Fraud Detection using Machine Learning and Deep Learning

Conference Paper · December 2019

DOI: 10.1109/ICCIKE47802.2019.9004231

Pradheepan Raghavan
Eindhoven University of Technology
1 PUBLICATION **158** CITATIONS
SEE PROFILE

Neamat El Gayar
Cairo University
57 PUBLICATIONS **1,653** CITATIONS
SEE PROFILE

2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)
December 11–12, 2019, Amity University Dubai, UAE

Fraud Detection using Machine Learning and Deep Learning

Pradheepan Raghavan
School of Mathematical and Computer Sciences
Heriot Watt University
Dubai, UAE
Pradheepan@hotmail.com

Neamat El Gayar
School of Mathematical and Computer Sciences
Heriot Watt University
Dubai, UAE
Elgayar.neamat@gmail.com

***Abstract*— Frauds are known to be dynamic and have no patterns, hence they are not easy to identify. Fraudsters use recent technological advancements to their advantage. They somehow bypass security checks, leading to the loss of millions of dollars. Analyzing and detecting unusual activities using data mining techniques is one way of tracing fraudulent transactions. transactions. This paper aims to benchmark multiple machine learning methods such as k-nearest neighbor (KNN), random forest and support vector machines (SVM), while the deep learning methods such as autoencoders, convolutional neural networks (CNN), restricted boltzmann machine (RBM) and deep belief networks (DBN). The datasets which will be used are the European (EU) Australian and German dataset. The Area Under the ROC Curve (AUC), Matthews Correlation Coefficient (MCC) and Cost of failure are the 3-evaluation metrics that would be used.**

Keywords—credit card, fraud detection, machine learning, deep learning, random forest, k nearest neighbor, support vector machine, autoencoder, restricted boltzmann machine, deep belief networks, convolutional neural networks

I. Introduction

Ever since the introduction of credit cards and online payments, many scammers have found ways to exploit people and steal their credit card information to use them for unauthorized purchases. This leads to a huge amount of fraudulent purchases every day. Banks and eCommerce websites are trying to identify these fraudulent transactions and stop them from happening again. With Machine learning and Deep Learning methods, they are trying to stop the fraudsters before the transaction is approved.

Machine learning is one of the hottest topics of this decade and a subset of Artificial Intelligence. More and more companies are looking to invest in machine learning to improve their services. Machine learning is a combination of various computer algorithms and statistical modeling to allow the computer to perform tasks without hard coding. The acquired model would be learning from the "training data". Predictions can be made or actions can be performed from stored experiential knowledge. Deep learning models are a part of machine learning techniques which involves Artificial Neural Networks. Convolutional neural networks, Deep Belief Network, Auto-encoders, Recurrent Neural Network, and Restricted Boltzmann Machine are all various methods. A properly trained NN would have the capability to capture unique relationships over the whole dataset.

Credit card fraud is a form of fraud involving the use of fake or stolen credit card information and causing financial harm to account holders or merchants involved. The total number of credit card fraud in Single Euro Payments Area (SEPA) in 2016 was 1.8 Billion Euros out of the total 4.38 Trillion Euros transaction, which is 0.4% lower than the previous year[1]. In 2015, according to the Nelson report, the total loss from the credit cards in the world was $21.84 billion and projected that in 2020 it would be $32 billion. [2]

In this paper, we will be looking into 3 data sets. They are the European dataset [3], the Australian dataset [4] and the German dataset [4]. In this work we aim to benchmark different ML and DL techniques. An ensemble of the best 3 performing models is also applied the all 3 datasets. We present our conclusions based on an empirical study comparing different ML and deep learning models.

The paper is organized as follows. Section 2 summarizes the related work and the background of the models implemented. Section 3 provides details on implementation and experimental setup. In section 4 we present and discuss the results. Conclusions and future work are summarized in sections 5 and 6 respectively.

II. Related Work

Tuyls et al. [11] outline several challenges regarding Fraud Detection. First, the highly unbalanced datasets in this application where only a small percentage of the available data is fraud makes training efficient models quite difficult. Besides other problems arise from noisy data and overlapping patterns. Most importantly the dynamics of frauds keep changing and classification models need to capture and adapt to this change. As follows we review some of the most relevant studies that have applied machine learning and deep learning models in the area of fraud detection.

A. A Comparative study on KNN and SVM

Zareapoor and Shamsolmoali [5] published a paper on Fraud Detection using different techniques namely Naïve Bayes, KNN, SVM and Bagging Ensemble Classifier. In their paper, they discuss the various concerns while handling this problem, such as there is a non-availability of real-world data which forces researches to work on faux data as the banks and other financial institutions don't make their data public due to privacy concerns as it is sensitive data. Also, most of the times the data is highly unbalanced as the number of fraudulent transactions is only 2% and 98% of the transactions are legitimate. They refer to the concerns of big data and the computation time it takes for larger datasets in these cases. One of the major challenges which frequently reported in many research papers is about the dynamic nature of fraud. The nature of fraud cannot be defined by one particular situation or style. Hence, it needs machine learning algorithms to be updated regularly so that malicious attempts could be caught in real time.

In their experiment, they used the data from the UCSD-FICO competition where it contained 100,000 instances of credit card transaction with 20 attributes from an e-commerce website. Only 2293 instance/transactions were fraudulent which makes the ratio of legitimate to fraud as 100:3.

The methodology they used was splitting the datasets into 4 different sets which contained 20% fraud, 15% fraud, 10% fraud and 3% fraud. In their experiment, they realized that the accuracy or error rate would not be the best evaluation measure in this case. Hence, they used True Positive Rate (fraud catching rate), True Negative Rate, False Positive Rate (false alarm rate) and False Negative rate. These four values help in reflecting the performance more than the accuracy and the error rate. They used a 10-cross-fold validation for their experiments. The final result showed that KNN had a much better false alarm rate and fraud catching rate as compared to SVM and Naïve Bayes Classifier on all four sub-datasets.

B. Random Forest in Fraud Detection

Randhawa et al's paper [6] on Credit card fraud detection using AdaBoost and majority voting explores many different machine learning algorithms such as Naïve Bayes, Random Forest, Gradient Boosted Tree, etc. In this paper, they use "Majority Voting" for combining two or more algorithms. The study also investigates AdaBoost ensemble model and reports that AdaBoost is very sensitive to anomalies and outliers.

They use the *RapidMiner* as an implementation Software and the experiments are conducted using the South-East Asia region credit card data. This dataset is a highly imbalanced data set with less than 1% fraud transaction. All of the classifiers were evaluated using 10-fold cross validation to reduce the bias. The classifier performance is evaluated using the Matthews Correlation Coefficient (MCC).

MCC helps in measuring the efficiency of the 2-class problem by taking TP, TN, FP and FN rates into account. In their experiment, they found that Random Forest had one of the best MCC rates at 0.990 compared to other methods such as SVM, gradient boosted trees, etc. By using AdaBoost, Random Forest with achieved 100% accuracy and MCC rate to be 1. In this study the generalization ability of the developed model should be carefully examined as it is not clear how good it would perform on unknown data.

In general, the study reports that hybrid approaches produce more reliable results compared to single classifier models.

C. Detecting Fraud using AutoEncoders based on Reconstruction Error

Tom Sweers in his bachelor thesis, [7] describes AutoEncoders as an effective neural network which can encode the data as it would learn to decode it as well. In this approach the Autoencoders are trained to non-anomaly points, introduced to the anomaly points to classify it as 'fraud' or 'no fraud' according to the reconstruction error which is expected to be high in the case of anomalies that the system has not been trained on. Here, any value above the upper bound value or threshold could be considered an anomaly. This is a method which was also used in Z. Chen el at's paper [8] on Autoencoder-based network anomaly detection.

Chen reported that stacked AutoEncoders worked better than for one hidden layer AutoEncoders in detecting anomalies. The AutoEncoder network he used had the structure 30-2-30, 30-10-30, 30-20-10-2-10-20-30 and 30-25-20-10-20-25-30, basically two with one hidden layer and two with 5 hidden layers. All this was built in python using Tensorflow with learning rate 0.01 and training the network for 100 epochs. For evaluating the performance, he used recall-at-k and precision-at-k where k is the number of instances.

With k at 1000, single hidden layer AutoEncoder worked much better compared to the stacked multilayered AutoEncoder. However, as k increased the stacked model produced much better results over single layer model.

D. Using Restricted Boltzmann for Fraud Detections

Restricted Boltzmann machines (RBM) can be used for data reconstruction in an unsupervised learning setting [9]. Pumsirirat and Yan's Paper [9] used Keras to implement this high-level NN. They used *H2O package* to find out the Mean Squared Error, Root Mean Square Error and Variable importance of the attributes in each dataset. Keras was used to get the Area Under the Curve and Confusion Matrixes for each case.

They found that RBM works much better in producing AUC and Accuracy for larger datasets (European) compared to the smaller datasets (German and Australian). This may be due to the fact that with smaller datasets, it's harder to recognize the fraud as the data is too sparse. However, with larger datasets its easier to detect "not-fraud" as there is a lot of data to learn/train from.

E. Using CNN for for detecting Suspicious Activity

In Chouiekha and El Haj's paper [10], Convolutional Neural Networks (CNN) are used for Fraud Detection. A database was created with 18000 artificial images of 300 customers' activity during 60 days. They used Customer Details Records in such a way that long conversation or an unusual number of vouchers used would be detected. CNN is applied to the images to detect fraudulent activity. 50% of the data set was used for training, 25% for validation and 25% of for testing. Images have been rescaled to improve classifier performance. The proposed Deep CNN(DCNN) contained 7 layers with 3 Convolutional layers, 2 pooling layers, 1 full connected layer and finally 1 SoftMax regression layer.

Results were evaluated using accuracy. Deep CNN's performance is compared against SVM, Random Forest and Gradient Boosting Classifier (GBC). The results show that DCNN outperforms SVM by 5%, Random Forest by 10% and GBC by 3%. Deep CNN was found to train almost twice faster than the rest of the methods.

F. Neural Networks vs Bayesian Belief Network

The study in [11] compares Bayesian Belief Network (BBN) against Artificial Neural Networks. BBN are found to better detect fraud cases and to use less training time. ANN however had faster prediction times when applied in real time.

G. Summary and Motivation

Various Machine Learning techniques have been used to detect frauds such as SVMs, KNN, K-Means, Random Forest, Naïve Bayes and many more. Most of them deal with unbalanced data sets where the amount of fraud is very low. The main evaluation metrics for fraud detection are True Positive Rate, False Negative Rate and Matthews Correlation

Coefficient. Some studies recommend using Neural Networks as a solution for unbalanced datasets.

In Tom Sweers' thesis [7], his methodology of introducing Autoencoders to normal data and detecting fraud based on reconstruction error is unique. In Pumsirirat et al's paper [9], we can see that AUC and Confusion matrix can be used to evaluate the models. We also see that how deep learning fails when there is less instances in a dataset (Australian and German). Working with smaller dataset failed to achieve good prediction scores. This shows that not always deep learning would able to solve problems for smaller dataset. While Chouiekha et al. [10] report that DCNN and outperforms SVMs, Random Forest and Gradient Boosted Classifier, Tuyls et al. [11], confirm that Artificial Neural Networks have a much faster fraud catching process compared to Bayesian Belief Networks.

In this study we present an empirical comparison of various machine learning and deep learning models inspired by the previous studies. Our intention is to investigate the performance of various models using data sets with different sizes, complexities and characteristics. The goal is to come up with recommendations on best practices of picking the most suitable model for a fraud detection application given the data of particular description.

In particular we compare the performance of SVM, KNN and Random forest to Deep Learning methods such as AutoEncoders, RBM, DBN and CNN. Since we will be working with three different datasets, we would be able to see to what extent these machine learning techniques hold valid. We also consider fine-tuning methods such PCA, reduction of features, cleaning data, hyperparameters etc. to improve classifier performance.

Furthermore, we investigate combining the 3 best performing models in an ensemble using majority voting. In the next section we describe the data used, the experimental setup and the main learning models.

III. Implementation and Analysis

A. Data Sets

In this study our experiments will be conducted on 3 data sets. The European Dataset contains the transactions made by credit card users in 2 days in September 2013 [3]. All of the fields except time and amount have been PCA transformed. It contains only 492 fraud instances out of 284,807 instances.

Both the Australian Dataset and the German Dataset [4] are acquired from the UCI ML repository. The data sets were anonymized such that no personal information is provided.

The Australian contains 383 normal instances and 307 fraud instances. In the same time the German data set contains 1000 instances; of which 700 of them are normal instances and 300 of them are fraud instances. The European dataset is considerably larger than the Australian and the German datasets.

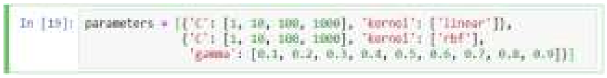
Our experiments try to investigate the effectiveness of different machine learning and deep learning models using data set with varying size and complexities.

B. Experimental Setup

All of the implementation were conducted in Python and using various libraries such as NumPy, Pandas, Keras, Scikit-Learn, and Tensorflow. Rstudio was occasionally used for data cleaning.

For **K-nearest neighbor**, we use cross-validation on the training data set to determine the best value of the neighbors K for each data set. The best K for each data set is then used to conduct further analysis on the entire data set.

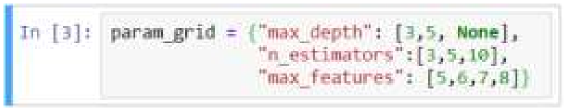
For **Support vector machines** and **Random Forest** we use a grid based search approach to find the best parameter for each model. We use the Python method G*ridSearchCV*, with the parameters depicted in figures 1 and 2 for the SVM and the Random forest, respectively.



```
In [15]: parameters = [{'C': [1, 10, 100, 1000], 'kernel': ['linear']},
                       {'C': [1, 10, 100, 1000], 'kernel': ['rbf'],
                        'gamma': [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]}]
```

Fig. 1. Parameters for SVM

The best parameters are used to evaluate the models with the entire data.



```
In [3]: param_grid = {"max_depth": [3,5, None],
                      "n_estimators":[3,5,10],
                      "max_features": [5,6,7,8]}
```

Fig. 2. Parameters for Random Forest

The basic idea behind the **Autoencoders** is that it could reconstruct its input. So here for fraud detection, we train the autoencoders only on the normal transaction. While running the experiment on the test data, it would produce reconstruction errors for each of the instances. The expectation is that normal transactions would produce lower reconstruction errors while fraud transactions/instances would produce higher values. A certain threshold value is set, such that if reconstruction error is above this threshold then the particular instance/transaction is fraudulent. Otherwise the transaction is considered. In our experiments we test different threshold values and present the related results.

Similar to Autoencoders, the **restricted Boltzmann machine (RBM)** produces free energy that is again tested against a threshold to determine normal vrs fraudulent transactions. The RBM model used here is created by Weiman Wang for fraud detection. [12]

For **deep belief network**, we use an adapted version of the model by AlbertUP, which is implemented in Tensorflow for supervised and unsupervised pattern recognition problems [13].

With **CNN**, we convert the dataset into a 2D array instead of a 1D array. The data passes through a series of convolutional layers and max-pooling layers followed by a layer to flatten the data. Finally, the data is classified at SoftMax layer. Fig 3 shows the architecture of the CNN we use.

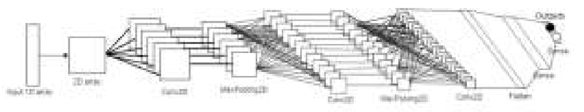


Fig. 3. CNN Architecture

We test our models using cross-validation and choose the **top three** performing models and combine them using majoring voting. The diagram below depicts the basic structure of the model.

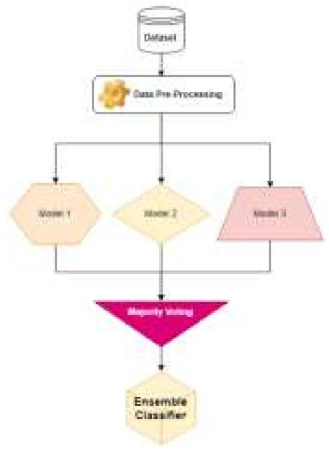


Fig. 4. Majority Voting Based Model Structure

C. Evalulation Metrics

As follows we describe the main evaluation criteria we consider in this study.

The Matthews Correlation Coefficient is a measure to evaluate the quality of a two class/binary classifier. It was proposed by Brain W. Matthews in 1975. The coefficient returns +1 for a perfect prediction, while a value zero indicates a random prediction. Matthews Correlation is also known as the phi coefficient. Davide Chicco mentions that MCC is a much better measure than accuracy and F1 score as the other two can be misleading because they do not consider all four values of the confusion matrix [14].

ROC curve is the receiver operation characteristic. It helps in determining the precision of the model because of the imbalance in the dataset. ROC curve is basically plotting of TPR on the x axis against FPR on the y axis. Sometimes two ROC curves may have the similar Area under the curve (AUC), then in that case, we need to look further into the finer details such as the Cost of failure.

The idea behind the cost of failure is that each of the False Negatives (Frauds detected as Normal) would have a cost of $1000 and False Positives (Normal instances detected as fraud) would have a cost of $100 to the company/entity. We are using this method to evaluate the top three models because sometimes all three of them have very similar MCC and AUC values. Similarly, the cost of the resulting ensemble classifiers is also calculated.

IV. Results

As follows we present the results obtained from the experiments on the European Dataset, the Australian Dataset and the German data set.

A. European Dataset

TABLE I. EUROPEAN DATASET RESULTS

Method			
	MCC	***AUC***	***Cost of Failure***
RBM	0.176	0.9109	**227360**
Autoencoders	0.2315	0.8943	**127220**
Random Forest	0.7947	0.8507	**30340**
CNN	0.8096	0.8764	**25700**
SVM	0.8145	0.9004	**21220**
KNN	0.8354	0.8887	**22660**
Ensemble (KNN, SVM and CNN)	**0.8226**	**0.8964**	**21740**

Table I summarizes the results obtained on the European Dataset. In particular the table lists the Matthew Correlation Coefficient (MCC) and the Area Under the Curve measure (AUC) for a set of machine learning models.

RBM and AE have high false positives (false alarm rate) and therefore perform poorly with respect to MCC and cost. Random Forest has good AUC and MCC values. CNN, SVM, and KNN have the best performance in terms of MCC and AUC. We can see the SVM has the least in terms of cost of failure while Autoencoders and RBM have the highest. Random forest, although producing good results, is poor in terms of the cost. The top three performing models for this data set are SVM, KNN and CNN.

The majority voting classifier is constructed from the top 3 performing models. The ensemble method performs better than SVM and CNN individually, however it has a similar cost to SVM. However, SVM has a better AUC value. Here, the recommendation would be to choose SVM instead of the ensemble if the company is looking to reduce the cost as much as possible as the ensemble method would take longer time in terms of both training and testing while SVM has the least in terms of testing and training.

B. Australian Dataset

TABLE II. AUSTRALIAN DATASET RESULTS

Method			
	MCC	***AUC***	***Cost of Failure***
RBM	0.15	0.5546	**24600**
Autoencoders	0.2318	0.6174	**12220**
CNN	0.6408	0.8227	**6430**
Random Forest	0.684	0.8416	**4700**
KNN	0.6905	0.8425	**6460**
DBN	0.6999	0.8441	**6790**
SVM	0.7085	0.8551	**3380**
Ensemble1 (KNN, SVM, DBN)	**0.7144**	**0.8573**	**5290**
Ensemble2 (KNN, SVM, Random Forest)	**0.7281**	**0.8655**	**3470**

Table II summarizes the results obtained on the Australian Dataset. We can see that RBM and AE have the worst performance of all the methods. SVM, DBN, and KNN have the best in terms of AUC and MCC. Random Forest and CNN also have good values compared to the others. We choose two ensemble models. The first ensemble model (*Ensemble 1*) the is composed of 3 classifiers: the KNN, DBN, and SVM. A second ensemble (*Ensemble 2*) is build using the models with the least cost of failures: KNN, SVM and Random Forest. RBM and AE have the maximum cost of failure as they again have a lot of false positives due to the threshold.

Table II shows that using *Ensemble 1* (KNN, SVM, DBN) has improved the MCC and AUC performance compared to single SVM and the other methods. However, the cost is higher than the random forest and SVM. This is because KNN and DBN have a high cost of failure values, which likely influenced the classification. *Ensemble 2* (KNN, SVM, Random forest) on the other hand, which was based on combining classifiers with the least cost of failure methods, classifier achieved a higher MCC, AUC, and lower cost value. Here we can see that combining methods have provided better results overall and *Ensemble 2* is the best method of all as it has the best MCC, AUC values and the lower cost.

C. German Dataset

TABLE III. GERMAN DATASET RESULTS

Method			
	MCC	***AUC***	***Cost of Failure***
RBM	0.0984	0.5524	**14160**
Autoencoders	0.139	0.5614	**22640**
KNN	0.2487	0.6047	**21100**
DBN	0.2725	0.5873	**23640**
Random Forest	0.2912	0.6437	**16970**
SVM	0.4038	0.6857	**16400**
CNN	0.4291	0.7056	**14220**
Ensemble (SVM, CNN, Random Forest)	**0.4439**	**0.7011**	**15620**

Table III summarizes the results obtained for the German Dataset. Examining the results of the table we can see that SVM, Random Forest and CNN are the best models in terms of performance (AUC and MCC values). Random forest, CNN and SVM also have a better cost of failure than other models. Therefore, we use an ensemble of these three models to build the majority voting classifier.

In general, the results depicted in tables I, II and III for the three data set investigated show that combing the top-performing models outperform single models. Ensemble improvement is more obvious on the smaller data sets (the German dataset and the Australian dataset). For the European dataset, it produces results a little less than SVM.

Random Forest works best for smaller datasets. Convolutional Neural Networks was found to be the best deep learning method as it produces good results for both European and German dataset, while its performance on the Australian dataset was the 4th best and it cost of failure was similar to KNN. For the German dataset it had the lowest cost as well.

Table IV summarizes the frequency of the individual models ranking at the top 3 performing models for all the datasets. SVM consistently was among the best performing models of all data sets. KNN also produces good results with both the large dataset and the smaller datasets.

TABLE IV. TOP PERFORMING MODELS

Method	
	Number of times in Top 3
Support Vector Machines	3 Times
K-Nearest Neighbors	2 Times
Convolutional Neural Networks	2 Times
Random Forest	2 Times
Deep Belief Network	1 Time

V. CONCLUSION

Research related to Fraud Detection has been around for over 20 years now and has used various methods from manual checking to customer end authentication. Machine learning models have also had wide successes in this area. Deep learning models have been recently adopted in many applications enabled by the rise in higher computation power and cheap computing cost.

This paper provides an empirical investigation comparing various machine learning and deep learning models on different data sets for the detection of fraudulent transaction. The main aim of this study is to find insights of which methods would best suitable for which type of datasets. As nowadays, many companies are investing in new techniques to improve their business this paper could potentially help practitioners and companies to better understand how different methods work on certain types of datasets.

Our study reveals that to detect fraud, the best methods with larger datasets would be using SVMs, potentially combined with CNNs to get a more reliable performance. For the smaller datasets, ensemble approaches of SVM, Random Forest and KNNs can provide good enhancements. Convolutional Neural Networks (CNN) usually, outperforms other deep learning methods such as Autoencoders, RBM and DBN.

A limitation of this study is however that it only deals with detecting fraud in a supervised learning context. Although supervised learning methods such as CNN, KNN, Random Forest seem attractive and produce good results, they do not work well for dynamic environments. Fraud patterns typically change over time and would be hard to catch. New data sets would need to be collected and machine learning models need to be retrained.

Autoencoders provide a good solution in that case as they are only trained on normal (i.e. non-fraudulent) traffic. Fraudulent transactions are detected as deviation from the normal patterns. Although training of Autoencoders is initially quite costly, it can be useful for the labelling of data sets. Once enough data is labelled, it can be used to retrain or build other supervised models.

VI. Future work

The idea of implementing neural networks for fraud detection is worth further exploration. The main problem with obtaining good NN models is the availability of appropriate training data. We are currently looking into more data sets and availability of larger sized fraud transactions to be able to train larger models.

We are also looking into other models of deep learning networks. In particular Generative Adversarial Networks (GANs) have been used for anomaly detection in the field of cybersecurity to detect cyber-attacks. Some research work has used GANs to simultaneously train both the generator and discriminator on healthy eyes data to identify anomalies in the human eyes. This could be extended to the application of fraud detection.

Fuzzy systems for anomaly/outlier detection can also be paired with a neural network to achieve better results and to provide more interpretable models. Genetic programming can also be used to provide optimization for neural network parameters and architecture.

In general fraudsters keep getting intelligent and it will remain challenging to catch them. Therefore, the development of hybrid, dynamic and adaptable systems for fraud detection will pose many open areas of research in the future.

References

[1] European Central Bank, “Fifth report on card fraud, September 2018,” 26 September 2018. [Online], Available: https://www.ecb.europa.eu/pub/cardfraud/html/ecb.cardfraudreport201809.en.html.

[2] N. Report, "The Nilson Report," The Nilson Report, 2016. [Online]. Available: www.nilsonreport.com.

[3] Credit card fraud detection anonymized credit card transaction labeled as fraudulent or genuine [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud.

[4] Dheeru Dua and Casey Graff. UCI Machine Learning Repository. 2017 [Online]. Available: http://archive.ics.uci.edu/ml/datasets/

[5] Masoumeh Zareapoora, Pourya Shamsolmoalia. “Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier” International Conference on Intelligent Computing, Communication & Convergence (ICCC-2015). Procedia Computer Science 48 pp 679 – 686. 2015.

[6] Kuldeep Randhawa, Chu Kiong Loo, Manjeevan Seera, Chee Peng Lim, Asoke K. Nandi. “Credit card fraud detection using AdaBoost and majority voting”. IEEE Access (Volume: 6), pp 14277 – 14284. 2018.

[7] Tom Sweers. “Autoencoding Credit Card Fraud”. Bachelor Thesis, Radboud University. June 2018.

[8] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau. “Autoencoder based network anomaly detection.” Wireless Telecommunications Symposium, pp 1-5. 2018.

[9] Apapan Pumsirirat, Liu Yan. “Credit Card Fraud Detection using Deep Learning based on Auto-Encoder and Restricted Boltzmann Machine”. International Journal of Advanced Computer Science and Applications, Vol. 9, No. 1, 2, pp 18-25. 2018.

[10] Alae Chouiekha, EL Hassane Ibn EL Haj. “ConvNets for Fraud Detection analysis”. Procedia Computer Science 127, pp.133–138. 2018.

[11] S. Maes, K. Tuyls, B. Vanschoenwinkel, B. Manderick. “Credit Card Fraud Detection Using Bayesian and Neural Networks”. 2002. [Online]. Available: https://www.researchgate.net/publication/2524707_Credit_Card_Fraud_Detection_Using_Bayesian_and_Neural_Networks.

[12] Weiman Wang, Restricted Boltzmann Machine. GitHub. Aug 2017. [Online] Available: https://github.com/aaxwaz/Fraud-detection-using-deep-learning/blob/master/rbm/rbm.py.

[13] Albertbup, “A Python implementation of Deep Belief Networks built upon NumPy and TensorFlow with scikit-learn compatibility”. GitHub. October 2018. [Online] Available: https://github.com/albertbup/deep-belief-network.

[14] D. Chicco, "Ten quick tips for machine learning in computational biology". BioData Mining, December 2017, pp 1–17.