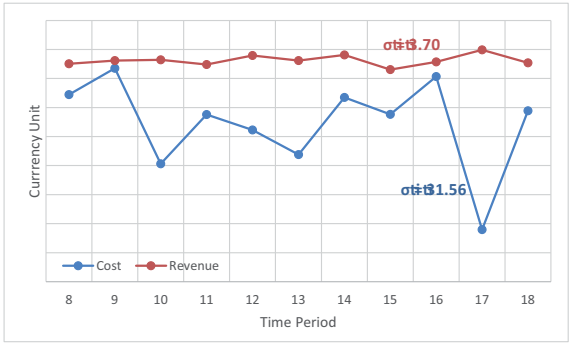

Figure 5. Example revenue and cost streams of a service. The cost stream is more volatile with $\sigma = 31.56$, while the revenue stream has $\sigma = 3.70$.

D. Types of Transaction

A transaction entry can be either a revenue or a cost. Figure 5 illustrates examples of cost and revenue streams of a service. A revenue stream tends to be smoother and steadier, compared to a cost stream. This is because, a client is usually required to pay a constant amount of money in each time period as specified in the contract. On the contrary, the cost stream can be highly volatile and less predictable due to the following reasons:

1. Service projects are different. Each project has different demands for one-time and long-run services. One-time services, such as installing hardware or software, may cause fluctuation in the stream especially at the beginning of the contract period.

2. Suppliers work hard on minimizing costs. This minimization process is dynamic and performed even after the contract is signed, and would cause the costs associated to a particular service to fluctuate.

In most cases, revenue streams tend to be better information to identify name replacements. However, there are cases where the revenue data is absent or non-reliable. In these cases, cost streams can be supportive information.

E. Summary of Extracted Features

Table I
THE PROPOSED FEATURES CAN BE DIVIDED INTO 8 GROUPS RESULTING FROM THE COMBINATION AMONG HIERARCHY OF TRANSACTION (I.E. *GLOBAL* AND *CONTRACT*), WINDOW SELECTION METHODS (*SW* AND *SG*), AND TYPES OF TRANSACTIONS (*COST* AND *REVENUE*).

	Scanning Based Window (SW)		Segmentation Based Window (SG)	
	COST	REVENUE	COST	REVENUE
GLOBAL	6	6	3	3
CONTRACT	32	32	17	17

For each candidate, 116 numeric features are extracted. These features are the combination of the window selection scheme (i.e. *SW* and *SG*), hierarchy of information (i.e. *GLOBAL* and *CONTRACT*), and transaction types (i.e. *COST* and *REVENUE*). The break-down summary of these 116 features are shown in Table I. Note that SW based feature space is twice as big as the SG based one because, the we use both R^2 and AS statistics as the objective functions (H).

V. Experiment, Results and Discussions

The following sub-sections will describe the synthesized dataset, the experiment and evaluation, and discussion of the results.

A. Synthesized Dataset

Table II
STATISTICS OF THE SYNTHETIC GROUND-TRUTH SAMPLES GENERATED FROM THE REAL-WORLD IBM STRATEGIC OUTSOURCING SERVICES IN FOUR GEOGRAPHICAL REGIONS.

Region	#Positive	#Negative	Region	#Positive	#Negative
A	308 (4.13%)	7150	C	201 (5.92%)	3196
B	513 (3.27%)	15156	D	538 (2.92%)	17888

The synthesized ground-truth data is used to validate the classification models. The positive samples are generated by breaking up an existing transaction stream, and assigning different names to them. The negative samples are generated by cross-matching the *before* stream with another *after* stream at a given time period.

The data used in this paper is generated from the IBM Strategic Outsourcing services where the data were collected from four geographical regions (i.e. *A, B, C,* and *D*). For each region, 25% of the all synthetic samples are randomly selected to include in the validation dataset to avoid possible bias. Table II lists the numbers of positive and negative samples of each region.

B. Experiment Setup

For each region, a 10 fold stratified cross validation is performed. First the positive and negative samples are split up into 10 equal subsets. For each fold i, the positive and negative samples in the subset i are combined used as *testing* set, and the rest of the data is used as the *training* set. The process is repeated for $i = 1, 2, ..., 10$, and the results are averaged. All the experiments are performed on a Windows machine with Intel Core i7 processor and 16GB of RAM.

C. Baseline Features

Our proposed features are tested against the baseline feature set, where the continuity statistics (i.e. R^2, AS, and AAS) are computed from the entire data (both *COST* and *REVENUE*) at the aggregate level. That is, for each candidate $p = \langle g_b, g_a \rangle$, the baseline features use the window $w_{baseline} = g_b.T \cup g_a.T$, and no contractual information is used.

The baseline features are only limited to aggregate level and not using any window selection mechanism, because our major novelties lie in the usage of contractual information

and window selection to extract meaningful, discriminative features.

D. Evaluation Metrics

Precision, *recall*, and *F1* (F-measure) are well-known evaluation metrics for classification models [13]. Let TP be the set of true positive samples, TN be the set of true negative samples, FP be the set of false positive samples, and FN be the set of false negative samples. Precision, recall, and F1 are defined as follows:

$$Pr = \frac{|TP|}{|TP| + |FP|}, Re = \frac{|TP|}{|TP| + |FN|}, F1 = \frac{2 \cdot Pr \cdot Re}{Pr + Re}$$

E. Classification Algorithms

Multiple machine learning based classification algorithms are tested with the combination of proposed 116 features. These algorithms include *C4.5 Decision Tree* (C4.5) [14], *Maximum Entropy* (MaxEnt) [15], *Naïve Bayes* (NB) [16], *Repeated Incremental Pruning to Produce Error Reduction* (RIPPER) [17], and *Support Vector Machines* (SVM). We use Weka[1] implementation of these algorithms. Note that we also tried ensemble methods such as Boosting, Bagging, Voting, Probability Averaging, and Rotation Forest, but the performance overall does not improve much. Since our main novel contributions lie upon the feature engineering, we will put less focus on rigorous fine-tuning classification techniques and will stick with the default configurations of the base classifiers mentioned earlier.

The probability cutoff threshold is tuned during the training process to maximize F1, using 10 fold cross validation. Specifically, the *training* data is first divided into 10 equal subsets. For each fold i, the subset i (also called *hold-out data*) is used to construct the threshold-vs-F1 curve, while the other nine subsets are combined and used to train the classifier. The optimized thresholds selected by the 10 folds are then averaged, and used as the classifier's probability threshold.

Figure 6 plots the classification results of each classifier on the four regions. All the 116 proposed features are extracted and used to train the classifiers. It comes to a consensus that *Random Forest* (RF) outperforms other classifiers in all regions. *SVM* performs the second best in all regions. Interestingly, *NB* classifiers perform significantly poorly in most regions. Not surprisingly, *RF* has a resilient built-in feature selection mechanism that chooses an appropriate subset of features to describe each group of data which has a distinct set of characteristics (i.e. volatility, frustration, sudden rise/fall, etc.). Therefore, *RF* is used as the main classification algorithm for further analysis throughout the paper.

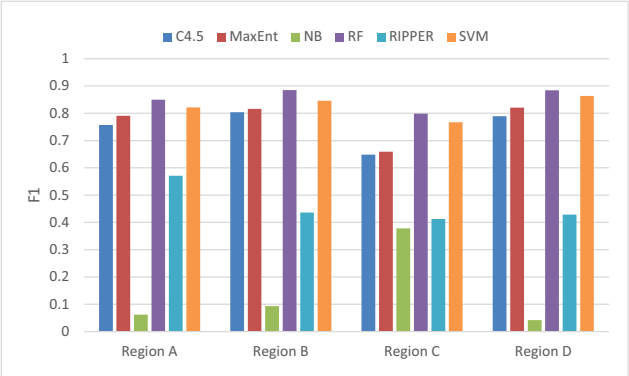

Figure 6. Comparison of the performance of the classification algorithms trained with our proposed combined features (in terms of F1) on each geographical region.

Table III
CLASSIFICATION RESULTS ON EACH INDIVIDUAL FEATURE TYPE AND COMBINED FEATURES AGAINST THE BASELINE FEATURES ON THE FOUR GEOGRAPHICAL REGIONS.

Region	Feature Type	Precision	Recall	F1	Region	Feature Type	Precision	Recall	F1
A	BASELINE	***0.97***	0.48	0.64	C	BASELINE	***0.95***	0.55	0.69
	GLOBAL-SW	0.56	0.57	0.57		GLOBAL-SW	0.60	0.55	0.57
	GLOBAL-SG	0.43	0.45	0.44		GLOBAL-SG	0.53	0.48	0.50
	CONTRACT-SW	0.88	0.75	0.81		CONTRACT-SW	0.81	0.72	0.76
	CONTRACT-SG	0.89	0.75	0.81		CONTRACT-SG	0.77	0.71	0.74
	COMBINE	0.92	***0.81***	***0.86***		COMBINE	0.83	***0.77***	***0.80***
B	BASELINE	0.92	0.47	0.62	D	BASELINE	0.92	0.52	0.66
	GLOBAL-SW	0.60	0.60	0.60		GLOBAL-SW	0.60	0.57	0.58
	GLOBAL-SG	0.46	0.46	0.46		GLOBAL-SG	0.44	0.46	0.45
	CONTRACT-SW	0.89	0.76	0.82		CONTRACT-SW	0.87	0.77	0.82
	CONTRACT-SG	0.89	0.78	0.83		CONTRACT-SG	0.86	0.79	0.82
	COMBINE	***0.93***	***0.86***	***0.89***		COMBINE	***0.93***	***0.85***	***0.89***

F. Classification Results

The proposed features are tested against the baseline features using the datasets from the four geographical regions. In each region, the combined features (*COMBINE*) along with other combination of the proposed features (i.e ***GLOBAL-SW*** (using only *GLOBAL* transaction data with *SW* window selection), ***GLOBAL-SG*** (using only *GLOBAL* transaction data with *SG* window selection), ***CONTRACT-SW*** (using only *CONTRACT* transaction data with *SW* window selection), and ***CONTRACT-SW*** (using only *CONTRACT* transaction data with *SG* window selection)) against the baseline features that compute the continuity statistics from the *GLOBAL* data using the entire data (i.e. no window selection mechanism).

The results are summarized in Table III. The best performance (in terms of F1) in all regions is achieved by the proposed *COMBINE* features. Interestingly, in some regions such as *A* and *C*, the best *precision* is achieved by the baseline features. This is because, since the baseline features use the entire data, it tends to favor services whose trends are steady in the long run. These steady services tend to

[1]http://www.cs.waikato.ac.nz/ ml/weka/

form trivial positive samples, which can be picked up by the baseline model. Since these steady services are merely part of the positive samples, we can see that the baseline models yield much lower recall compared to our proposed models.

It is worth noting that the feature extraction time and the learning time of the *COMBINE* models are larger than those of the *BASELINE* model. This is because for each candidate, the proposed model does not only extract features from the aggregate level time series, but go further to extract them from the underlying contracts that cover the *before* and *after* names. Hence, the feature extract time highly correlates with the number of signed contracts in each region. The classifiers also take more time to learn the *COMBINE* features due to the larger feature space (116 vs. 6 features). However, since the feature extraction and model training processes can be performed off line, the increase in performance (roughly by 31% on average) outweighs the cost of longer training time.

It is worth noting also that the *COMBINE* methods perform statistically significantly better than the baseline in most regions (except Region C). From careful observation, we notice that Region C has relatively small samples compared to other regions, and most positive samples are trivial. These trivial sample would likely be captured by the baseline method as well.

One may notice that the proposed *GLOBAL-SW* and *GLOBAL-SG* both perform relatively worse than the baseline. Specifically, they achieve similar recall, but worse precision. Similar to baseline, both *GLOBAL-SW* and *GLOBAL-SG* extract continuity statistics from the global cost and revenue data. The only difference is that they draw the statistics from the selected windows. Hence, both *GLOBAL-SW* and *GLOBAL-SG* methods are exposed to less data than the baseline (which also result in less feature extraction time). Hence, we conclude that the proposed window selection methods would be most effective when used on both global and contractual data.

VI. Conclusions

A machine learning based classification methodology was proposed to automatically discover service name replacements using features extracted from transaction time series data available in ledger databases. A set of heterogeneous features are drawn from different aspects such as transaction types, contractual information, and window selection mechanisms. A set of ground-truth synthesized data generated from a real-world ledger database, provided by IBM strategic outsourcing services, was used to validate the proposed models. The experimental results on a limited dataset show great promise that the proposed methodology could be applicable on larger, real-world datasets. Future work could strengthen the evaluation by applying the proposed methods on real-word data and having accounting experts verify the results.

References

[1] J. Blomberg, N. Boyette, A. Chandra, S. Oh, R. Zhou, R. Strong, W. Jones, O. Gehb, A. Vogt, and G. Satzger, "Forecasting service profitability," in *SCC '14*. IEEE, 2014, pp. 370–377.

[2] F. Wang, L. Liu, and C. Dou, "Stock market volatility prediction: a service-oriented multi-kernel learning approach," in *SCC '12*. IEEE, 2012, pp. 49–56.

[3] S. Oh, R. Strong, A. Chandra, and J. Blomberg, "Forecasting workloads in multi-step, multi-route business processes," in *IEEE Int. Conf. on Services Computing (SCC) 2014*. IEEE, 2014, pp. 355–361.

[4] P. Christen, *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer Science & Business Media, 2012.

[5] L. C. Gerhard, W. E. Harrison, B. M. Hanson *et al.*, *Geological Perspectives of Global Climate Change: AAPG Studies in Geology 47*. AAPG, 2001, no. 47.

[6] W. Mach, B. Pittl, and E. Schikuta, "A forecasting and decision model for successful service negotiation," in *SCC '14*. IEEE, 2014, pp. 733–740.

[7] G. Ganeshapillai, J. Guttag, and A. Lo, "Learning connections in financial time series," in *ICML '13*, 2013, pp. 109–117.

[8] R. S. Tsay, *Analysis of financial time series*. John Wiley & Sons, 2005, vol. 543.

[9] S. Tuarob, S. Bhatia, P. Mitra, and C. L. Giles, "Automatic detection of pseudocodes in scholarly documents using machine learning," in *ICDAR 2013*. IEEE, 2013.

[10] S. Tuarob, C. S. Tucker, M. Salathe, and N. Ram, "An ensemble heterogeneous classification methodology for discovering health-related knowledge in social media messages," *Journal of biomedical informatics*, pp. 255–268, 2014.

[11] S. Tuarob and C. S. Tucker, "Automated discovery of lead users and latent product features by mining large scale social media networks," *Journal of Mechanical Design*, 2015.

[12] P.-C. Chang, C.-H. Liu, J.-L. Lin, C.-Y. Fan, and C. S. Ng, "A neural network with a case based dynamic window for stock trading prediction," *Expert Systems with Applications*, 2009.

[13] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. NY, USA: Cambridge University Press, 2008.

[14] R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers, 1993.

[15] S. le Cessie and J. van Houwelingen, "Ridge estimators in logistic regression," *Applied Statistics*, vol. 41, no. 1, pp. 191–201, 1992.

[16] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *UAI '95*, 1995, pp. 338–345.

[17] W. W. Cohen, "Fast effective rule induction," in *ICML '95*. Morgan Kaufmann, 1995, pp. 115–123.