Ibn Al Haitham Journal for Pure and Applied Science

Journal homepage: http://jih.uobaghdad.edu.iq/index.php/j/index

Design the Modified Multi Practical Swarm Optimization to Enhance Fraud Detection

Zainab Khamees Muter **Abeer Tariq Molood**

Department of Computer Science, University of Technology/Baghdad

zooz_kh1990@yahoo.com 110032@uotechnology.edu.iq

Article history: Received 13 November 2019, Accepted 5 January2020, Published in April 2020.

Doi: 10.30526/33.2.2425

Abstract

Financial fraud remains an ever-increasing problem in the financial industry with numerous consequences. The detection of fraudulent online transactions via credit cards has always been done using data mining (DM) techniques. However, fraud detection on credit card transactions (CCTs), which on its own, is a DM problem, has become a serious challenge because of two major reasons, (i) the frequent changes in the pattern of normal and fraudulent online activities, and (ii) the skewed nature of credit card fraud datasets. The detection of fraudulent CCTs mainly depends on the data sampling approach. This paper proposes a combined SVM- MPSO-MMPSO technique for credit card fraud detection. The dataset of CCTs which consists of 284,807 transactions performed by European cardholders in 2013 was used in this study. The proposed technique was applied to both the raw dataset and the pre-processed dataset. The performance of these techniques is evaluated based on accuracy, and the fastest time it takes to detect fraud. This paper, proposed a technique that uses SVM, MPSO and MMPSO to form an ensemble for the detection of credit card fraud.

Keywords: Fraud detection, SVM, MMPSO, Financial fraud**.**

1. Introduction

Fraud is defined as unlawful activities or behaviors encountered in industrial & commercial organizations like insurance companies, banks, and mobile phone companies. In the credit card companies, fraud is a serious issue and the rate of its increase demands effective methods for its early prevention and detection [1]. Financial fraud detection using the conventional internal audit technique is a complicated task which in some cases, is even an impossible venture [2]. This is because, first, most auditors may not be equipped with the current knowledge of accounting fraud characteristics, and secondly, accounting data manipulation is not done always, hence, most auditors may not have the required expertise to prevent or detect such manipulations. Another issue is related to the management officers of banks, such as the Chief Financial Officer (CFO), accountants, and financial managers. These people always try to intentionally confuse or deceive the auditors [3]. Since they are aware of the limitations of an audit, they may decide that the standard or traditional auditing techniques are not enough for fraud detection. With these

Financial auditing limitations, it becomes necessary to device automatic data analysis tools and procedures that will ensure perfect and efficient detection of financial statements' falsification.
The security of information and communication systems are ensured by using effective fraud detection systems (FDS) which are similar to other measures, such as firewalls, antivirus software, and access control schemes. FDS is conventionally categorized into signature-based, anomaly-based, and hybrid detection systems [4]. For the signature-based detection (SBD) system, the pattern of the network traffic or application data is taken to be malicious, but in the anomaly-based detection (ABD) systems, network activities are matched against a pre-defined normal pattern. Regarding hybrid fraud detection systems, they are a combination of SBD and ABD systems. There are advantages and challenges of each technique; some of the advantages of ABD systems over the other systems include: (i) ability to detect insider attacks (for instance, when a stolen account is being used to perform transactions beyond the normal profile of the legal owner of the account, the system will generate alarm), (ii) ability to detect invasions based on customized profiles (making it difficult to launch an attack without triggering the alarm), and (iii) ability to detect previously unknown attacks. The FDS monitors the network for anomalous activities instead of the attacks. The role of credit cards in today's economy is significant as it has become an integral part of peoples' daily lives. Despite the huge benefits of using credit cards when used responsibly, fraudulent activities on such cards can cause significant financial damages. Several methods of preventing the proliferation of card fraud have been proposed and among all, fraud detection remains the most explored aspect [6]. Credit card frauds are prevented using several authorization techniques, such as credit card number, signatures, cardholder's address, identification number, etc. Meanwhile, these methods cannot effectively prevent credit card fraud; thus, fraud detection techniques equipped with the capability to analyze data in a bid to detect and prevent credit card-related fraud must be developed and used [7]. Pattern recognition remains the commonest approach to credit card fraud detection; it involves an automatic analysis of the users' spending pattern as the spending pattern of a user normally contains information related to transaction amount, item category, day of the week, customer address, and elapsed time since last purchase. Anomaly-based detection systems can depend on classification as a technique to learn a set of training data in order to be capable of efficiently classifying a test dataset into normal or anomalous categories. This study proposes the utilization of SVM learning for CCTs detection; furthermore, the optimization of the SVM parameters using a modified multi-practical swarm optimization algorithm (MMPSO) is also projected in this study. This MMPSO approach provides chaos optimization, as well as a processing mechanism and a premature judgment.

2. Related work

Johannes et al (2016) used Long Short-Term Memory (LSTM) networks to integrate transaction sequences in order to describe the issue of fraud detection as a sequence classification task. They also incorporated state-of-the-art aggregation techniques for features and reported the outcomes using traditional metrics for retrieval. A comparison with a baseline random forest (RF) classification showed that the LSTM increased the precision of detection on offline transactions when the cardholder is physically present at a merchant. Both sequential and non-sequential learning methods benefit greatly from aggregation strategies for manual features. Nuno et al. (2017) explored the combination of manual and automatic classifications to provide insights into the entire process of growth and compared distinct techniques of machine learning. The study can assist scientists and professionals in designing and executing fraud detection or comparable issues based on information mining technologies. The proposed solution enhanced the manual revision process not only with an automatic scheme but also with insights to the fraud analysts, resulting in a superior overall result. Han & Cho (2005) proposed an evolutionary neural network-based intrusion detection system (IDS) for the determination of the weights and structure of call sequences, while a hybrid flexible neural tree-based IDS was developed by Chen, Abraham, & Yang (2007); this system combined three algorithms which are evolutionary algorithm, flexible neural tree, and PSO. Among the ANN-based IDS, hybrid ANN has attracted the most attention; however, the performance of a hybrid IDS is dependent on the way that the hybrid ANN is constructed.

3. Methods & Algorithms

- **Support Vector Machine (SVM)**

SVM is one of the commonest algorithms for classification tasks due to its strong theoretical basis [8]. The ability to process highly dimensional dataset, as well as its generalization performance. An overview of SVM learning is provided in this section, beginning with linear SVMs, then, to their application in non-linear cases. As a statistical learning approach, SVM has been applied in several fields. Its first introduction was made by Cortes & Vapnik [9]. And then, has been successfully applied in several classification problems. SVMs are closely related to neural networks (NN); since SVMs depend on kernel functions, they are normally considered an alternative approach to the development of NN classifiers. SVMs are supervised machine learning frameworks that have found application in one-class setting anomaly detection [9]. In such techniques, one-class learning techniques are used to train SVM in the region that contains the training datasets [8]. SVM classification frameworks are based on the concept of mapping a hyperplane as the decision plane which maximizes the distance between the positive and negative modes. The robustness of SVMs is attributed to their kernel representation and margin optimization. SVM can learn complex regions using RBF kernel. SVMs are mainly attractive owing to the ease of their training phase; they are not prone to local optimal as in NN. SVMs perform extremely well in high dimensional datasets and the balance between error and classifier complexity can be explicitly regulated. The major problem of SVM is the need to find the best kernel function to apply.

In a high dimensional feature space, the dot product of projections of 2 data points is represented by the kernel function. The classification function in SVMs is a hyper-plane that demarcates the different data classes. Its basic concept requires the demonstration of the smallest hypersphere in the kernel space with all the training features, followed by the determination of the side of the hypersphere where the test case lies. Test cases lying on the outer side of the hypersphere are considered suspicious. A special type of linear model called the maximum margin hyperplane (MMH) is found by the algorithm; the MMH correctly classifies all the training cases by inserting a hyperplane that separates them into correct classes. The MMH provides the maximum separation between the classes. The MMH is expressed as follows:

$$\boldsymbol{magin = arg_{x \in D}^{d(x)} = argmin_{x \in D} \frac{|x \times w + b}{\sqrt{\sum_{i=1}^{d} w_i^2}}} \qquad (1)$$

Where x represents the vector point, w represents the weight (also a vector). For a correct data separation [a] must always be > 0. From all the possible hyperplanes, the one with the largest hyperplane distance is selected by the SVM. With good training data, every test vector should be located within a radius r from the training vector. However, if the farthest possible hyperplane from the data is selected [10]. This anticipated hyperplane with the maximum margin will also divide the lines between the nearest points on the convex hull of the two datasets. Hence, [a], [b] & [c] will be derived. Being that x is on the hyperplane, the distance of the hyperplane with the closest point to the origin can be determined by maximizing the x. The closest instances to the MMH are referred to as the support vectors (SV). Each class has at least one SV but often, there are more. For each test case in the detection of credit card fraud, the aim is to determine the location of the test instance within the learned region. Test instances that fall within the learned region are considered normal while those outside the learned region are considered anomalous. This model achieved higher detection accuracy compared to other algorithms. It is time-efficient and has generalization ability. When calculating the SVM, it is obvious that the aim is to ensure the correct classification of all the data. For mathematical calculations:

$A\ if\ Y_i = +1;$

$B\ if\ Y_i = -1; wx_i + b \le 1$

$For\ all\ i, Y_i(w_i + b) \ge 1$

The scenario is similar to the other side points. The total distance from the separating hyper plane to the closest points is derived by solving and subtracting the two distances, Maximum Margin = M = 2 / ||w||.

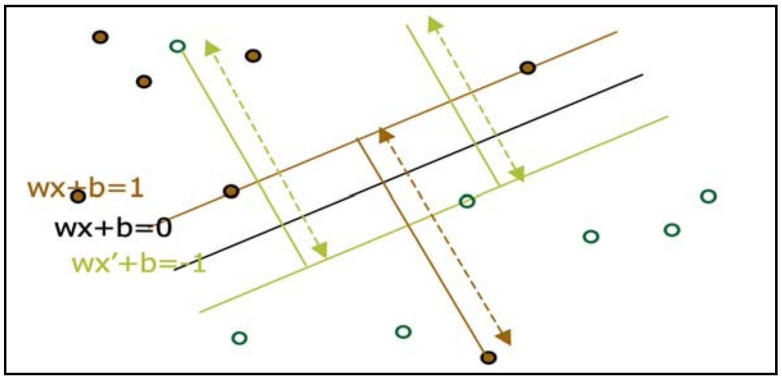

Figure 1. Representation of hyper planes.

- **Multi-Swarm PSO (MPSO)**

Multi-swarm concepts are mainly based on the inter-relationship between natural groups during a search for the solution to a common problem. Several multi-swarm-inspired techniques have been presented based on inspiration from natural concepts. This work suggests a novel multi-swarm interaction scheme based on the inspiration from human social behavior (inspired by the way a group of people (clan) interact with their leaders). In the suggested approach, there are several swarms known as clans and each clan is made up of numerous solutions that are represented by the members of the clan. Each clan's greatest member is the leader of the clan and has power over its clan members as to when to move and where to move [11,12].

In each generation, the clan leaders occasionally assemble in a room to elect an overall best leader (OBL) whose new-found position will be used to adjust that of the other leaders using Eqs. 2, 3, & 4.

$$y_i(t+1) = \begin{cases} yi(t) & if \ f(x_i(t+1)) \geq f(y_i(t)) \\ xi(t+1) & if \ f\left(x_i(t+1)\right) < f(y_i(t)) \end{cases} \tag{2}$$

$$V_{ij}(t+1) = V_{ij}(t) + C_1 r_{1j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + C_2 r_{2j}(t)\left[\hat{y}_{ij}(t) - x_{ij}(t)\right] \tag{3}$$

Where ˆyij is the best position, found by the neighborhood of particle i in dimension j. The local best particle position, yˆi, i.e. the best position found in the neighborhood Ni, is defined as:

$$\hat{y}_i(t+1) \in \{N_i \,|\, f\left(\hat{y}_i(t+1)\right) = \min\{f(x)\}, \ \ \forall x \in \ N_i\} \tag{4}$$

This knowledge-sharing behavior ensures a balanced exploration stage with the exploitation stage of the PSO. The model of the proposed Meeting Room Approach (MRA) allows that each clan performs only one PSO search (inclusive of velocity and position updating) during the generation of a new local population. Upon the emergence of a new generation, a clan leader is chosen for each clan and sent to the common meeting room where the overall best leader (global best) will be chosen from the local clan leaders. The position of the newly-elected overall best leader is shared with the rest of the normal leaders as follows:

$$w^{Ln} = \left(\frac{w^{Lg} - w^{Ln}}{Itr}\right) x \ rand() \tag{5}$$

$$v_i^{Ln}(t+1) = w^{Ln} x v_i^{Ln}(t) + rc(P_g^L - P_n^L(t)) \tag{6}$$

$$x_i^{Ln}(t+1) = x_i^{Ln}(t) + v_i^{Ln}(t) \tag{7}$$

Where Ln is the normal clan leaders and Lg is the overall best leader. x_i^L is the normal leaders' current position, while $v_i^{\ Ln}$ is the normal leaders' velocity. w^{Lg} is the inertia weight of the overall best leader and w^{Ln} is the inertia weight of the normal leaders. At the end of each generation, there is a change in the position of the swarm members; hence, a new leader will be elected for each swarm or should be updated in the meeting room. The exploration search of the algorithm is controlled by the new inertia equation in the meeting room. The evaluation of the performance of MPSO showed that it can solve various numerical problems. However, its dependence on uniform distribution equations for initialization can make it initiate from inappropriate positions; in this situation, the particles can search wrong search spaces, making the algorithm susceptible to local optima. In this study, the significant contribution is the initialization of the MPSO particles in each swarm using a popular chaotic map (logistic map); this is aimed at enhancing the particles' starting positions and improving the convergence performance of the framework [13]. The **Figure 2.** shows the MPSO-SVM algorithm:

Output: Best Position [x, y];

Input: Learning Dataset;

```
Set Number of Swarms (s);
Set Number of Particles in Swarms (p);
Set Number of Generations (g);
Initialize Particles with Random Positions (pp);
Initialize swarm Best Fitness (sbf);
Initialize Swarm Best Position (sbp);
Initialize Leader Best Fitness (lbf);
Initialize Leader Best Position (lbp);
Initialize worst-swarms (ws);
while gCount i Generation do
    for swarm in Swarms do
        for particle in swarm.particles do
            particle.fitness (pf) = SVM(particle.position);
            if pf < sbf then
                sbf = bf;
                sbp = particle[pp];
            end
        end
        if sbf < lbf then
            lbf = sbf;
            lbp = sbp;
        end
        if ws.fitness < swarms.fitness then
            ws = swarm_current ;
        end
    end
    Generate new swarm (ns);
    while true do;
        if ws in meeting-room  ns.fitness i  ws.fitness then
            if meeting-room is full then
                remove ws from swarms list;
            end
        else
            Generate new swarm (ns)
        end
    end
    gCount = gCount + 1
end
```

Figure 2. MPSO-SVM algorithm.

- **Modified MPSO (MMPSO)**

Multi-swarm optimization was based on the decomposition method which uses the conventional mathematic technique to decompose the MPSO into several single objective problems (also known as sub-problems to be solved). Each sub-problem represented a multi-objective weight vector, and the optimal solution to each sub-problem was a Pareto solution of the MPSO. Each sub-problem is optimized by the neighborhood sub-problems with closest weight vector owing to their closeness to the Pareto solution on the Pareto front. This was achieved by estimating the Euclidean distance of the weight vectors, thereby determining the neighborhood problem of the sub-problem. This is the major concept of the multi-objective decomposition evolution strategy.

The problem of the MPSO algorithm is the performance of the meeting room. The existence of many swarms in the meeting room degrades the learning performance. The proposed solution suggests the use of a new mechanism for the meeting room in order to improve performance.
Adding and removing swarms from the learning space leads to the execution of many operations as each new swarm needs to start learning from zero. Furthermore, adding new swarms to the learning space will not always give a better solution because the new swarm may not offer better fitness. Additionally, the new swarm may be removed after one iteration due to the fitness value if it has the worst solution. To improve the effectiveness of the PSO algorithm, a number of modifications towards the standard version were introduced. These improvements were on exploring the search space and the way of information collection by the swarm. The essential modification relies on a rebuilt of the velocity vector updating equation to which an additional element was introduced. The modified MPSO focused on the mechanism of the meeting room. In the meeting room, the swarms were being tested in every iteration in order to find the worst swarm with the worst fitness. The detected swarm was not removed from the room until a new swarm was generated and added to the learning space. The new swam was compared with the worst swarm in order to find if the new swarm could present better fitness for the learning space. If the new swarm is better than the worst one, a new swarm was added to the space and the worst one was removed, else, the new swarm will be discarded, and another new swarm will be generated. This mechanism was repeated until a better swarm with better fitness was found. This has allow finding the best swarm in less time with higher accuracy because there was no useless swarm added or removed until a better one is found.

The MPSO-SVM algorithm, as shown in **Figure 3.** represents the main steps and modifications that were made to use MMPSO with SVM for credit card fraud detection. To summarize the concepts and process of MMPSO-SVM solution, the steps of learning should be described. The proposed solution involved the following steps towards finding the best position for the learning of fraud detection:

1. First, the dataset was loaded to the SVM model after reading all the dataset records and selecting 30% of those records for learning purpose.
2. MMPSO algorithm was initialized with the specific number of swarms and particles in each swarm.
3. Each particle in each swarm was initialized with initial position and speed.
4. Now, each swarm run its particles by performing the SVM model as fitness function for the particles.
5. The error value was used as the fitness value for the particle; therefore, the particle with the least error value was considered as the best fitness particle.
6. The swarm leader toked the position and the fitness value of the best particle in the clan.
7. The global leader of MPSO toked the value of the position and fitness of the best swarm.
8. Each swarm was examined to find the worst swarm in the clan.
9. The worst swarm was removed if the MMPSO finds a better new swarm.

10. The fitness value, meeting room size, and the fitness of the new swarm were controlled.
11. The meeting room manages the swarm addition/removal mechanism. **Figure 3.** shows flowchart of the MMPSO.

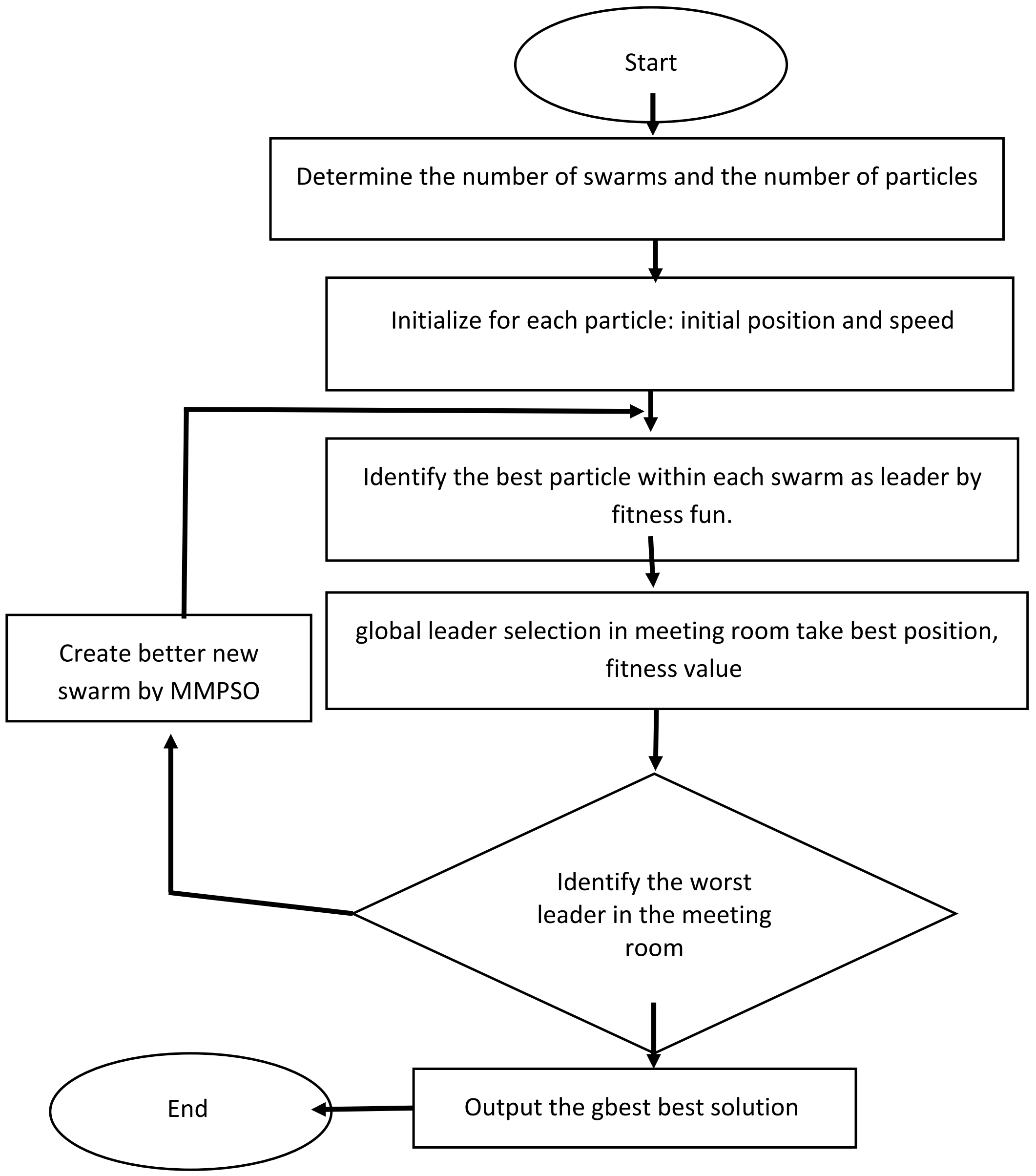
Start
Determine the number of swarms and the number of particles
Initialize for each particle: initial position and speed
Identify the best particle within each swarm as leader by fitness fun.
global leader selection in meeting room take best position, fitness value
Create better new swarm by MMPSO
Identify the worst leader in the meeting room
Output the gbest best solution
End

Figure 3. Flow chart of MMPSO.

4. Results and Performance Measures:

The proposed solution was implemented and developed using the Python programming language with a real dataset. Using such tools to implement the proposed solution was to ensure getting the best performance and results with real-world data. Datasets are an integral part of the field of machine learning. Gathering data is one of the hardest tasks, especially when it is related to the financial domain such as credit card fraud. In this study, the dataset comprised of credit card transactions performed by European cardholders within two days in September 2013. The datasets contain 284,807 records and 28 features for each record. This section presents the results of the benchmarking functions usually deployed in evolutionary studies [14]. There are variations in each test function with respect to the number of dimensions (fixed and dynamic) and modality (unimodal and multimodal). The evaluation of the MPSO for performance in this study was based on comparison with the standard PSO [15]. The proposed solution has been implemented using specific parameters in order to balance the number of swarms, particles, and the amount of processing that is needed to perform the solution in the environment. The MPSO-SVM parameters are set to number of swarms = 5, number of particles in each swarm = 5, number of generations for learning = 10, amount of learning data = 30% of the overall data, amount of accuracy testing data = 100% of the dataset records, inertia factor = 0.729, cognitive weight = 1.49445, social weight = 1.49445, global weight = 0.3645.

4.1. Accuracy

The accuracy value represents the number of frauds successfully detected by the detection system over the number of frauds in the dataset; therefore, the higher the accuracy value, the better the detection system. The accuracy of learning was calculated for SVM, PSO-SVM, and MPSO-SVM algorithms. The results show that the MPSO-SVM algorithm obtained higher learning accuracy than SVM and PSO-SVM as shown in **Figure 4.** And **Table 1.** PSO-SVM achieved less accuracy because the algorithm requires a high number of iterations to find the best result and improve the accuracy of SVM.

4.2. Mean Square Error (MSE)

The MSE is another comparison parameter used in many literature solutions to compare between optimization algorithms. The MSE value was calculated for SVM, PSO-SVM, and MPSO-SVM as shown in **Figure 3.** And **Table 1.** From the table, the best value of MSE was achieved with MPSO-SVM because of its high accuracy, followed by SVM with about 0.5 and PSO-SVM with about 0.55.

4.3. Learning Speed

It can be noticed that the PSO-SVM algorithm gave better learning speed than MMPSO-SVM because there was only one swarm in the learning space; however, the accuracy was worse than that of SVM without any improvement as mentioned before. Compared with the MPSO-SVM algorithm, the proposed solution offered better performance, being about 4 sec faster than MPSO-SVM. The amount of time needed to perform MMPSO-SVM algorithm was acceptable compared with the accuracy value since accuracy is a very critical part of credit card transactions. So, using the MPSO-SVM algorithm for fraud detection can efficiently increase the learning accuracy with better performance.

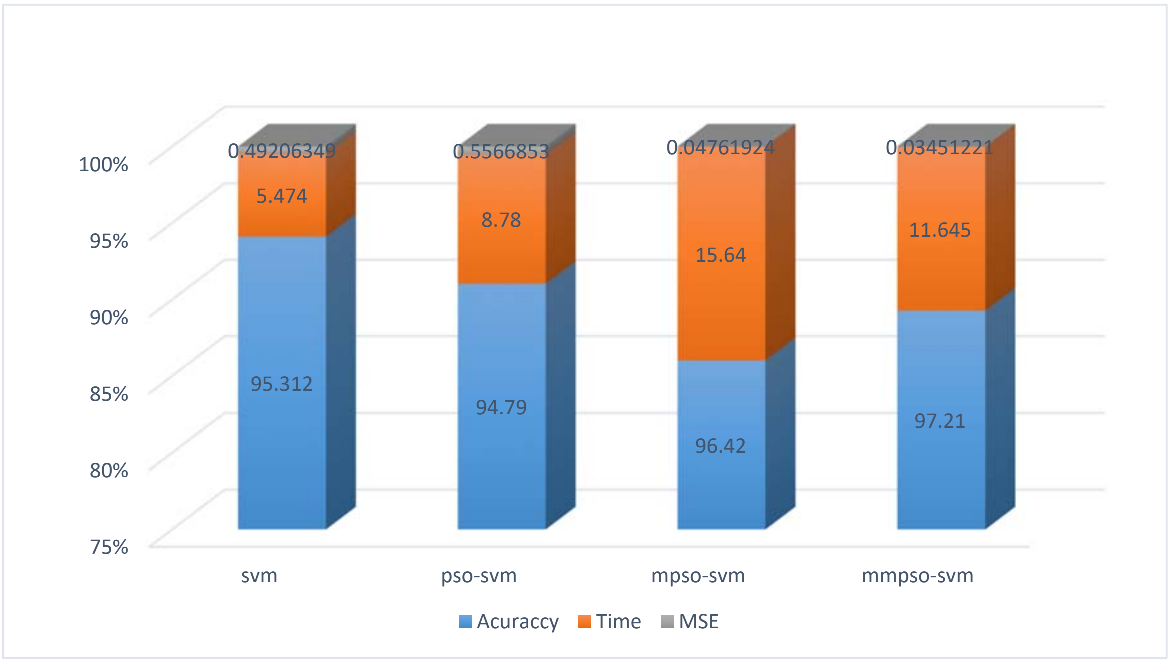

Figure 4. Comparison of the performance of the proposed approach with other techniques.

Table 1. Performance evaluation of the proposed approach.

Algorithm	**Accuracy (%)**	**Time(sec)**	**MSE**
SVM	95.31	5.47	0.4920635
PSO-SVM	94.79	8.78	0.5566853
MPSO-SVM	96.42	15.65	0.0476192
MMPSO-SVM	97.21	11.65	0.0345122

5. Conclusion

In this paper, the modified MPSO-SVM (MMPSO-SVM) algorithm was used to predict and classify credit card transactions into fraudulent or normal categories. The data set contains the record of credit card transactions performed by European cardholders within two days in September 2013. There are several conclusions drawn from the performance of the modified hybrid algorithm. The present research investigated a new method of optimizing credit card fraud detection based on multi-swarm particle optimization and SVM algorithm to ensure a better detection system. The MPSO's performance was validated in terms of accuracy and learning speed and compared with the traditional SVM and PSO-SVM detection systems. The results show that the proposed solution achieved better accuracy with acceptable performance. It outperformed SVM and PSO-SVM in terms of accuracy. The optimization process was executed on the performance of the detection system. Using multi-swarm with many particles in the learning space can decrease performance because of the number of processes and learning tasks to be performed on the dataset. In all, the use of this hybrid technique to enhance fraud detection can offer better performance accuracy and enhanced fraud transaction detection.

References

1. Accounting Fraud Definition and Examples retrieved http://www.accountingelite.com/accountingtips/accounting-fraud-definition-and-examples-freeaccounting-fraud-article.
2. Ngai, E.W.T.; Hu, Y.; Wong, Y. H.; Chen, Y.; Sun, X. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature, Decision Support System. **2010**, doi:10.1016/j.dss.2010.08.006.
3. Fanning, K.; Cogger, K.; Srivastava, R. Detection of management fraud: a neural network approach. *International Journal of Intelligent Systems in Accounting, Finance & Management*.**1995**, *4*, *2*, 113– 26.
4. Fanning, K.; Cogger, K. Neural network detection of management fraud using published financial data. *International Journal of Intelligent Systems in Accounting, Finance & Management*. **1998**, *7*, *1*, 21- 24.
5. Patcha, A.; Park, J.M. An overview of anomaly detection techniques: Existing solutions and latest technological trends; *Computer Networks*. **2007**.
6. Bolton, R.J.; Hand, D.J. Statistical fraud detection: A review. *Statistical science*. **2002**.
7. Awoyemi, J.O.; Adetunmbi, A.O.; Oluwadare, S.A. Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)*, *IEEE*. **2017**, 1-9.
8. Chapelle, O.; Vapnik, V.; Bousquet, O.; Mukherjee, S. Choosing multiple parameters for support vector machines. *Machine learning*. **2002**, *46*, *1-3*, 131-159
9. Shi, Y. Particle swarm optimization: developments, applications, and resources. In *Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546) IEEE*. **2002**, *1*, 81-86.
10. Singh, A.; Jain, A. Adaptive Credit Card Fraud Detection Techniques Based on Feature Selection Method. In *Advances in Computer Communication and Computational Sciences*. Springer, Singapore. **2019**, 167-178.
11. Singleton, T.W.; Singleton, A.J.; Bologna, G.J.; Lindquist, R.J. *Fraud Auditing and Forensic Accounting*. John Wiley & Sons. **2006.**
12. Panigrahi, S.; Kundu, A.; Sural, S., Majumdar, A.K. Credit card fraud detection: a fusion approach using Dempster–Shafer theory and Bayesian learning. *Information Fusion*. **2009.**
13. Salih, S.; Alsewari, Q.A.; Al-Khateeb, A.B.; Zolkipli, M.F. Novel Multi-swarm Approach for Balancing Exploration and Exploitation in Particle Swarm Optimization, *in Recent Trends in Data Science and Soft Computing*. **2019**.
14. Jamil, M.; X.S. Yang, H.J.D. Zepernick. Test Functions for Global Optimization: A Comprehensive Survey, in Swarm Intell. *Bio-Inspired Comput*. **2013**, https://doi.org/10.1016/B978-0-12-405163-8.00008-9.
15. Shi, Y.; Eberhart, R.C. Empirical study of particle swarm optimization, *Evol. Comput*. **1999**, https://doi.org/10.1109/CEC.1999.785511.