Journal of Theoretical and Applied Information Technology
31st July 2022. Vol.100. No 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

CREDIT CARD FRAUDS SCORING MODEL BASED ON DEEP LEARNING ENSEMBLE

SHUMUKH AL-FAQIR[1],OSAMA OUDA[2]

Department of Computer Science, College of Computer science and information, Jouf University, Saudi Arabia

E-mail: 421204018@ju.edu.sa, omalsayed@ju.edu.sa

ABSTRACT

Credit card frauds can result in substantial financial losses, particularly when fraudulent transactions have large values. Thus, it is essential to detect fraudulent transactions prior to their authorization by card issuers. Most conventional fraud detection systems are based on machine learning models. Recent studies explored utilizing deep learning (DL) models to detect fraudulent transactions efficiently. However, such studies depend merely on a single DL model. In this paper, we present various deep learning and ensemble methods for detecting credit card fraudulent transactions. The main motivation behind this presented work is to contribute toward reducing both missed frauds and false alarms, where our contribution in this work lies specifically in combining the resulting scores of three different distinct DL models, namely, convolutional neural networks (CNN), autoencoders (AE), and recurrent neural networks (RNN). Experiments on a public credit-card dataset demonstrated that, for the single DL-based models, AE has the best validation accuracy (93.4%) compared to CNN (91.4%) and RNN (91.8%). For the ensemble results, the validation accuracy (94.9%) was superior to all the three implemented DL-based models.

Keywords: *Deep Learning, Convolutional Neural Networks (CNN), Auto Encoders (AE), Recurrent Neural Networks (RNN), Ensemble Learning.*

1. INTRODUCTION

With the recent increase in websites and mobile applications, online financial transactions have become widely used for purchasing all sorts of products and services over the internet due to the many advantages such as ease of use, the instant purchasing experience, the ability to purchase online at any time during the day, from any place on the go. The problem to be stated here is that this ease of conducting online financial transactions has led to a severe threat which is the massive spread of fraudulent online transactions among users, where financial credentials get stolen by unauthorized users, causing significant financial loss among users over the internet, which is a severe problem that needs to be seriously tackled using the latest programming techniques. [1]

As a result, researchers started to add security layers on top of online websites and applications to secure users' financial credentials. These security layers are meant to provide users with a safe and secure purchasing experience over the internet and hinder attackers from stealing users' financial credentials, such as credit card number, the date of expiring, and the card security number, with the purpose of either buying goods and services without paying for it or stealing unauthorized amounts of funding from banking accounts. [2]

In addition to securing financial credentials, there has become a need to use detection systems that aim to detect any suspicious behavior while conducting online financial transactions. Credit card companies have started hiring specializing investigators to check the fraud financial transactions; although this is considered a time consuming and a tedious task to analyze each fraudulent transaction, they end up analyzing only a few fraud transactions each day, leaving the rest of the fraud transactions unchecked, while on the other hand, users send fraud reports to the credit card company whenever they notice and realize the fraud transaction. [3] Such situations led to the need to build automatic fraud detection mechanisms that would analyze and automatically detect financial credit card fraud transactions.

Moreover, these fraud detection mechanisms have got to be highly accurate; this means that such systems need to be immune to false detection alerts to avoid the cancellation of legal and financial transactions and hence avoid the loss of legal funding. For instance, the famous credit card company Mastercard has stated that during 2014, $118 billion dollars were lost based on false alarms of fraud detection, while $9 billion dollars were lost to actual fraud transactions; this is due to the usage of fraud detection based traditional fraud detection system [4]. As a result, there has become a need to use detection systems that are based on real datasets being fed into machine learning and deep learning algorithms that would automatically and deeply learn the fraud patterns and relations being found in such fraud datasets and hence build a prediction or a classification model that would improve and enhance the fraud detection mechanisms effectively and in an accurate manner.

In this work, we aim to use machine learning algorithms to learn previously recorded suspicious behavior being detected during fraudulent financial transactions to develop a fraudulent financial detection system. To do this, we read through the literature review to see where the research has reached so far. We have noticed that machine learning and deep learning is the dominant search direction when it comes to building and evaluating credit card fraud detection mechanisms. Therefore, we will experiment with deep learning algorithms such as Convolutional Neural Networks (CNN), Auto-Encoders (AE), and Recurrent Neural Networks (RNN) to develop a classification model that can be used for finance fraud detection, and finally build an ensemble classification model, using Majority Voting Ensemble (MVE) method, this model is going to be achieved by combining the scores of the three different distinct DL models, namely, CNN, AE, and RNN, hence contribute towards building a fraud classification model that gives the best, highest and most accurate classification rates based on the ensemble combination of the three build models of CNN, AE, and RNN and calculating the majority votes of the labels being predicted to classify the labels accordingly and compute the ensemble accuracy rate.

The rest of this paper is organized as follows: Section 2 lists a number of previous related works, section 3 introduces the proposed ensemble methodology, how the dataset is being preprocessed, and the building of the three classifiers with three deep learning algorithms, which are the CNN, the AE, and the RNN. Section 4 describes the used dataset and mentions the imbalance issue and the features being extracted and selected. Section 5 discusses the results of the experimentations, section 6 concludes the paper, and finally, the references are listed in section 7.

2. RELATED WORKS

In 2018, D. Choi and K. Lee from Korea [5] conducted a survey through the machine learning techniques to be used in fraud detection in the IoT environment to see the best algorithms and implement them. Their experimentation used actual financial data in Korea, then experimented with clustering, classification, and deep learning algorithms, giving relatively high accurate results among these implemented algorithms.

Y. Abakarim et al. [6] have proposed a real-time fraud detection mechanism based on deep learning integrated with Auto-Encoder to be applied in banks and financial institutions to detect fraudulent credit cards based on legitimate transactions. For the experimentation, the authors have used four binary classification models, and for the analysis, they have calculated Accuracy, Recall, and Precision.

A. Roy et al. [7] have also experimented with deep learning algorithms to evaluate their inner topologies of the hidden layers of complex neural networks subsection by subsection, with the usage of built-in timer and memory components and test it with different parameters to evaluate in which it can giver better results with higher performance in terms of classification accuracy. Their experimentation was based on a dataset that contained 80 million credit card fraud detection records being gathered from a distributed cloud computing environment. Furthermore, they experimented with handling the issues of class imbalance and scalability.

A. M. Mubalaike. And E. Adal [8] have also experimented with deep learning to be used against financial fraud detection systems, where they have experimented with Ensembled decision tree and stacked autoencoders on a data set extracted from accurate financial logs of a financial company in Africa, where it contains over six million financial transactions. Moreover, they calculated the performance criteria such as Accuracy, Sensitivity,

Specificity, and Precision to assure the validity of their proposed classification model.

In [9], R. Zhang et al. have proposed a "sequential behavioral data processing" technique using the Recurrent Neural Network (RNN) algorithm while integrating it with Markov Transition Field (MTF), showing that this sequential integration will give better performance when used for fraud finance detection for online shopping and online money transactions.

In 2019, [10] X. Zhang et al. have proposed a new methodology regarding how to apply deep learning algorithms for fraudulent financial transactions, where they introduced the usage of features engineering based on "*homogeneity-oriented behavior analysis- HOBA*" to help in signifying the needed information in the dataset that is related directly to the suspicious behavior during fraudulent credit card transactions. For the dataset, they used a real dataset from a commercial bank in China containing fraudulent credit card financial transactions.

In [11], H. Ye et al. have experimented with many machine learning algorithms to detect financial fraud detection, and they found that Random Forest has given better results than the other tested algorithms such as Artificial Neural Networks (ANN), Support Vector Machine (SVM) and Logistic Regression (LR). The Random Forest algorithm was tested while integrated with "SMOTE: Synthetic Minority Over-sampling Technique" this is to signify the features that are mainly used to recognize suspicious behavior during finance fraud detection. For the experimentation, they used a dataset that contains 11726 fraudulent financial Chinese transactions that took place from 2007 to 2017.

E. Kim et al. [4], have experimented with deep learning and ensemble learning, where they have applied DNN, CNN, RNN, and then the combination of them as an ensemble, giving their comparisons among these classifiers using a massive dataset from a famous credit card company in South Korea. The learning process was conducted in two phases; the first is offline during conducting the experimentation, while the second phase is after launching the classification model for a month. Important techniques were mentioned, such as imbalanced dataset handling and the early stopping techniques, where the iterations of the learning process stop when there is no improvement in the accuracy rates.

In 2021, S. Sanober et al. [3], have proposed a deep learning-based fraud detection system in wireless networks, where they have used a dataset that captured financial transactions taking place by credit cards during September 2013 in the European countries, where 492 fraudulent transactions were detected out of 284.807, which is 0.172% of fraudulent transactions among all online financial transactions. For the experimentation, they have experimented with many machine learning algorithms such as Random Forest, SVM, KNN, Decision Tree, and Logistic Regression. For the analysis, they have given their computed results for Accuracy, Precision, Specificity, and F1-Score.

J. I. Chen et K. L. Lai [12] have proposed a Deep Convolutional Neural Network (DCNN) to be used in real-time credit card fraud detection with the aim of learning complex patterns of fraud data dynamically. Their dataset contains 5 million transactions with only 6223 fraud records, which is considered imbalanced, and it was handled using the undersampling technique. For evaluation, their built model was tested against other machine learning techniques such as LR and SVM models to show that it gave a higher performance of 99%.

3. PROPOSED METHODOLOGY

Although DL-based classification models demonstrated superior performance than ML-based models for various classification problems, including credit-card fraud detection [6] , [7], [11], classification models based on a single DL or ML technique could not be sufficiently accurate due to prediction errors resulting from high variance. To tackle this issue, ensemble techniques have been widely used in machine learning to combine the outputs produced by different predictors and improve the final prediction accuracy. It has been shown that ensemble methods can produce reliable estimates compared to those achieved through a single model. Generally, ensemble techniques can be linear or non-linear. Linear ensemble techniques can use simple or weighted averaging, whereas non-linear ensemble techniques obtain an ensemble output using a black-box model as non-linear kernels.

In this paper, our focus is to improve the accuracy of detecting credit-card frauds by combining the outputs of three different DL models,

Journal of Theoretical and Applied Information Technology
31st July 2022. Vol.100. No 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

namely Convolutional Neural Networks CNN, Recurrent Neural Networks RNN, and Autoencoders AE. Because credit card fraud detection is a binary classification problem where the model output is a fraud/non-fraud transaction decision, we utilized the majority voting ensemble to implement our proposed ensemble method. Precisely, as illustrated in Figure 1, we utilize a hard majority voting ensemble in which the final prediction is decided with the largest sum of votes from the contributed models, as shown in Equation (1):

$$\hat{y} = \frac{1}{|S|} \sum_{k \epsilon S} \hat{y}_k, \quad (1)$$

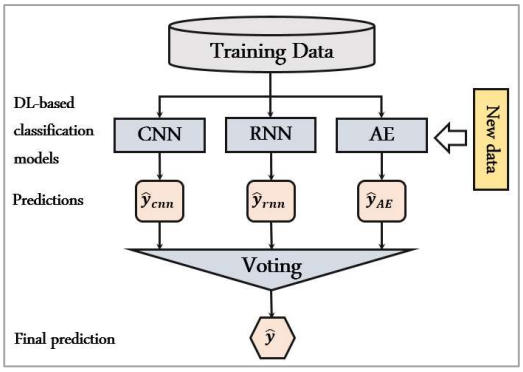

Figure 1: Proposed majority voting ensemble DL classifier

where S = {CNN, RNN, AE} is the set of utilized models of models` indices, and $|S|$ is the size of S. The following subsections describe how we utilized each single model to detect credit card transaction fraud

3.1 Fraud detection using CNN classifier

A Convolutional Neural Network (CNN) is considered one of the most important structures used in deep learning models. It has been used in datasets that contain highly related features, where it achieves good performance and results; it is also good in avoiding the problem of model over-fitting due to its ability to create complicated CNN models that could implement complex jobs with huge sized datasets [16].

Convolutional Neural Networks contain multiple layers of neurons which are considered statistical functions that compute the summation of several inputs and outputs with an activation value. The first layer of CNN often detects basic features of the input image. The output of each layer is supplied as input for the next layer, which extracts more complicated attributes each time. As we go deeper into the CNN, the layers begin detecting higher-level features, as shown in Figure 2.

The model will calculate the intermediate value *Z*, resulting from the input data convolution from the previous layer with *W* tensor, which contains filters and the bias vector *b*. The model applies non-linear activation function *g* on the intermediate value [17]. Equation (2) represents the CNN function as follows:

$$A^{[i]} = g^{[i]}\left(W^{[i]} A^{[i-1]} + b^{[i]}\right) \quad (2)$$

The structure of our CNN model constructs Convolutional Layers, Batch Normalization Layers, Dense Layers, Max-Pooling Layers, and A Full Connection Dense Layer with Sigmoid. Batch

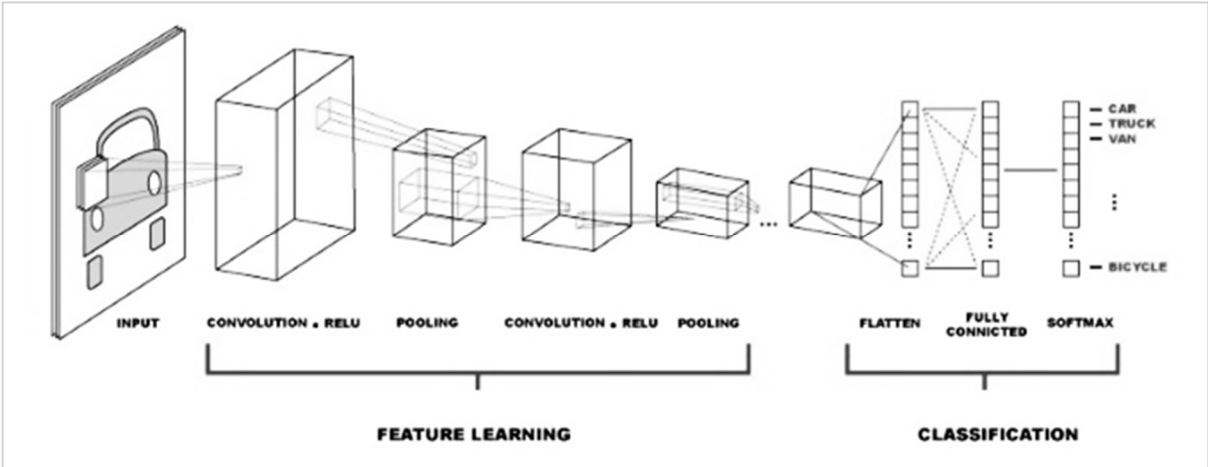

Figure 2: Structure of CNN [16]

Normalization allows every layer of the network to do the learning process more independently in each learning epoch. The Max-Pool Layer is used to avoid the Over-Fitting problem and improve that Accuracy value. The Dense Layer with the Sigmoid activation function was the last layer, where Sigmoid was used for the binary classification as in our case (normal and fraud classification). We used Adam optimizer in the model compilation with Accuracy metric and Loss function being set to Binary Cross-entropy, which in general can be represented as shown in Equation (3) as follows:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i.\log\big(p(y_i)\big) + (1 - y_i).\log\big(1 - p(y_i)\big), \quad (3)$$

where *y* is the label, and *p(y)* is the predicted value of the point being either fraud or normal for all *N* points.

3.2 Fraud detection using Autoencoder

Auto-Encoder is considered an unsupervised deep learning algorithm where it is developed to learn from a high-dimensional dataset with low-dimensional features. This model encodes the input data in the encoder model while trying to decode and reproduce the data during the decoder model so that the number of input instances should be equal to the number of the output instances. An auto-encoder includes two components: Encoder and Decoder, as shown in Figure 3 [18]. The input (the number of features) is compressed by the Encoder and reduced to a lower input size and generates some representation accordingly so that this representation can be used later in the decoder model. During the implementation, we had 29 features that were reduced by the Encoder into 20 features. The Encoding function can be represented in Equation 4 as follows:

$$h = f(x) \quad (4)$$

The second component, which is the Decoder builds the input again using this representation. The Decoder function can be explained in Equation (5) as follows:

$$r = g(h) \quad (5)$$

The standard Auto-Encoder function can be simply described in Equation (6) as follows:

$$g\big(f(x)\big) = r \quad (6)$$

Where we want ***r*** to be equal to the input, and the standard Auto-Encoder model architecture is represented in Figure 3.

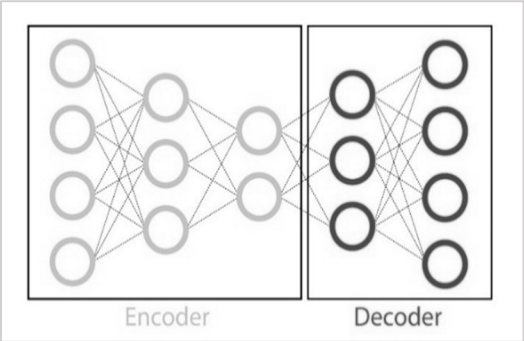

Figure 3: Autoencoder model components

To explain the Auto-Encoder equations in more detail, we have a data model *A,* and we have *f* features; the Encoder output *E* describes the reduced size of *A*, then the Decoder should rebuild the primary data *A* from the reduced representation E. Encoder, and Decoder use an activation function a_f, matrix of weight W_m and bias vector b_x [19]. Accordingly, the encoder equation is represented in Equation (7) as follows:

$$E = f(A) = a_f(W_m A + b_x) \quad (7)$$

While the Decoder equation is represented in Equation (8) as follows:

$$A = d(E) = a_f\big(W_m E + b_y\big) \quad (8)$$

Where *d* is the Decoder function that maps the *E* coding (representation) back to rebuild the data *A*.

Our Auto-Encoder model has learned the features of the legal records, and their input will be close to the output when performed. But for the case of fraud transactions (anomalies), the input will be different from the output because it is considered unexpected data. We used the Sequential model with the "Encoder" parameter to build the encoder model and the Sequential model with the "Decoder" parameter to build the decoder model. Then we built the final Auto-Encoder model using both models of Encoder and Decoder.

Journal of Theoretical and Applied Information Technology
31st July 2022. Vol.100. No 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

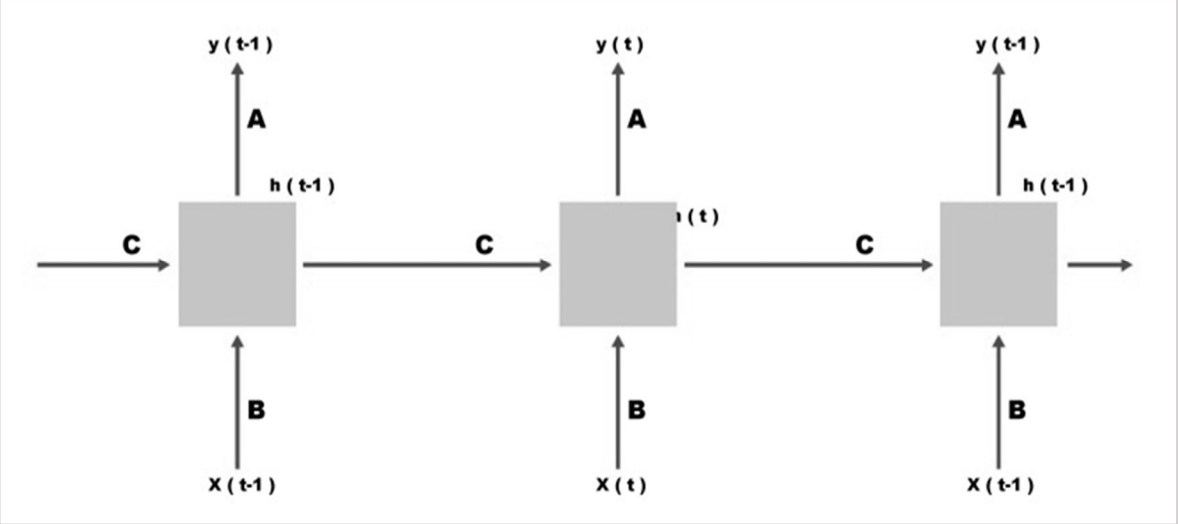

Figure 4: The architecture of Simple RNN algorithm

Algorithm 1: Auto-Encoder procedure

Input: T_E= training_epochs
B_S= batch_size
a = input matrix
L_R= learning_rate
$\Theta = (W_m, b_x, b_y)$
Output: AE model, evaluation stats
for 0 to T_E ***do***
for 0 to B_S ***do***
$f(A) = a_f(W_m A + b_x)$
$d(E) = a_f(W'_m E + b_y)$
$L1(\Theta) = \sum_{k=1}^{n} \|a_k - d(f(a_k))\|^2$
$L2(\Theta) = \sum_{k=1}^{n} \|a_k \log(b_k) + (1 - a_k) \log(1 - b_k)\|^2$
$\Theta = min_{\Theta} L(A, A')$
C = compute the cost with respect to Θ
foreach Θ_n, C_n in (Θ, C) ***do***
$\Theta_n = \Theta_n - L_R * C_n$
end
end
end

We used Dense layers with both models. We set the activation function as 'relu' with the first layers while the sigmoid activation function was used in the last fully connected layer. The Mean Squared Error (MSE) was used as a Loss Function with an Accuracy metric to compile and fit the model. The MSE loss function L can be generally represented as shown in Equation 9 as follow:

$$L = \sum_{k=1}^{n} \|a_k - d(f(a_k))\|^2 \quad (9)$$

After we got the result of MSE, we calculated the threshold value (cut off) so that we could classify any transaction that is above this calculated threshold as a fraud transaction. The general algorithm of our Auto-Encoder model can be found in Algorithm 1.

3.3 Fraud detection using RNN classifier

This algorithm is one of the most common deep learning techniques that depend on neural networks. It can effectively process and handle huge amounts of data in sequential nature, and it is perfect for model construction. It works on saving the helpful output of a specific layer and feeding it back as input to be processed with current data and helps in predicting the layer's output [20]. The architecture of the RNN algorithm is illustrated in Figure 4. The three variables X, Y, and Z are considered as the input layer, the hidden layer, and the output layer, respectively. The network parameters A, B, and C are used to improve the performance of the model. At any t time, the input is a combination of *x(t)* and x*(t-1)*. For the output, it is fetched back into the network as an improved output at any given time t. The activation function was used in the hidden layer of the RNN as a hyperbolic tangent function (tanh). Accordingly, RNN computation is explained in Equation (10) as follows:

$$y_t = tanh(w_{ht} y_{t-1} + w_{xt} x_t + b_t) \quad (10)$$

Algorithm 2: RNN procedure

Input: T_E= training_epochs
B_S= batch_size
x = input matrix
L_R= learning_rate
$\Theta = (W_m , b)$
layerType = simpleRNN, fully-connected, dense fully-connected;
g = activation_function = relu, sigmoid
Output: RNN model, evaluation stats
for 0 to T_E ***do***
for 0 to B_S ***do***
$y_t = g(w_{ht}y_{t-1} + w_{xt}x_t + b_t)$
$L(\Theta) = -\frac{1}{N}\sum_{i=1}^{N} y_i.\log(p(y_i)) + (1 - y_i).\log(1 - p(y_i))$
end
end

RNN gave excellent results in solving problems, and it is characterized by accepting the current input data and previously received inputs. RNNs can memorize previous inputs due to their internal memory. Its structure is represented as a cyclic deep learning architecture and a type of neural network in which connection between nodes exists along a temporal sequence and takes the form of a directed graph [21].

In our model, we used SMOTE (Synthetic Minority Over-sampling Technique) to handle the imbalance in the dataset. Then, the sequential model was used, fully connected Simple RNN layer and two Dense layers to build the model structure. We used (Binary Cross-entropy) as the Loss Function and (Adam) optimizer and 100 epochs for compiling the model where accuracy values get increased in each epoch iteration. The general algorithm of our RNN model is given in Algorithm 2.

3.4 Ensemble Modeling

This model is an approach that targets combining the predictions of several different modeling algorithms to give the best possible scores and hence improve the accuracy and the overall performance. Multiple ensemble modeling methods could be used, such as Ensemble_prediction, weighted averaging, majority voting, AdaBoost, and others. These methods are techniques that combine various models to produce one optimal predictive model with enhanced and more accurate results compared to the used single model.

In this work, we have used three techniques: the Ensemble_prediction technique, which predicts class labels using a mode of member predictions, and it is calculated as the argmax of the summed score of each class label. The second method is a weighted averaging technique, where the prediction of each model is multiplied by the weight, and then their average is calculated, and the third is the hard majority voting, where it takes the majority votings of the three independent used classifiers models and gives the prediction of the classified class accordingly.

4. RESULTS AND DISCUSSION

In this section, we first introduce the utilized dataset and explain how class imbalance is handled. Then, we describe and discuss the prediction results of each of the implemented DL models. Finally, we compare the obtained results for each model with the classification accuracy of the ensemble model.

4.1 Dataset

The dataset we have utilized is "Credit Card," which has been gathered and analyzed by the Machine Learning Group of ULB during their work on fraud detection research. It is publicly available on the Kaggle repository [13]. This dataset includes data about credit card transactions made by European people in 2017. The dataset consists of 31 columns representing 28 features (V1, V2, ..., V28) produced from original data by principal components analysis (PCA) to protect people's credentials confidentiality in addition to three other attributes disclosed in their actual names. These attributes are:

- **Time:** This feature presents time by seconds between the current credit card transaction records with the first one.
- **Amount:** this attribute presents the amount of the transaction.
- **Class:** this attribute presents the classification of a transaction which is (1) if it is fraud or (0) otherwise.

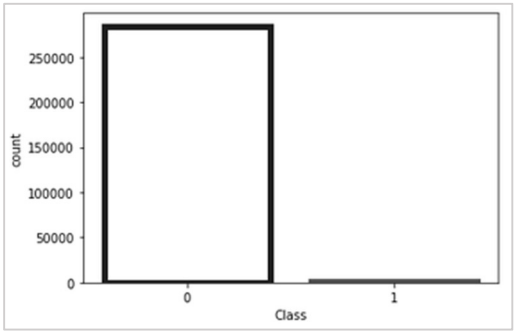

Figure 5: Number of fraud and normal transactions in the dataset

The dataset contains 284,807 normal transaction records against 492 fraud transactions. Thus, the dataset is noticeably imbalanced. As shown in Figure 5, the non-fraud class transaction records remarkably outnumber the fraud transactions. It is essential to balance the data before training the classification models. There are two general methods to handle data imbalance; these are over-sampling and under-sampling. In this work, we utilized the latter method to select a subset of normal samples equal to the number of fraud samples, that is, 492 samples. This method is chosen because it allows us to use different sets of training data and repeat the experiments to reduce the variance of the trained models.

After we gain a balanced number of positive and negative samples, the data are normalized and scaled so that the values of the data range from 0 to 1. This scaling step is essential to enhance the performance of the learning model during the training phase. Finally, we split the balanced dataset into two smaller subsets where 80% of the data was used to train the classification model, while 20% of the dataset was used to validate the classification model during the test phase.

4.2 Work Environment

We used Google Colab, also called "Google Collaboratory" which is a working environment that is a free web-based collaborative programming environment that provides an interactive and easy-to-use platform for implementing and building deep learning models to work on data science projects. We used the Colab environment with 12 GB RAM and 68 GB hardware disk. We wrote the code for building models in Python because of its compatibility with ML algorithms.

4.3 Evaluating CNN model

In the process of CNN modeling, we used the built-in Adaptive Moment Estimation (Adam) optimizer in TensorFlow to adjust the optimal parameters in this work. The use parameters are: learning rate = 0.001, beta_1 = 0.9, beta_2 = 0.999, epsilon = 1e-07, epochs = 50, validation_split = 0.2, dropout = 0.2, and activation = ReLU and sigmoid. The CNN model Loss and Accuracy curves are shown in Figure 5 and Figure 6, respectively. Figure 5 shows that the validation loss is lower than training loss, which is considered a good metric. The curve in Figure 6 shows that the validation accuracy is higher and better than the training accuracy, which means that the learned model is very well. Moreover, the confusion matrix of the CNN model is shown in **Error! Reference source not found.**

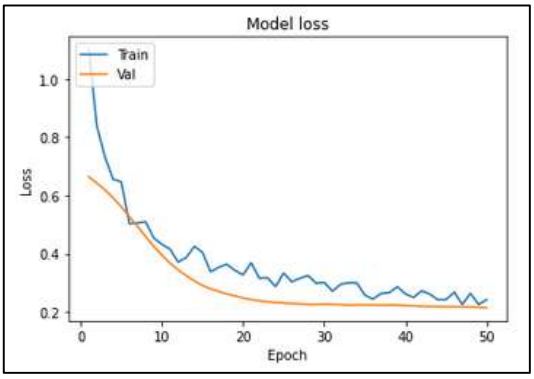

Figure 5: CNN model Loss curve with Max-Pool

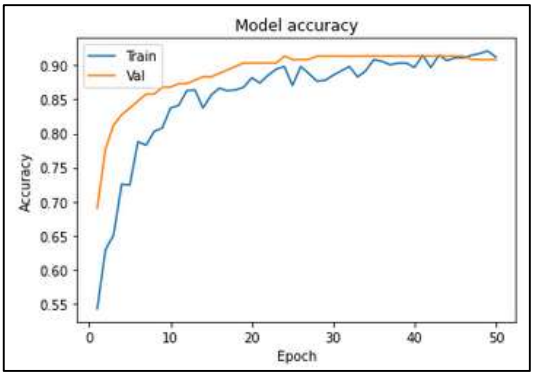

Figure 6: CNN model Accuracy curve with Max-Pool

Journal of Theoretical and Applied Information Technology
31st July 2022. Vol.100. No 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

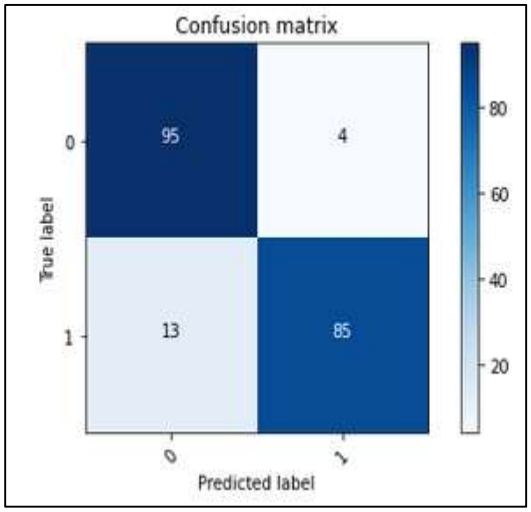

Figure 7: *Confusion matrix of CNN model*

4.4 Evaluating AE model

The parameters and inputs of the second model of Auto-Encoder are 100 training epochs, 256 batch sizes, and a 0.01 learning rate. The auto-encoder model loss curves are shown in Figure 8. We found that our threshold (cut_off) is 0.002. Then we selected 100 fraud samples and 100 normal samples and plotted them against the threshold. As shown in Figure 9, most fraudulent transactions have higher Mean Squared Errors (MSE) than normal transactions. Figure 9 shows that the model has learned the features of the normal transactions very well, and as we can see, all data under the threshold line are normal, which are being recognized by the model. On the other hand, all data represented over the line are fraudulent transactions that the model did not recognize since they are anomalies as they were unexpected.

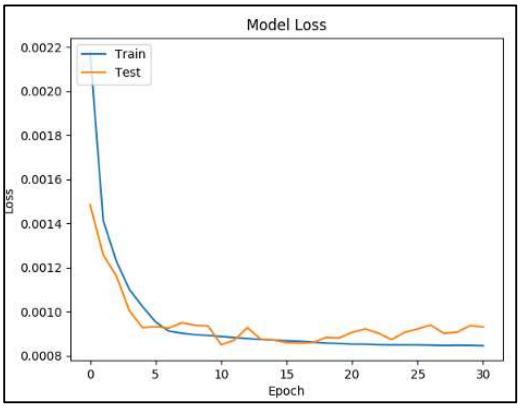

Figure 8: Auto-Encoder Model Loss

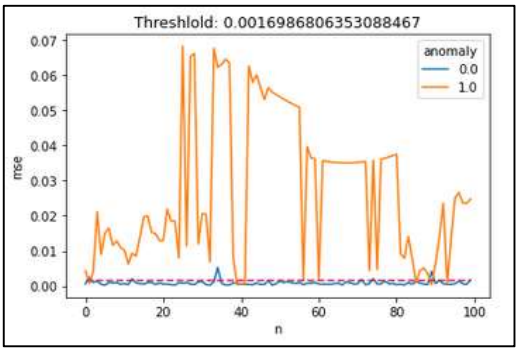

Figure 9. MSE for fraud and normal transactions

4.5 Evaluating RNN model

The third model, which is the RNN model, has some inputs and parameters that help to get better results, such as 100 training epochs, 20000 batch size, 0.01 learning rate, and relu, sigmoid activation functions. The RNN model Loss and Accuracy curves are shown in Figure 10 as follows:

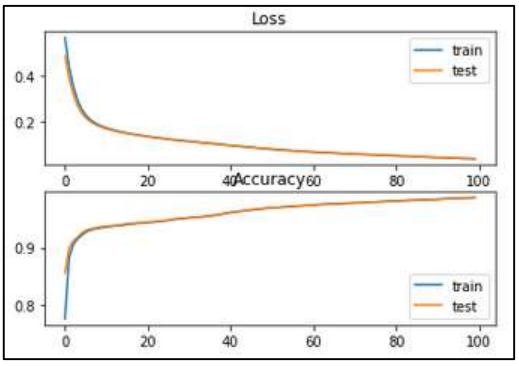

Figure 10: RNN Model Loss and Accuracy Curves

As we can see in Figure 10, the training Accuracy is almost the same as test Accuracy, and the training Loss is also nearly the same as test Loss. The Confusion Matrix of the RNN model is shown in Figure 11.

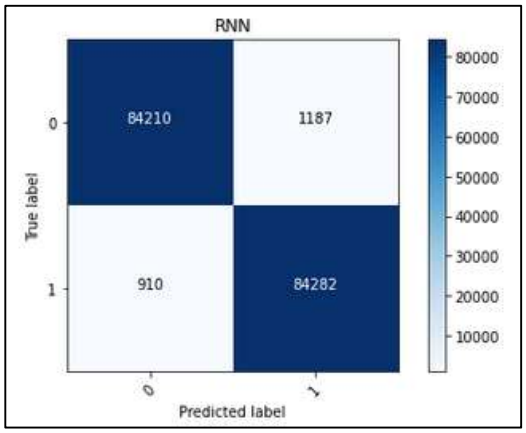

Figure 11: RNN confusion matrix

4.6 Evaluating the ensemble model

The methods we used are the Ensemble Prediction method, the Weighted Averaging method, and the Majority Voting Ensemble MVE method, where the last method (MV) gave us an accuracy result of 94.92% accuracy, which outperformed the accuracy obtained from the Auto-Encoder as a single classification model and improved the overall accuracy rate by 1.5% percent.

Table 5 compares the performance of the implemented DL models and the proposed MVE Ensemble model. As shown in the table, the validation accuracy of the ensemble model outperforms the accuracy achieved by each of the other three models.

Table 1: comparison of all models scores

Model	Accuracy	Recall	Precision
CNN	91.4	68.7	95.5
Auto-Encoder	93.4	-	-
RNN	91.8	98.9	98.6
Ensemble	94.9	83.8	97.05

5. CONCLUSION

In this paper, we have proposed a majority voting based ensemble technique that combines three deep learning algorithms, namely, convolutional neural network, auto-encoder, and recurrent neural network, for building a highly accurate classification model that would be used for credit card fraud detection and classification. These three learning algorithms were trained using an available public dataset called "Credit Fraud", which is dataset that is containing 284,807 normal transaction records and 492 fraud transactions. As a result, this dataset is considered unbalanced, and hence, we needed to handle this imbalance issue during the data preprocessing phase using random oversampling and random under-sampling in order to create a balanced dataset where normal and fraud transactions are of equal size. Other preprocessing steps have also been conducted on the dataset, such as features selection and extraction and values normalization.

After that, we started experimenting with the three deep learning algorithms CNN, Auto-Encoder, and RNN, and in each experimentation, we computed both values of Accuracy and Loss to evaluate the performance of each model being built, where CNN has given an accuracy value of 91.4, and a loss value of 0.3004, the Auto-Encoder model have given and accuracy value of 93.4 and a loss value of 8.5641e-04, while the RNN model has given an accuracy value of 91.8 and a loss value of 0.369. By observing these results, we can see that the AE classification model has given the best results so far, where the accuracy value is the highest among the three classification models, as well as the loss value is the lowest value being given. Finally, in the last step, where these three deep learning algorithms were combined as an ensemble algorithm to construct an ensemble deep learning model in order to give better results with higher performance. We used the Majority Voting Ensemble, which gave us an accuracy rate of 94.92%. This way, we managed to obtain a higher accuracy result than the one we obtained from Auto-Encoder as a single classification model, which contributed to improving the accuracy result by 1.5%.

For the study limitations that we have faced during working on this project, the first issue is that we had to work with handling the imbalance issue for the dataset before feeding the dataset as an input into the phase of building the classification model. Another issue is the long time being consumed during the training and learning process, as well as building the ensemble model, which is considered time and effort cumbersome. To conclude this paper, for future directions of this research, we will consider experimenting with more deep learning algorithms in order to come up with models that

would give the most possible height accuracy rates and the least possible loss rates, as well as create more ensemble algorithms that would give the best-optimized results in credit card fraudulent financial transactions.

ACKNOWLEDGEMENT

The authors would like to thank the Deanship of Graduate Studies at Jouf University for funding and supporting this research through the initiative of DGS, Graduate Students Research Support (GSR) at Jouf University, Saudi Arabia.

REFERENCES

[1] Y. Lu, Deep neural networks and fraud detection, Department of Mathematics - Uppsala University, 2017.

[2] D. Choi and K. Lee, "An Artificial Intelligence Approach to Financial Fraud Detection under IoT Environment: A Survey and Implementation," *Security and Communication Networks,* 2018.

[3] S. Sanober, I. Alam, S. Pande, F. Arslan, K. P. Rane, B. K. Singh, A. Khamparia and M. Shabaz, "An Enhanced Secure Deep Learning Algorithm for Fraud Detection in Wireless Communication," *Wireless Communications and Mobile Computing,* 2021.

[4] E. Kim, J. Lee, H. Shin, H. Yang, S. Cho, S.-k. Nam, Y. Song, J.-a. Yoon and J.-i. Kim, "Champion-challenger analysis for credit card fraud detection: Hybrid ensemble and deep learning," *Expert Systems With Applications,* vol. 128, pp. 214 - 224, 2019.

[5] D. Choi and K. Lee, "An Artificial Intelligence Approach to Financial Fraud Detection under IoT Environment: A Survey and Implementation," *Security and Communication Networks,* 2018.

[6] Y. Abakarim, M. Lahby and A. Attioui, "An Efficient Real Time Model For Credit Card Fraud Detection Based On Deep Learning," in *International Conference on Intelligent Systems: Theories and Applications*, Morocco, 2018.

[7] A. Roy, J. Sun, R. Mahoney, L. Alonzi, S. Adams and P. Beling, "Deep Learning Detecting Fraud in Credit Card Transactions," in *Systems and Information Engineering Design Symposium (SIEDS)*, Charlottesville, VA, USA, 2018.

[8] A. M. Mubalaike and E. Adali, "Deep Learning Approach for Intelligent Financial Fraud Detection System," in *3rd International Conference on Computer Science and Engineering (UBMK)*, Sarajevo, Bosnia and Herzegovina, 2018.

[9] R. Zhang, F. Zheng and W. Min, "Sequential Behavioral Data Processing Using Deep Learning and the Markov Transition Field in Online Fraud Detection," 2018.

[10] X. Zhang, Y. Han, W. Xu and Q. Wang, "HOBA: A novel feature engineering methodology for credit card fraud detection with a deep learning architecture," *Information Sciences,* 2019.

[11] H. Ye, L. Xiang and Y. Gan, "Detecting Financial Statement Fraud Using Random Forest with SMOTE," 2019.

[12] J. I.-Z. Chen and Kong-Long Lai, "Deep Convolution Neural Network Model for Credit-Card Fraud Detection and Alert," *Journal of Artificial Intelligence and Capsule Networks,* vol. 03, no. 02, pp. 101-112, 2021.

[13] M. L. G. -. ULB, "Credit Card Fraud Detection - Anonymized credit card transactions labeled as fraudulent or genuine," Kaggle, [Online]. Available: https://www.kaggle.com/mlg-ulb/creditcardfraud. [Accessed 01 2022].

[14] A. Saini and S. D. Sarkar, "Credit Card Fraud Detection using Machine," *International Journal of Engineering Research & Technology (IJERT),* 2019.

[15] B. Rocca, "Handling imbalanced datasets in machine learning," 28 Jan 2019. [Online]. Available: https://towardsdatascience.com/handling-imbalanced-datasets-in-machine-learning-7a0e84220f28.

[16] manav_m, "Introduction to Convolutional Neural Networks CNN," 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/.

[17] Z. Zhang, X. Zhou and X. Zhang, "A Model Based on Convolutional Neural Network for Online Transaction Fraud Detection," *Open Access,* 2018.

[18] Catak, F. Ozgura, Mustacoglu and A. Fatihb, "Distributed denial of service attack detection using autoencoder and deep neural networks," *Journal of Intelligent & Fuzzy Systems,* 2019.

[19] M. A. Al-Shabi, "Credit Card Fraud Detection Using Autoencoder Model in Unbalanced Datasets," *Journal of Advances in Mathematics and Computer Science,* 2019.

[20] J. Forough and S. Momtazi, "Ensemble of Deep Sequential Models for Credit Card Fraud Detection," *Applied Soft Computing,* 2020.

[21] I. Benchaji, S. Douzi and B. E. Ouahidi, "Credit Card Fraud Detection Model Based on LSTM Recurrent Neural Networks," *Journal of Advances in Information Technology,* 2021.