On-Line Evolving Clustering for Financial Statements' Anomalies Detection

Samir Omanovic
Department of Computing and Informatics
Faculty of Electrical Engineering
Sarajevo, Bosnia and Herzegovina
somanovic@etf.unsa.ba

Zikrija Avdagic
Department of Computing and Informatics
Faculty of Electrical Engineering
Sarajevo, Bosnia and Herzegovina
zavdagic@etf.unsa.ba

Samim Konjicija
Department of Computing and Informatics
Faculty of Electrical Engineering
Sarajevo, Bosnia and Herzegovina
skonjicija@etf.unsa.ba

Abstract—**This document proposes an approach for financial statements' anomalies detection by using on-line evolving clustering [1]. Official records of the financial activities of a business are called financial statements and they are recorded in journals and general ledger in a supervised process. Anomalies in financial statements are caused by human mistakes during forming of financial statements, or as a result of changes in the software that produced un-expected errors, or as possible financial fraud.**

Keywords – anomalies detection; evolving clustering; fraud detection;

I. INTRODUCTION

Modern accounting systems put hundreds and thousands of financial statements per minute in general ledger. Supervising this process is not an easy task and failure in identifying anomalies in financial statements can cause serious damages to business [2]. Anomalies in financial statements are caused by human mistakes during forming of financial statements, or as a result of changes in the software that produced un-expected errors, or as possible financial fraud. There are various examples of use of artificial intelligence methods for financial information classification for different purposes [2] [3] [4] [5] [6], including fraud detection, anomalies detection, etc.

This paper has focus, not only on fraudulent anomalies or anomalies caused by humans during financial statements creation, but also on anomalies that are result of software maintenance changes. Information systems are often changed during their maintenance lifecycle. Most of these changes are result of new business requirements and most of these changes are related to parts of the system other then accounting. Although most of the changes are on the parts of the system other then accounting most of that changes reflects accounting because events and results of business processes are recorded through financial statements. During business processes execution many documents are created, such as an invoice, receipt, etc. that are the source for identifying financial transactions and creating financial statements that best describes those transactions. The classical approach in controlling of these financial statements is based on balance control and the supervisor's experience in finding anomalies in financial statements, but nowadays this is not sufficient.

II. ON-LINE EVOLVING CLUSTERING

The on-line evolving clustering is an algorithm for dynamic clustering of an input stream of data (Kasabov and Song, 2002). It is one-pass algorithm and clusters are created dynamically, i.e. number of clusters is not predefined. Clustering is done based on distance between example point and the closest cluster center. If distance of some point to the closest cluster center is longer then a threshold value – D_{thr}, then new cluster is created.

Basic algorithm is:

1) Create the first cluster C_1 where first example for classification is cluster center Cc_1. Set cluster radius R_1 to *0*.

2) If there is no new example for classification then the clustering process ends. Else, distances D_{ij} between current example for classification x_i and all n already created cluster centers Cc_j is calculated:

$$D_{ij} = || x_i - Cc_j || \quad (j = 1, 2, \ldots, n). \tag{1}$$

3) If there is a cluster C_k (center Cc_k, radius R_k) such that distance D_{ik} is a minimal distance, ie. $D_{ik} = \min \{D_{ij}\}$ (j = 1, 2, …, n), and that distance is smaller then the radius R_k ($D_{ik} < R_k$), then x_i belongs to cluster C_k. This does not

978-1-4244-4221-8/09/$25.00 ©2009 IEEE

creates a new cluster nor updates existing one. The algorithm returns to step 2.

4) Else, find a cluster C_a (center Cc_a, radius R_a) such that value $S_{ia} = D_{ia} + R_a$ has a minimal value, ie. $S_{ia} = \min \{S_{ij}\}$ ($j = 1, 2, \ldots, n$).

5) If S_{ia} is greather then 2 x D_{thr}, the current example does not belong to any existing cluster and a new cluster is created as is described in step 1. The algorithm returns to step 2.

6) If S_{ia} is not greather then 2 x D_{thr}, the cluster C_a is updated by moving its center, Cc_a, and increasing its radius value, R_a. The new radius value R_a^{new} is set to $S_{ia}/2$ and the new center Cc_a^{new} is located on the line that connects x_i and the old cluster center Cc_a on the R_a^{new} distance from the x_i. The algorithm returns to step 2.

III. ANOMALIES DETECTION

Here we propose anomalies detection in two levels, using on-line evolving clustering method. First level classifies financial statements based on their account, partner/equipment/employee/… codes, amount debits, amount credits. Second level classifies financial documents based on their type and accounts they use for recording financial statements.

Whenever is detected financial statement/document that can not be placed in any existing approved or disapproved cluster new cluster is created and put on the list for approval. Each candidate cluster can be approved or disapproved and based on that future financial statements that mach those clusters are automatically approved or disapproved. The person authorized for approving financial statements can approve or disapprove clusters but also it can approve one specific financial statement that belongs to disapproved cluster allowing it as one specific exception which will not create new clusters nor change existing one.

A. First Level of the Anomalies Detection

The first level of the anomalies detection is based on four parameters. The first two parameters are account and code of partner, equipment, employee or other entity relevant to the account. For each combination of these two parameters, that is used in financial statements is created at least one cluster. The cluster properties and classification are based on the other two parameters: amount credits and amount debits.

B. Second Level of the Anomalies Detection

The second level of anomalies detection is based on document type, as the first parameter and list of accounts used in financial statements on that document. For each document type is created at least one cluster. All accounts that are on the document are used and count of occurrences for each account is calculated (Fig. 1).

Acc_1	Cnt_1
Acc_2	Cnt_2
…	…

Figure 1. List of accounts on the document

IV. DISTANCE CALCULATION

One of the most important issues for evolving clustering algorithm execution is calculation of distance between an example for classification and already created cluster centers. For the first level of anomalies detection, values of amount debits and amount credits are easy to use for that, but for the second level of anomalies detection we defined specific distance measuring method, because it is necessary to measure the distance between two lists of accounts.

For that purpose, we propose method of measuring the distance between two lists of accounts as presented on Fig. 2. This method is mostly based on experience from experiments.

x_i	
Acc_1	$Cnt_{1,A}$
Acc_2	$Cnt_{2,A}$
Acc_4	$Cnt_{4,A}$
Acc_5	$Cnt_{5,A}$
…	…

Cc_k	
Acc_1	$Cnt_{1,B}$
Acc_2	$Cnt_{2,B}$
Acc_3	$Cnt_{3,B}$
Acc^*_5	$Cnt_{5,B}$
…	…

Distance
$d_1=\lvert Cnt_{1,A} - Cnt_{1,B}\rvert$
$d_2=\lvert Cnt_{2,A} - Cnt_{2,B}\rvert$
$d_3=1000$
$d_4=1000$
$d_5=10\cdot\lvert Cnt_{5,A} - Cnt_{5,B}\rvert$
…
$D_{ik}=\sum d_j$ (j=1,2,...)

Figure 2. Distance calculation

For equal accounts, distance is calculated as absolute difference between number of occurrences of that account on example document x_i and cluster center document Cc_k. If some account is present on one document and is not present on the other then total distance is increased for 1000. When accounts on two documents are similar (marked with * on Fig. 2), i.e. when they belong to the same class or subclass from the chart of accounts (example: **1200** and **1200**10), then distance is calculated as the distance between two same accounts multiplied by 10.

V. THRESHOLD VALUE

For the execution of algorithm it is necessary to define algorithm parameters. One of the most important parameter is threshold value that defines when to create new cluster (D_{thr}). If we define small threshold value then too many clusters will be defined and they will all go for approval. If we define large threshold value lower number of anomalies can be detected.

Our opinion is that most convenient approach in defining threshold value for the first level of anomalies detection is to consider size of anomaly we want to determine. For example, if we want to determine all financial statements where amount debits or amount credits differs for more then 100 EUR (or other banknote) from the approved cluster centers then threshold value should be set to 100 (D_{thr}=100).

This means that none of the created clusters will have radius bigger then 100. This value of parameter will allow us to detect most of the anomalies like the one where one numeric character of a number is missing or is added, which is the one of the most often error done in manual creation of financial statements. In that case the number is approximately 10 times bigger or smaller then the expected number and threshold value will work, except for values less then 10.

For the second level of anomalies detection, threshold value can be determined based on the method of distance calculation that is described before (Fig. 2). If there is at least one account on the document that does not exists on the cluster center document, or contrary, then it should be treated as new cluster. In that case, the distance is minimally 1000 (d_3 or d_4 on Fig. 2). This means that for the second level of anomalies detection we can set the threshold value to 1000 (D_{thr}=1000). In all other situations, where the distance between the example document and the cluster center is less then 1000, we can threat documents as relatively similar.

VI. IMPLEMENTATION

Since the idea is to implement real-time controlling and the algorithm is for on-line clustering then we decided to implement algorithm as two database triggers [8] – one for the first level and the other for the second level of anomalies detection. Additionally, three auxiliary tables for recording cluster information are created. Structures of these tables are presented in tables I, II and III.

TABLE I. CLUSTERS – LEVEL 1 (TABLE CC_LEVEL1)

No	Column name	Type	Size
1.	ACCOUNT	VARCHAR2	10
2.	ENTITY_TYPE	VARCHAR2	2
3.	ENTITY_CODE	VARCHAR2	10
4.	DEBIT_CREDIT	VARCHAR2	1
5.	CLUSTER_CENTER	NUMBER	
6.	CLUSTER_RADIUS	NUMBER	

TABLE II. CLUSTERS – LEVEL 2 (TABLE CC_LEVEL2)

No	Column name	Type	Size
1.	DOCUMENT_TYPE	VARCHAR2	2
2.	ACCOUNTS_LIST_REF [a]	NUMBER	
3.	CLUSTER_CENTER	NUMBER	
4.	CLUSTER_RADIUS	NUMBER	

a. Reference to the list of accounts placed in the table ACCOUNT_LIST

TABLE III. ACCOUNTS LISTS (TABLE ACCOUNTS_LISTS)

No	Column name	Type	Size
1.	ACCOUNTS_LIST_REF	NUMBER	
2.	ACCOUNT	VARCHAR2	10
3.	OCCURRENCE_CNT	NUMBER	

VII. TEST

For the purpose of testing this approach we prepared the database of 1000 documents that we approved automatically and 100 documents that we modified in the way that we created some anomaly on them. In some financial statements on the documents we changed account number, or we changed the amount debits or amount credits by shifting it for one decimal place, or we put some nonexistent code of entity (partner, equipment, employee, etc.). All documents used in tests are from real system. After executing algorithm with automatic approval on stream of 1000 documents, clusters are created with their center and radius values. During insertions of documents with anomaly, algorithm detected them, depending on the anomaly, as anomaly of level 1 or as anomaly of level 2 and recorded information about the anomaly.

For example, one of the document types that we used in tests was invoice. Invoice can have a tax item. Let us assume that software has a configuration parameter that can include or exclude the tax on the invoice and that financial statements are created automatically based on configuration parameters, including the one related to the tax. Also, let us assume that the current wanted configuration is that tax is calculated. Typical error that can be done during the software maintenance is that the software or the configuration change reflects the invoice creation or the automatic creation of financial statements in the way that tax is not calculated or created, i.e. the tax item is zero or the tax item does not exists. Without controlling is possible that this error is not detected immediately, but after some time and is possible that serious damage is created by this way (tax is not paid in the correct amount, for example). This is the one of situations that we tested and suggested approach is successful in detecting these situations. When item is created with zero value of the tax then it is caught by the first level of anomalies detection, and when the tax item is not created then document does not contain the account related to the tax and is detected by the second level of the anomalies detection. Described case that we tested is just one of many similar and all are related with the automatic creation of financial items that is based on accounting schemas and system configuration parameters.

Second important segment that we tested is related to the financial documents that are created rarely, i.e. few times in a year. Usually these documents contain very specific financial statements related to one non frequent situation (compensation agreement with large amounts, for example). Test shows that when we approve these documents as a rule, detecting of problems on the first level becomes weaker. But, when these documents are approved as the exception, then they do not have the influence on the clusters created, thus approach works.

Generally, tests show that approach is good, but it is highly dependant on the person that monitors the process, i.e. if the person that decides what and how to approve understands the circumstances of the decision then the system will work as expected.

VIII. MONITORING PROCESS

Based on the proposed approach for financial statements' anomalies detection by using on-line evolving clustering, we propose, on Fig. 3, monitoring process schema for a practical implementation.

Steps of the monitoring process are numbered as 1 to 6 on Fig. 3. Each document with its financial statements is caught by triggers when inserting or updating [8][9] (step 1). Triggers perform two levels of anomalies detection (step 2). If non anomaly is detected, i.e. there is approved cluster where belongs controlled document and its financial statements then it is placed in the destination tables as approved (step 3). If anomaly is detected, i.e. cluster where belongs controlled document is not approved or there is no cluster for that document then it is placed in the destination tables as "disapproved" or "waiting for the approval" (step 4). All

documents that are marked as “disapproved” or “waiting for the approval” are viewed through the Monitoring Application. The Monitoring Application enables functionalities as: approving or disapproving new clusters, approving one specific document as an exception, etc. If one document or cluster where it belongs is approved through Monitoring Application then its status is changed and, if necessary, information about new cluster is recorded in the auxiliary tables (step 6).

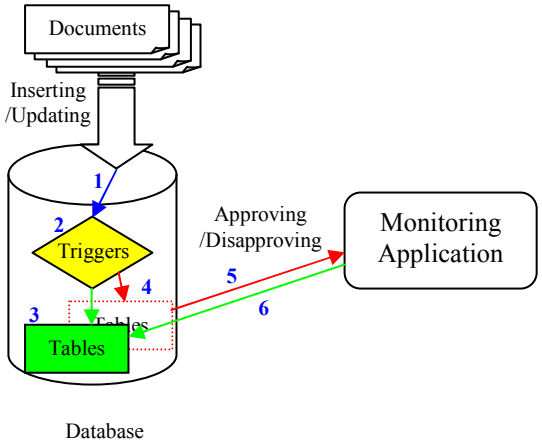

Figure 3. Monitoring process

The Monitoring Application interface can be very simple, containing only one data grid to show the records that are marked as “disapproved” or “waiting for the approval”, one combo box with the list of available statuses and one button that will change the status of the current record into the selected one in the combo box.

IX. Conclusions

Implementation of the proposed approach is not too complicate. Algorithm has simple steps and calculations are relatively simple. Implementation requires few tables and triggers written in SQL language [8][9] if we exclude the Monitoring Application. Algorithm implemented in tables’ triggers has a fast execution and does not cause notable time delay in inserting and updating operations. This conclusion is relevant for small systems since we performed tests on small system. On larger systems speed of execution can be an important issue and should be measured more precisely. To have efficient work of the system there are few practical advices that are result of practical experiments and are placed here to decrease the dependency of the approach on the user's capabilities:

- All clusters of documents that are usual and contain usual financial statements should be approved as soon as possible. This will prevent unnecessary overloading of the approval process and allow to system normal functioning.
- Documents that are obvious errors or contain obvious errors should be marked as disapproved and their clusters should be marked as disapproved. This action is very important if we decide that system automatically reject all documents that are disapproved, i.e. if we instead of monitoring shift to controlling process.
- Documents that are erroneous in some situations, but are not erroneous in other situations, or documents whose financial statements are erroneous in some situations, but are not erroneous in other situations should be approved or disapproved as exceptions of rule. This is important because if we approve or disapprove clusters of these documents or their financial statements then we can cause wrong classification which can degrade system quality.

To simplify principles of use we can say that usual thing should be generally approved, obviously erroneous things should be generally disapproved and thing that are sometimes correct and sometimes incorrect should be approved or disapproved one by one, as exceptions.

X. Further Work

This approach can be seen as the first step in building robust financial statements’ anomalies detection system. On-line evolving clustering method is a good algorithm for this implementation, but we plan to investigate some other algorithms and combination of the of-line and on-line learning. Especially is important to investigate possibilities for more precise classification of cases that are sometimes correct and sometimes incorrect. Using fuzzy logic for that purpose can be a direction of our research. As we noted in tests, weakness of the approach is high dependency on the user of the system and one of the directions for further work can be to create a subsystem who will suggest to the user what and how to approve or disapprove and in that way to decrease the dependency on the user's capabilities.

References

[1] N. Kasabov, Evolving connectionist systems, Springer-Verlag London Berlin Heidelberg, 2003, pp. 40–42.

[2] B. Dubinsky, C. Warner, Uncovering accounts payable fraud using "fuzzy matching logic": Part 1, Business Credit, March 2008, Vol. 110 Issue 3, pp. 6-9.

[3] B. Dubinsky, C. Warner, Uncovering accounts payable fraud using "fuzzy matching logic": Part 2, Business Credit, April 2008, Vol. 110 Issue 4, pp. 64-66.

[4] C. Serrano-Cinca, Feedforward neural networks in the classification of financial information, European Journal of Finance, September 1997, Vol. 3 Issue 3, pp. 183-202.

[5] B.P. Green, J.H. Choi, Assessing the risk of management fraud through neural network technology, Auditing, Spring 1997, Vol. 16 Issue 1, pp. 14-28.

[6] L. Kryzanowski, M. Galler, Analysis of small-business financial statements using neural nets, “Journal of Accounting, Auditing & Finance”, Winter 95, Vol. 10, Issue 1, pp. 147-170.

[7] C.E. Hogan, Z. Rezaee, R.A. Riley, K.U. Velury, Financial statement fraud: Insights from the academic literature, Auditing, November 2008, Vol. 27, Issue 2, pp. 231-252.

[8] Oracle, Develop PL/SQL Program Units – student guide, Oracle Corporation, June 2000, Production 1.4, 41024GC14.

[9] B. Bryla, Oracle database foundations, SYBEX Inc. Alameda, 2004.