Volume 39(1), pp. 1–44
http://orion.journals.ac.za

ORiON
ISSN 0259–191X (print)
ISSN 2224–0004 (online)
©2023

Anomaly detection using autoencoders with network analysis features

Richard Ball[*], Hennie Krüger[*], and Lynette Drevin[*]

Received: 21 February 2022; Revised: 7 July 2022; Accepted: 7 July 2022

Abstract

Fraudulent activity within a financial ecosystem often involves the coordinated efforts of several bad actors. Expressing the interactions between participants in a system as a mathematical graph allows researchers to apply social network analysis to understand the nature of these relationships better. This article proposes and extends a unified approach using an autoencoder to detect anomalies in a transactional setting. The methodology begins with a neural architecture search to determine the best autoencoder model architecture configuration. This is followed by a threshold optimisation process to find a reconstruction error that best discriminates between non-anomalous and anomalous classes. Gaussian scaling is applied to the raw anomaly scores in order to represent the output in an interpretable and universally transferable form. The unified approach is extended by selecting and including network metrics as features, for the purpose of producing a model that can detect anomalies from both standard transactional data and network data representing the relationships between users within a financial system. Applying SHAP on the model output highlighted the strongest contributing or offsetting network metric features for all anomalies detected. The PageRank and degree centrality network metrics were most important in detecting anomalous instances within the data. Including network metrics in the feature space generated encouraging model performance results, leading to a potential low operational cost of fraud.

Key words: Anomaly detection; neural networks; autoencoders; neural architecture search; social network analysis; Shapley values.

1 Introduction

As the financial landscape becomes more competitive through globalisation and the mass adoption of technology [100], the number of financial transactions being executed via digital channels is increasing. Together with benefits such as efficiency and profitability

[*]North-West University, Potchefstroom, South Africa, email: rballe33@hotmail.com

http://dx.doi.org/10.5784/39-1-711

[32], this also brings more opportunities for fraudulent actors to take advantage of these channels, hence increasing the volume and complexity of the fraud, extortion and money laundering being conducted in the industry [78]. Due to the increase in suspicious activities, many studies have been conducted to try and find appropriate ways of combatting the nefarious use of digital financial channels (see [5], [43] and [90]). Despite these studies, detecting fraudulent transactions remains a non-trivial exercise and has become a common challenge for many organisations. One of the most popular approaches used for detecting anomalous transactions is the use of anomaly detection techniques. An anomaly can be seen as data that deviates substantially from the norm [67]. There are no universally accepted criteria to define anomaly detection as different mechanisms in different systems may be the cause of an anomaly (see [87] and [111]). A general description, which will be accepted for this study is that anomaly detection is the process of identifying rare observations which differ substantially from the majority of the data from where they are drawn. Various anomaly detection techniques have been proposed in the literature, and examples include the use of outlier techniques [47], isolation forests (see [56] and [62]), and a one-class support vector machine [105] algorithms. Other examples can be found in [61], [106] and [111].

Another popular approach for anomaly detection that has gained momentum recently is the use of autoencoders (see [70], [79] and [93]). An autoencoder, which is classified as an unsupervised learning technique, offers a different approach to anomaly detection. The technique aims to learn some low-dimensional feature representation space on which the given data instances can be well reconstructed [93]. In a fraud detection context, an autoencoder can be trained to learn to reproduce non-fraudulent transactions only, due to the fraudulent transactions being removed from the training set. When fraudulent transactions are introduced, the autoencoder's reconstruction performance worsens, and this deteriorated performance is then used as the method whereby anomalies are detected. Another characteristic of autoencoders that makes them suitable for anomaly detection is that they are mainly applied to tabular data, which is typically the format of financial transactions. However, attempts have been made to apply them to graphs [64] and sequence data [33].

Financial transactions in a system typically consist of technical details such as client details and specifics about the transaction such as the amount. A drawback of these financial transactions is that they exclude any information concerning the underlying relationship that may exist among clients, devices, types of transactions *etc.* Traditional methods of constructing the handcrafted features required for tabular or transactional data may in specific scenarios be less expressive of the underlying phenomena making it again much more complex for fraud investigations. Therefore, network data approaches have been shown to improve model performance by learning a more expressive representation of the problem [64]. For example, when analysing the behaviour of their customers, organisations frequently resort to understanding how users of a system are connected to one another. These connections are particularly useful when attempting to detect suspicious activity. An example of performing such an analysis can be found in cases where financial crime investigators use social network analysis (SNA) to represent the interconnectedness of multiple users within a system [17].

In this paper, it will be argued that, given the importance of the underlying relationships among transactions in unsupervised anomaly detection, the use of certain basic network metrics in SNA can assist in detecting anomalies by providing metrics that aim to measure the influence that nodes or users have within a transactional system. This includes understanding how the nodes are connected by edges that describe the relationship between users. Having this information available will then improve the accuracy of anomaly detection systems.

The choice of SNA as an additional tool to enhance the unsupervised anomaly detection process offers several advantages. As mentioned, modern organisations are typically networked, meaning that the relationships between their employees, customers, and suppliers can in some sense be linked using data. Examples of data that could link users within a transactional system include shared IP addresses, credit cards, and digital devices. Fraud can naturally occur in any part of the network, and SNA can therefore be used to reveal the connections and behaviours that would be difficult to detect through raw data alone [17]. Furthermore, SNA also proves to be beneficial due to the dynamic nature of fraudulent activity. Incorporating network analysis techniques into the fraud detection process helps to augment the performance of human-in-the-loop analysts who typically work in conjunction with static rule-based systems. It is expected that SNA data will provide additional insights that these rule-based systems fail to detect. The incorporation of social network metrics will also provide an alternative viewpoint that highlights connections in a broader network, thereby providing a more comprehensive view of each fraud investigation in the context of a more extensive network. Advantages of a more technical nature include the visual and mathematical capabilities of SNA and the relatively easy translation of data into matrix form, allowing for the calculation of various network metrics that aim to uncover the theoretical properties of the graph [63]. This would further help to characterise the network as a whole and assist with conducting queries of the social network graph data.

While SNA is an established technique for identifying fraudulent behaviour in financial networks, using it for visualisation purposes alone has proved challenging to work with as the size and complexity of the network increases [17]. The network metrics derived from SNA can instead act as features that are used to train a machine learning model able to detect anomalies in a transactional setting. This approach removes the burden on financial investigators having to struggle to interpret large network visualisations and placing less emphasis on them having to explicitly interpret the importance of each of the possible network metrics that typical SNA platforms calculate.

Based on the brief preceding motivational introduction, this paper proposes an approach that combines the strengths of using an autoencoder for unsupervised learning, with selected metrics offered by SNA to perform anomaly detection. This involves using both standard transactional data and network metrics to formulate features used to train a model capable of detecting anomalies in a real-world context. In order to assess the performance of the proposed method, traditional classification metrics suited for fraud detection models and Shapley values are considered [3]. The recommended method implements a previously proposed unified approach to anomaly detection [8], where multiple candidate autoencoder architectures were simulated via a neural architecture search.

The remainder of this article is organised as follows: Section 2 provides some literature background on the use of SNA in an anomaly detection context. Section 3 presents the industry data set used for experimental research. A proposed unified approach is then introduced and applied in Section 4, including the use of network metrics as features. The effectiveness of the approach is evaluated using both Shapley values and standard classification performance metrics, with the results being discussed in Section 5. Finally, the conclusion of the study is presented in Section 6.

2 Related work

Autoencoders and SNA are two different approaches for modelling relationships in data. While the former is frequently used for detecting anomalies in data, SNA can be used to represent the importance of participants in a graph structure. To contextualise the proposed methodology, a cursory overview of examples of existing work related to autoencoders and SNA is presented. Although there are many examples of these concepts in the literature, this study will highlight only a few key examples to put the topics into context and show the wide variety of applications for the approaches. Furthermore, the references aim to provide an understanding of previous attempts to combine both autoencoders and SNA in order to produce a more effective anomaly detection solution.

2.1 Autoencoders for anomaly detection

Various autoencoders have been developed to detect anomalies in different application areas (see [2], [20] and [24]). These can be categorised according to the different architectural variations of the autoencoder, where examples from the literature for some of the categories are presented below.

Schreyer *et al.* [24], used a *sparse autoencoder* to detect anomalies in large-scale accounting data. Nine distinct architectures were evaluated and the approach produced reconstruction errors that could be used for a highly adaptive anomaly assessment of journal entry data. In another study, Liang *et al.* [58] demonstrated applying a sparse autoencoder to detect pump fault anomalies. Classical principal component analysis is compared with the autoencoder, where the latter is shown to perform better at detecting anomalies and isolate abnormal features. An ensemble approach is proposed by Narayana *et al.* [73], where a sparse autoencoder is applied to detect anomalous attacks in an intrusion detection context. Sparse autoencoders are regularly applied in different domains, and additional examples of the application for anomaly detection can be found in [69], [85] and [101].

In the category of *variational autoencoders*, Zavrak and Iskefiyeli [109] showed that these types of architectures can be used to detect anomalies in network traffic. A semi-supervised approach is utilised to detect network intrusions using flow-based data, and the experimental results illustrated that this approach outperformed a one-class support vector machine baseline model. Variational autoencoders were also found to be helpful in detecting anomalies in multidimensional time-series data [39]. The authors proposed an approach where a Gaussian mixture model was employed to allow for a series of input distributions. Further anomaly detection applications of variational autoencoders are discussed in [13], [22] and [57].

Denoising autoencoders, another category of autoencoders, also form the basis of many studies. In the area of network anomaly detection, Mohamed *et al.* [71] trained a denoising autoencoder to detect network anomalies as part of an intrusion detection system. They found that the model forced the extraction of intrinsic features, which increased the detection accuracy of the intrusion detection system. Another application of denoising autoencoders is reported by [22]. In their study, they achieved success in using stacked denoising autoencoders for detecting anomalies in wind turbine data, where the model was trained on standard operating data, using the reconstruction error as a monitoring indicator.

A popular architecture that is often used in autoencoders is a *convolutional* structure. Some examples to illustrate the use of this architecture include the following. In a study by Chow *et al.* [26] a convolutional autoencoder was leveraged to detect defects (anomalies) in concrete structures. The model was trained in an unsupervised manner, saving the authors from conducting extensive labelling of the data. Superior precision and recall scores were reported in comparison with baseline segmentation methods. Another research project by Wang *et al.* [104] illustrated that a convolutional autoencoder could be used for hyperspectral anomaly detection, with the intention of detecting observations that differ substantially from their surroundings. Convolutional structures also cater to problems that contain a time interval element. For example, Cui *et al.* [28] proposed a convolutional autoencoder to detect system failures from console log data. To capture time dependencies in the logs data, a 2D matrix was constructed to reveal more informative system behaviours, ultimately aiding in determining system failures over time.

The final architecture for which examples from the literature are given is *Long Short-Term Memory (LSTM)*. In a study by Nguyen *et al.* [77], the LSTM architecture was demonstrated to be an effective technique for learning long-term dependencies and for representing the relationship between current and previous events. It was shown how LSTM can be included in an autoencoder, for the purposes of detecting anomalies in the supply chain management industry. In another example, Said Elsayed *et al.* [91] also employed an LSTM-based autoencoder to detect malicious attacks in network data. The actual anomaly detection classifications were made by a one-class support vector machine algorithm, with the LSTM autoencoder being used to compress the input network data into latent features. Other examples of studies employing LSTM for anomaly detection can be found in [21] and [44].

2.2 Social network analysis

SNA has been applied to the problem of fraud and anomaly detection in many ways (see [18], [54] and [84]). Some of these applications fall into the categories of visual analysis, network metrics, feature extraction, and SNA for fraud detection.

Visual analysis is a qualitative application of SNA, which according to Decuypere [31], involves collecting and coding relational data, visualising network diagrams, analysing the form of network diagrams, and finally interpreting the form of these network diagrams. The study claims that using SNA for visualisation purposes offers the possibility to scrutinise and visualise networks without necessarily inferring causal relationships to the underlying network structures. The benefits of visual analysis were also displayed in the *PeakVis* network visualisation tool developed by Sanvido *et al.* [97], which analyses social media data, allowing users to identify relevant segments and grasp the subjects being discussed in real-time. Kucher *et al.* [52] agree that SNA can be used for the visual analysis of social and text-based data. Their research project highlighted that besides text data visualisation, SNA can also be used for network data visualisation and exploratory analysis. Additional examples of SNA for visual analysis applications can be found in [14] and [50].

Another important application of SNA is the computation of *network metrics*. Centrality metrics are network metrics that are employed as a means to identify essential elements by determining their behaviour and roles within a complex network [38]. According to Zhang and Luo [110], social networks focus on the relation of participants to each other within the network, with network metrics providing a method for surfacing the influence that participants have on the network as a whole. Their findings indicate that degree centrality, betweenness centrality, and closeness centrality are among the most critical network metrics in SNA. Another study by Riquelme *et al.* [88] considered the impact that centrality metrics have on linear threshold models, with results confirming that network metrics were important in ranking actors and networks in a distinguishable way. Network metrics have also shown promise in psychological social networks [16]. The authors state that the three main centrality metrics mentioned previously should be augmented with additional, more relevant centrality metric approaches when dealing with psychological social network data. Suggested examples of more relevant centrality metrics include neighbour-inclusion and relative ranking-related metrics.

Zandian and Keyvanpour [108] indicated that SNA could be used as a *feature extraction* method, whereby network features derived from SNA were combined with other transactional features to assist in detecting fraud in banking accounts. The authors claim that many fraud detection solutions do not consider both user-level and network-level features simultaneously, and having a method that is able to learn from both kinds of features would increase the accuracy of their overall fraud detection approach. A similar approach was adopted by Beigi *et al.* [9], who used linked network data to engineer features to build a signed link analysis model. Due to the sparse nature of the linked network data, other sources of information are included as features to increase model performance. Another example of using SNA for feature extraction is presented by Kumari *et al.* [53]. In this research, data representing the topological structure of the network is used in conjunction with content-based information to build a model for link prediction. Finally, Oskarsdottir *et al.* [81] proposed implementing SNA in the insurance sector, where a network was created by linking claims data to all involved parties. Features from the network were extracted and combined with claim-specific transactional features to train a supervised learning model. This approach yielded results indicating that the combination of network and claim-specific features improved the performance of supervised learning models.

SNA is also applied extensively in the *fraud detection* domain, where users who transact with one another are modelled as a network, with nodes and edges representing users and the transactions between them respectively. Colladon and Remondi [27] proposed using SNA to prevent money laundering activities, where network metrics were explicitly created to model clients' risk profiles within the network. Interesting to note that the authors suggest the use of a hybrid approach, with SNA being used to determine the risk profiles through network metrics and create a visual analysis of the network to gain a broader understanding of suspicious behaviour. Pourhabibi *et al.* [84] highlighted many of the key graph-based fraud and anomaly detection approaches. The review argued that social networks are typically used for anomaly detection in a two-step approach; namely for selecting and calculating network features and classifying observations from the feature space determined via the first step. Other researchers [55] argue that visual analysis is also an essential tool for fraud detection, including examples of the technique being applied across multiple industries such as telecommunications, stock exchanges, insurance, and banking. Visual analysis is then considered in parallel to more analytical network metrics approaches, with the two seen as influencing each other during pre-processing, post-processing, integrated, and pure visualisation processes.

The preceding examples from the literature clearly indicate the importance and prominence of employing different architectures of autoencoders for anomaly detection. From the studies quoted, it is also clear that other existing techniques such as SNA can be used to evaluate and analyse data for fraudulent activities and that this technique is often used to complement other anomaly detection methodologies.

3 Modelling prerequisites

In this section, a brief review of the concepts that are necessary to construct the network features for unsupervised anomaly detection is presented. The relevant modelling concepts include autoencoders for anomaly detection, the previously proposed unified approach for anomaly detection [8], and SNA.

3.1 Modelling basics of an autoencoder

Autoencoders are neural networks that can be used to detect anomalies in an unsupervised or semi-supervised manner. According to Goodfellow *et al.* [37], autoencoders are trained to reconstruct its input to its output, an example of which is provided in Fig. 1. The encoder in Fig. 1 is responsible for compressing the feature space down into a lower dimension, as determined by the bottleneck layer. The compressed data are then mapped back into a reconstructed vector by the decoder function, where the encoder and decoder are typically symmetrical architectures, each containing multiple neuron layers.

The main components of an autoencoder can be explained as follows [98]. The model is trained using a set of N aggregated records

$$X = \{x^1, x^2, \ldots, x^n\}$$

where each of the aggregated records consists of a tuple of k transactional attributes

$$x^i = (x^i_1, x^i_2, \ldots, x^i_j, \ldots, x^i_k)$$

x^i_j representing the j^{th} transactional attribute of the i^{th} aggregated record.

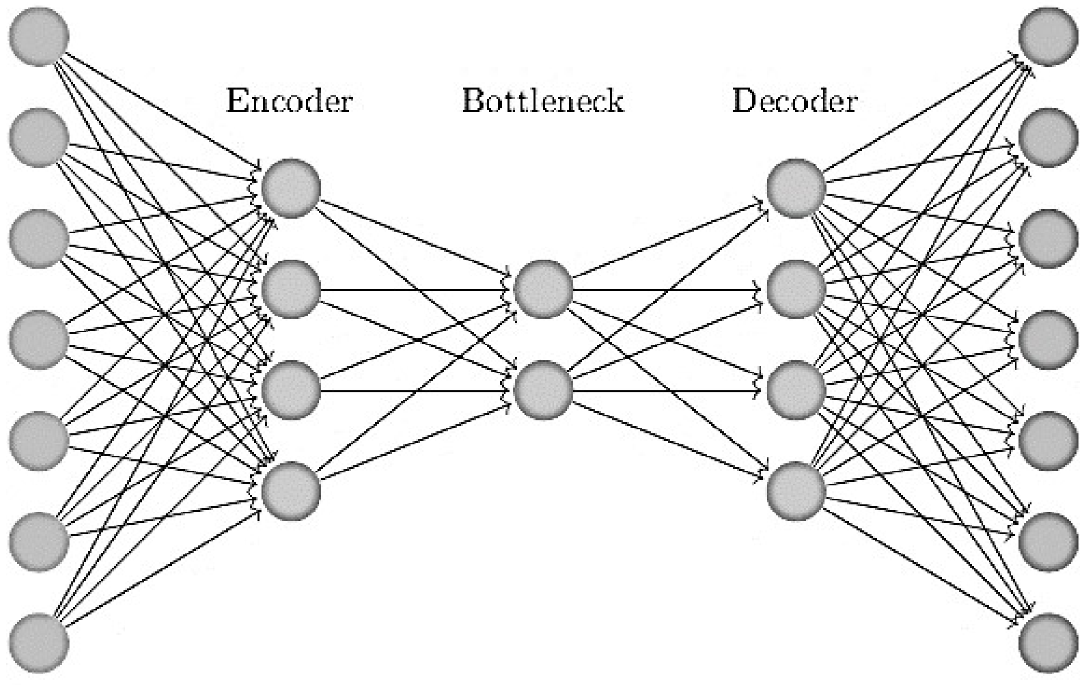

Figure 1: *An example of the architecture of an autoencoder.*

The network contains a hidden layer h which generates a function that describes the input of the model. The model architecture also includes an encoder function $h = f_\theta(x)$ and a decoder function $r = g_{\theta(h)}$, which both assist as nonlinear mappings. More specifically, the encoder function maps an input vector x^i into a compressed representation z^i within the latent space Z. The compressed representation is then mapped back onto a reconstructed vector $\hat{x}^i$ by the decoder function, with the reconstruction of the original input data being represented by $\hat{x}^i$. While not always the case, the encoder and decoder are typically symmetrical in nature, where deep autoencoders are those considered to contain a number of hidden layers within the encoder and decoder.

The encoder can be defined as

$$f_\theta^l(\cdot) = \sigma^l \left(W^l \left(f_\theta^{l-1}(\cdot) \right) + b^l \right)$$

while the decoder mapper is defined as

$$g_\theta^l(\cdot) = \sigma'^l \left(W'^l \left(g_\theta^{l-1}(\cdot) \right) + d^l \right)$$

where σ and σ' denote the nonlinear activations, which often takes the form of the rectified linear unit (ReLU) activation function. W, b and d represent the model weights and biases respectively, while θ represents the parameters of the model. The number of hidden layers in the model are described by l.

Training the autoencoder entails attempting to achieve an accurate representation of the input data, with the model learning parameters θ that minimise the reconstruction error between the input and the reconstructed output. The training objective therefore optimises

$$\underset{\theta}{\operatorname{argmin}} ||X - g_\theta(f_\theta(X))||.$$

The approach is well suited for anomaly detection, where the autoencoder is used to learn the most salient features in a data set, with the intention of reproducing these salient features. The output, namely the reconstructed features, is the model's attempt at producing a reconstruction of the input data based on the data that was used to train it. If the input data were similar to the data used to train the model, then the output would be similar to the input, namely due to the model having learnt an effective input mapping function via the training process. This is relevant for anomaly detection purposes because input data that is dissimilar from the data used to train an autoencoder will produce an inferior reconstruction. The quality of the reconstruction can be measured by means of the mean squared error (MSE), which calculates the mean of the squares of the differences between the reconstruction and its input. The MSE is known as the reconstruction error in this context, where a higher reconstruction error is achieved in instances where the reconstruction deviates substantially from the input data. Conversely, a lower reconstruction error is achieved in cases where the reconstruction is more similar to the input data.

This approach of assessing the quality of the reconstruction using the MSE is used to detect input data that differs substantially from the data that was used to train the autoencoder. An autoencoder can therefore be trained using non-anomalous data only. As soon as anomalous data are introduced as the input, the MSE would determine that the mapping function is not familiar with the data, thereby producing a high reconstruction error. This has powerful implications for fraud detection, where an autoencoder can be trained on non-anomalous transactional data only, for the purposes of detecting anomalous transactions. Of particular importance is the fact that, by definition, anomalous data occurs less frequently.

3.2 A unified approach

One of the frequent problems in implementing autoencoders, and neural networks in general, is determining an optimal model architecture. No single solution is guaranteed to deliver an optimal architecture, which is important because architecture design considerations substantially impact model results. Traditionally, researchers used a grid search method where a large number of hyperparameter combinations are explored and the best one selected [95]. There are also other suggestions in the literature ([82]; [86]) on how to construct autoencoder architectures.

In this study, it was decided to use a unified approach previously proposed by Ball *et al.* [8] to ensure that the architecture of the chosen autoencoder is both derived using a structured methodology, and is optimal from a model performance perspective. The unified approach uses a combination of performance metrics proven to be well suited for anomaly detection problems and extensive experimentation conducted in [8], produced satisfactory and highly accurate results. Other benefits of the approach include the absence of having to make subjective visual inspections of the model threshold and having an interpretable model output that is transferable across business domains.

The approach is comprised of three main phases that are performed sequentially.

- A neural architecture search is performed by simulating multiple autoencoder architectures;
- A threshold optimisation algorithm is applied to the simulated architectures to assist with determining optimal fraudulent binary classification outcomes; and
- A Gaussian scaling process is implemented to prohibit any subjective interpretation of the raw anomaly scores.

In the remainder of this section, the algorithms for the three main phases of the proposed approach will be highlighted, while the details as well as the initial experimental results can be found in [8].

Neural architecture search: Algorithm 1 presents the first phase of the approach that is employed to simulate a population of candidate autoencoder architectures. The architectures themselves are created by assigning them randomly generated depths and numbers of nodes. An upper bound is applied to the size of the input layer of each architecture, with each architecture operating under the same regularisation constraint. A balanced metric, comprised of the average of the area under the receiver operating characteristic (ROC) curve (AUC), average precision (AP), and normalised Matthews correlation coefficient (MCC), is derived in order to select the best performing architecture. ROC is a curve that plots the true positive rate against the false positive rate of a binary classifier, to determine the classification performance across different thresholds. The AUC [48], AP [42] and MCC [25] have all been used effectively as a classification performance metric for problems with a class imbalance. Combining these into an average or balanced score helps normalise the different magnitudes of each performance metric. This is especially important for the AP metric as the baseline for the AP is the fraction of positive classes [92]. In a typical anomaly detection problem, this could lead to an AP baseline of 1% or lower depending on the severity of the imbalance. The AP baseline is in contrast to the AUC metric which has a baseline of 50%.

Threshold optimisation: The second phase of the approach entails further evaluating the MCC component of the balanced score obtained in Algorithm 1. The MCC requires that the output of the autoencoder architecture be presented in binary form, where the class labels come from two distinct groups, namely positive and negative classes. This is not necessarily the case for AUC and AP, which can be calculated directly from the raw anomaly score produced by the autoencoder. In order to represent the raw scores in binary form, a threshold is required to split the scores into a binary fraudulent classification outcome. Determining a threshold is subjective, where an optimal split between the non-anomalous and anomalous classes are sought. Traditionally, the maximum reconstruction error achieved on the training set was used as the threshold for classifying anomalies [30]. Other approaches suggest using the proportion of anomalous data in the training data as a heuristic for threshold determination [49]. Neither of these approaches considers classification performance metrics as part of the thresholding decision, instead they evaluate the performance of their thresholds using classification metrics after the fact. A formalised

Algorithm 1 Neural architecture search algorithm

Input: number of n training epochs, intended population size m of generated architectures, size of the input layer l

Output: An array $S = s_1, \ldots, s_m$ containing the balanced scores of the m simulated architectures.

```
1: for $i = 1$ to $m$ do
2:     generate random depth $d$;
3:     generate random architecture $A$, with a node upper bound $l$ and length $d$;
4:     for $j = 1$ to $d$ do
5:         if $A[j] < min$ then
6:             $min \leftarrow A[j]$;
7:         end
8:     end
9:     remove min from $A$;
10:    encArchitecture $\leftarrow A$ sorted in descending order;
11:    decArchitecture $\leftarrow A$ sorted in ascending order;
12:    build input layer of size $l$;
13:    for $k = 1$ to $d - 1$ do
14:        build encoder layer with encArchitecture[$k$] nodes;
15:    end
16:    build bottleneck with min nodes;
17:    for $n = 1$ to $d - 1$ do
18:        build encoder layer with decArchitecture[$n$] nodes;
19:    end
20:    build output layer of size $l$;
21:    train the autoencoder;
22:    calculate the balanced score $s_i$ using test data;
23:    $S \leftarrow s_i$;
24: end
25: output S;
```

method for determining an optimal threshold is introduced as Algorithm 2, where the balanced score is incorporated as part of the thresholding decision. More specifically, a range of possible threshold values is simulated to determine the impact that each threshold has on the balanced score. The threshold that achieves the highest overall balanced score is chosen as the optimal threshold.

Algorithm 2 Anomaly score threshold optimisation algorithm

Input: $Y = \{y_1, \ldots, y_m\}$ is a linearly spaced vector, starting at zero and extending to a value of n, containing m samples.

Output: An optimal threshold value $Y[index]$.

```
1:  for $i = 1$ to $m$ do
2:      read $y_i$ from $Y$;
3:      calculate the binary classification outcomes for $y_1$;
4:      calculate the $MCC$, $AUC$ and $AP$ scores from the binary classification outcomes;
5:      populate array $R \leftarrow r_i$ // the balanced score computed
6:  end
7:  $max \leftarrow R[0]$;
8:  for $i = 1$ to $m$ do
9:      if $R[i] > max$ then
10:         $max \leftarrow R[i]$;
11:         $index \leftarrow i$;
12:     end
13: end
14: output $Y[index]$;
```

Gaussian scaling: The final phase in the unified approach is scaling the raw scores to prevent subjective interpretation. The optimal architecture can be discovered after implementing Algorithms 1 and 2, with the output of the autoencoder model taking the form of a reconstruction error. This error is the MSE between the actual input data and the reconstructed input data that the autoencoder produced. The reconstruction errors are the raw anomaly scores, with a higher score indicating that the underlying transaction has a high likelihood of being anomalous. Interpreting raw anomaly scores is a subjective exercise, which presents an opportunity to normalise the scores into an easy to interpret range of $[0; 1]$. A simple approach entails applying linear scaling to the scores, although this approach does not capture the separation between the classes as required. Alternatively, a Gaussian scaling approach can be applied, as presented in Algorithm 3.

Kriegel *et al.* [51] found that the Gaussian technique was a good general choice for transforming raw anomaly scores from a wide variety of anomaly detection algorithms. Applying Algorithm 3 resulted in a far superior separation between the non-anomalous and anomalous classes, with the majority of the non-anomalous class residing in the lowest score tier and the majority of the anomalous class falling within the higher, 90%+ score tier. Having a distribution of scores that clearly discriminates between the two underlying classes is useful in an operational context, where fraud detection practitioners can easily discern the likelihood of a transaction being anomalous based on its probability score.

Algorithm 3 Gaussian scaling of anomaly score algorithm

Input: $S = s_1, \ldots, s_m$ is an array of anomaly scores, containing m samples.

Output: An array $U = u_1, \ldots, u_m$ containing the associated unified scores for m samples.

```
1: for one pass of array $S$ do
2:     calculate $\mu_S$; // mean
3:     calculate $\sigma_S$; // standard deviation
4:     $cdf = \frac{S(0)-\mu_S}{\sigma_{S^*}\sqrt{2}}$; // calculate the cumulative distribution function of $S$
5:     $erf(cdf)$; // transform the cdf with the Gaussian error function
6:     normalise the transformation to the range $[0;1]$;
7: output $U$;
```

By combining a neural architecture search, threshold optimisation, and Gaussian scaling, the unified approach provides practitioners with a structured methodology to create autoencoder models for anomaly detection purposes. The output of the process is a trained autoencoder model, with the threshold calibrated according to performance metrics relevant for problems with imbalanced classes.

3.3 Basic concepts of social network analysis

Social networks refer to the structure of social actors that share interests of activities [96]. O'Malley and Onnela [80] also describe SNA as the understanding of the interdependencies in behaviour related to configurations of social relations. Both the relationships between individuals and their attributes can be considered the observations of interest within a social network. The observations within a social network are thought to be dependent in part on other observations within the network, due to their interconnected nature. Therefore, the techniques used to analyse the dependent data in social networks differs somewhat from the traditional statistical methods used to model independent observations.

In SNA, a graph or network is a finite set of n nodes or vertices that are connected by a finite set of m edges. In a financial services context, these nodes could be considered users of a particular system, and the edges would represent the connections or relationships between the users. A graph is typically referred to as G and is formally expressed as $G = (V, E)$. V is a finite set, called the vertices of G, while E is a set of two-element subsets of V, called the edges of G. The edges E are unordered pairs of vertices [10]. Fig. 2 illustrates an example of a simple graph. The graph has five nodes as represented by the circles and six edges as represented by the pairs of two letters highlighting the relationships between the nodes. To give an example explaining how Fig. 2 works, B and C represent nodes, whereas BC is the line or edge that connects these two nodes.

The edges between nodes within a graph can also have a directional element to them, where the graph is referred to as a directed graph. A directed graph (or digraph) places importance on the direction of the relationship between the nodes. An example of a directed graph is displayed in Fig. 3. The direction of the arrow between nodes A and B, for example, confirms that the edge flows in the direction from node A to node B. Directed graphs can also have edges that flow in both directions as well as edges that flow from the starting node back to itself (self-loops).

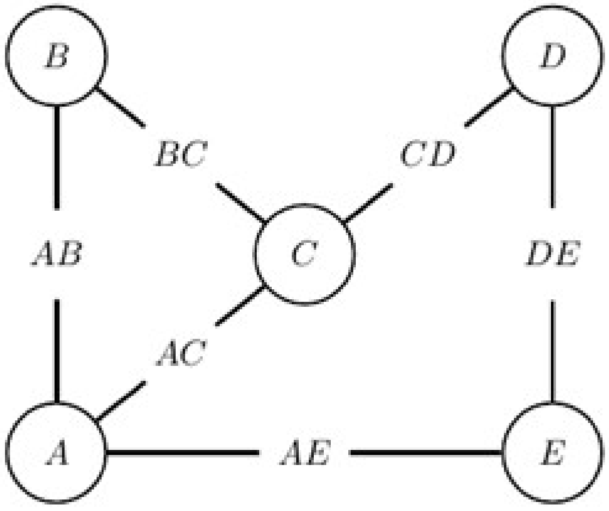

Figure 2: *An example of a simple undirected graph*

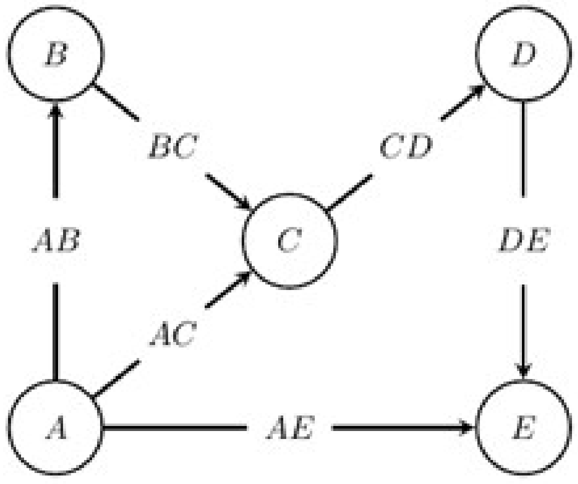

Figure 3: *An example of a directed graph*

A numerical value can also be assigned to the edges within a graph to represent the importance of the edge, a concept known as a weighted graph. The importance can be incorporated by using numerical values or other drawing techniques, where the less important edges are indicated by the dashed lines in Fig. 4.

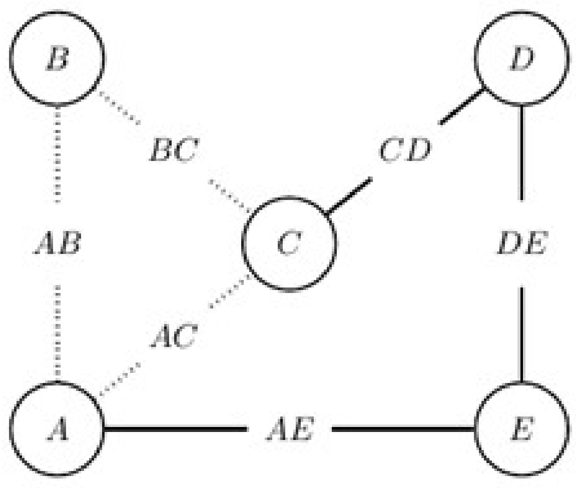

Figure 4: *An example of a weighted graph*

Graphs can also be further divided into smaller components known as subgraphs. A subgraph H of graph G is considered a clique if each node in the subgraph is connected by an edge to every other node within H. A particular subgraph H can also be thought of as a maximal clique in G in cases where H is a clique and there is no node in G but not in H that sends an edge to every node of H [76].

SNA is a broad field of study and many relations and sub-specialisations have been identified and studied. Because this paper uses SNA and some of its associated metrics, it was necessary to mention only a few of the basic ideas of SNA. This section is by no means an exhaustive overview of graph theory or SNA. The field itself is a highly mathematical area of study, and although the mathematical details do not form part of the scope of this research, interested readers are referred to [7] for more detail on the mathematical foundations of graph theory.

4 Methodology and experimental work

The validation of the proposed approach using a real-world data set sourced from the financial services industry is presented in this section. The discussion includes the data used in the study, the creation of network features using SNA, conducting a practical experiment whereby the network features are included to train an anomaly detection model, and finally, a discussion on the experimental results of the study.

4.1 The transactional dataset

The study was conducted using a data set sourced from an organisation operating in the financial services industry. The data are anonymous in nature, and care was taken not to expose any sensitive user information during the data extraction process. Two components make up the data set, namely a set of standard transactional features and a set of links between the same users in the set of standard transactional features, which makes the calculation of SNA metrics possible.

The standard transactional features are comprised of 120,676 records, of which 439 records are confirmed as fraudulent. This equates to a class imbalance where only 0.36% of the data are attributed to the minority class. The reason for the imbalance is that levels of fraud usually are far lower than the number of valid transactions that an organisation process. Therefore, far more data are collected on valid transactions compared with anomalous transactions, hence the imbalance in favour of the valid transactions. Highly imbalanced data sets are also reported in the literature as commonplace in the fraud detection context [66]. While class imbalances generally occur in standard classification problems, highly imbalanced tend to fall into the anomaly detection domain, where algorithms have been specifically developed to handle these imbalances. Standard classification approaches are not well suited to highly imbalanced problems, as the majority class overwhelms the training process and performance metrics associated with these. An example of a sampling technique that addresses oversampling the minority class is the synthetic minority oversampling technique (SMOTE) [34].

To address the problem of an imbalanced data set, this paper uses stratified sampling combined with a well-known and proven anomaly detection approach that trains a model on the majority class only, for which there is sufficient data [36]. This combined methodology was devised in order to prevent bias when creating the various data partitions required for effective model training and performance evaluation.

Stratified sampling is a sampling approach where the population data set is partitioned into non-overlapping groups known as strata [46]. In the case of a binary classification problem, the population data set can be partitioned into two strata, namely the positive and negative classes. When constructing training, validation, and test data sets, proportional allocation of the classes can be applied via stratified sampling. This means that any class imbalance exhibited in the population data set can be enforced in the subsequent training, validation, and test set samples, ultimately removing potential bias from the sampling process by ensuring that the balance of the classes is represented correctly in all samples.

The approach to train a model on the majority class only, produces a function that can be leveraged to detect non-anomalous transactions that are similar to the data that were used to train the model. Conversely, the anomaly detection model would flag anomalous transactions that are highly dissimilar from the data used to train the model. This is advantageous, as it allows researchers and practitioners to circumvent the issues related to training a binary classifier on data with limited instances of the minority class. In cases where there are no opportunities to collect additional positively labelled data, or where over and under-sampling techniques fall short, this approach becomes a viable option.

The sampling and model building, based on the above methodologies, is presented in Figure 5 and can be summarised as follows. The sampling process begins with $n = 120,676$ records, which are sorted in random order to prevent sampling bias. These are then split into training set T_1 and test set T_2, where T_1 contains 90% and T_2 contains the remaining 10% of the transactional dataset D. T_1 and T_2 are split using stratified sampling, so that each resulting partition contains a proportional share of the anomalous class. T_2 can then be considered the hold-out test set, while T_1 is used for further processing.

Training set T_1 is randomised again and subsequently split into a further training set T_3 and a validation set T_4. T_3 comprises 90% of T_1 while T_4 is made up from the remaining 10%, with stratified sampling again ensuring that the partitions receive their proportional share of the minority class.

S_1 is the set of fraudulent instances in T_3, and is removed so that the final training set contains data from the non-anomalous class only. On the other hand, validation set T_4 keeps its set of fraudulent instances S_2. This enables training on the non-anomalous cases only while validating the performance on data including both classes. Following the determination of the optimal architecture, hold-out test set T_2 is used to make a final evaluation of the model's performance on unseen data.

The set of fraudulent instances (S_1) removed from T_3 are then added onto validation set T_4 to augment the validation process with additional fraudulent samples. Removing the set S_1 from the training set yields the final training set T_5, whereas the updated validation set including S_1 becomes T_6. The final validation set T_6 ends up including set S_1, which contains 395 fraudulent instances, while the hold-out test set T_2 contains set S_2 with 44 fraudulent instances. Together these make up the set S, which includes all 439 fraudulent instances from the original transactional data set.

There are 49 features in the data set, where one is the unique identifier of each user, and another is the class label or target variable indicating a confirmed credit card fraud case. Instances of confirmed fraud are assigned a target variable of 1, while instances of no credit card fraud are assigned a target variable of 0. For the purposes of model training, the identifier is excluded but can be used to map the vectors and their respective prediction output back to each user's unique identifier. The class label is also kept aside but will be used for classification purposes. Therefore, the 49 features are reduced to 47 features for the purposes of training each model. No missing values are found in the data set, as the aggregation process used returns a 0 rather than a null value in instances where no values are counted for the respective feature. The real-world data features are transformed via standardisation to a $N(0, 1)$ distribution. Standardisation entails centering each feature around its mean with a unit standard deviation. The values returned by the standardisation process are not defined to a particular range as is the case with normalisation, and therefore values greater than 1 are to be expected.

Due to the confidentiality of the data, and considering the sensitivity of financial data, the remaining 47 features will not be described in detail. These features are typical features found in a transactional database and relate to financial and behavioural attributes associated with each transaction. Examples include the IP address from where the transaction originates, how many cards the user has attempted to use, and whether there are differ-

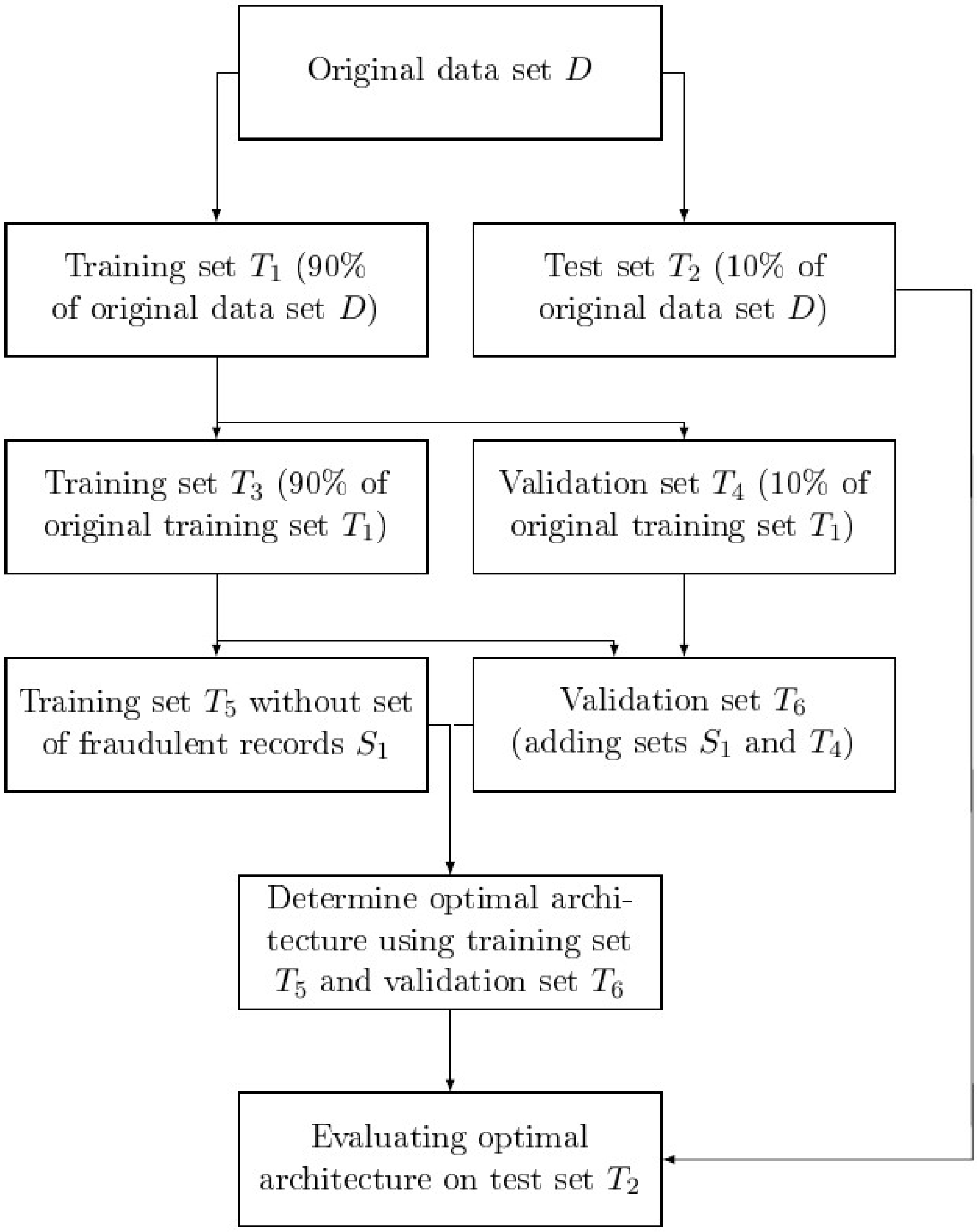
Original data set D
Training set T_1 (90% of original data set D)
Test set T_2 (10% of original data set D)
Training set T_3 (90% of original training set T_1)
Validation set T_4 (10% of original training set T_1)
Training set T_5 without set of fraudulent records S_1
Validation set T_6 (adding sets S_1 and T_4)
Determine optimal architecture using training set T_5 and validation set T_6
Evaluating optimal architecture on test set T_2

Figure 5: *Sampling process.*

ences between which country the user is signed in from and where their identity documents suggest they are located.

4.2 Extending the unified approach to include social network analysis metrics

Following on from the creation of the transactional features in Section 4.1, data representing the links or relationships between users are considered in order to structure the users and their interactions as a graph. Examples of links between users would include the same credit card, IP address, or login device used. Using the links between users as edges, and the users themselves as nodes, a network structure was formed, which could be explored via several network modelling techniques. This exploration led to the creation of features describing the underlying network metrics. The intention is to extend the unified approach by incorporating SNA metrics into the transactional features so that a final, comprehensive feature set can be used for anomaly detection.

Considering the large size of the clusters of network data, only certain visual examples of the process to incorporate the social network metrics will be given. Firstly, a smaller cluster with predominantly non-anomalous nodes is shown in Fig. 6, highlighting the non-anomalous and locked nodes, as well as the edge connections between them. In Fig. 6, the white nodes indicate non-anomalous users, whereas the black node indicates a user account in a locked state. A locked state is defined as the state when an account has been closed and all of its associated functionality has been removed. This can be temporary or permanent, depending on the severity of the suspicious activity associated with the account. The edges are comprised of a combination of device or IP address links between users.

The largest cluster in the entire social network is presented in Fig. 7. This cluster is a densely connected web of anomalous, locked, and non-anomalous accounts and contains 1,515 distinct nodes, including 8 instances of fraud as indicated by the large dark nodes. Some segments of the cluster splinter off into smaller, yet connected clusters, while the majority of the remaining nodes form a dense mesh of hundreds of accounts. This cluster, while alarming, would be challenging to investigate manually.

In terms of selecting the appropriate network metrics to calculate on this graph, a number of related studies were considered to provide assistance with these choices. Centrality metrics were an essential set of features used by Bonilla [12] and Van Vlasselaer *et al.* [103], prompting the selection of the degree, eigenvector, closeness, betweenness, load, and harmonic centrality in this study. Van Vlasselaer *et al.* [103] combined transactional features with network-based features to effectively detect credit card fraud in an e-commerce setting, where the inclusion of the network features was shown to have a powerful impact on the prediction results. Similarly, Bonilla [12] used a variety of network features to detect blockchain-related fraud, with their feature importance process indicating that centrality metrics were strong indicators of fraud.

Degree centrality is the fraction of nodes that a node is connected to within a network and can be summarised as the number of ties that any nodes have to other nodes in a graph [38]. A higher degree centrality indicates that nodes are well connected within the net-

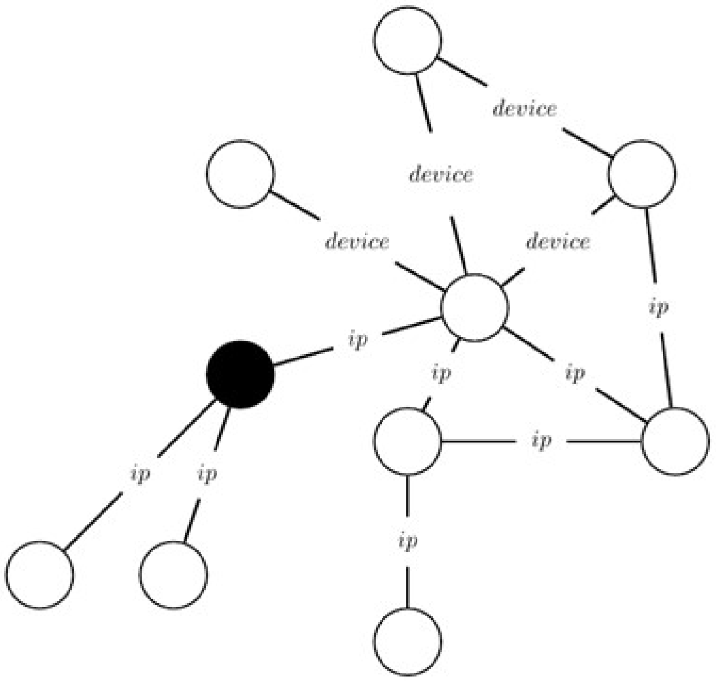

Figure 6: *An example of a subgraph within the real-world dataset.*

work, which is not standard behaviour in a financial services context. Users should ideally not be sharing devices, credit cards, or IP addresses, and therefore the more connected a user is, the more likely they are involved in suspicious activity. The second centrality measured used is eigenvector centrality. Eigenvector centrality is similar to degree centrality, although it considers that connections to nodes that are themselves influential contribute more to the centrality score of the node. Therefore, the higher the eigenvector centrality score of a node, the more connected the node is to other nodes which themselves have high scores [40]. Closeness centrality is the third centrality metric used, measuring the average length of the shortest path between the node in question and all other nodes within a graph [76]. The fourth centrality metric is betweenness centrality, which determines the number of times that a node acts as a bridge along the shortest path between two other nodes in the network [40]. Load centrality, the fifth centrality metric used, is similar to the betweenness centrality, and determines the fraction of all shortest paths that pass through each respective node [15]. The sixth and final centrality metric used, harmonic centrality, is considered to be a practical representation of an average shortest path [74]. Centrality metrics all measure the connectedness of nodes within a network in different ways, and higher centrality metrics all tend to indicate a higher likelihood of anomalous behaviour in an anomaly detection context.

In addition to the choice of centrality metrics, various other network metric approaches were also considered. Hamilton [40] found that the clustering coefficient metric was an important metric for measuring importance within graphs, while Yin *et al.* [107] motivated that clustering metrics can be used as powerful outlier detection signals. Link analysis metrics like PageRank were effectively used by Fakhraei *et al.* [35], with the clique metric

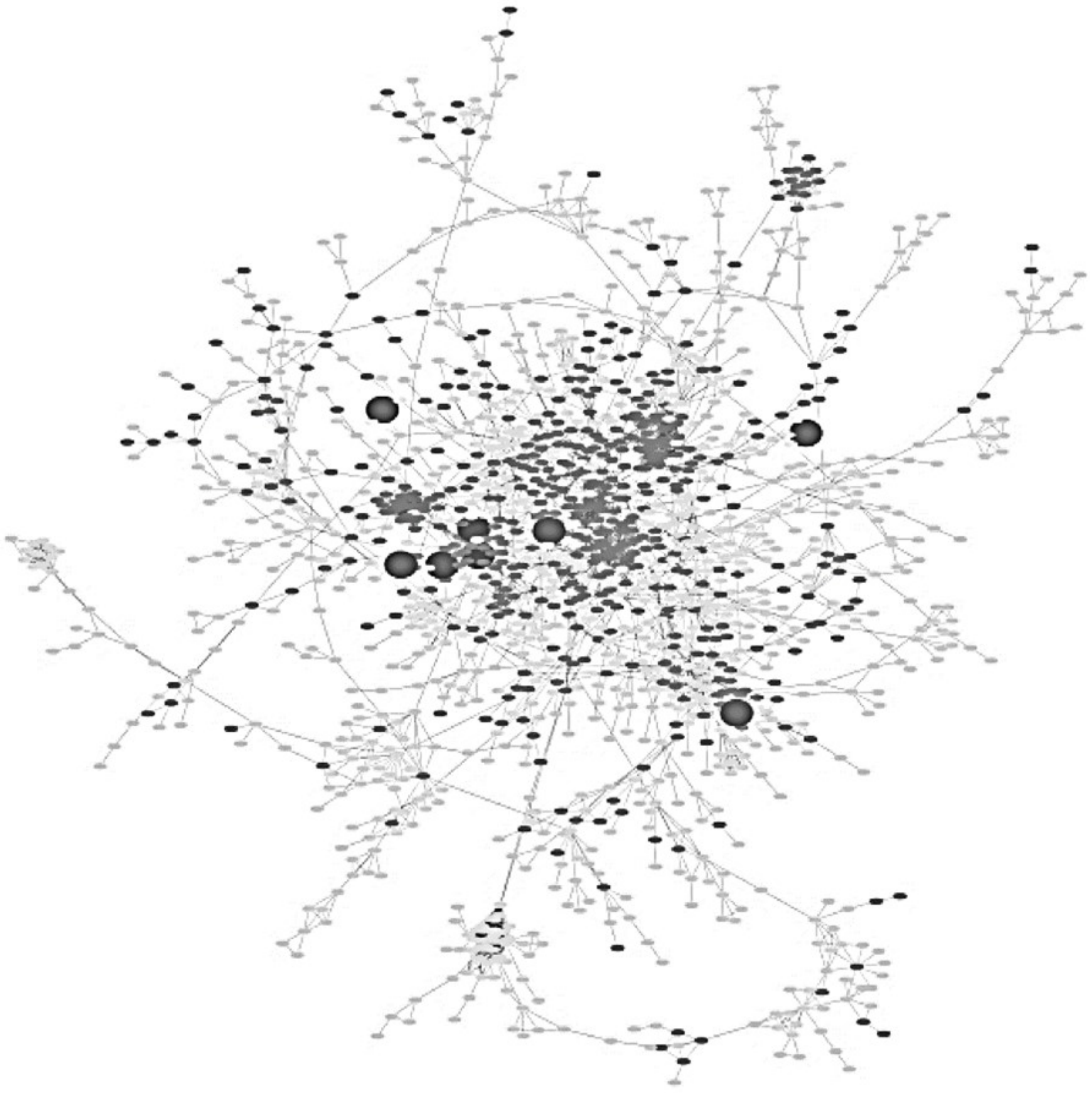

Figure 7: *Largest cluster in the overall social network.*

also showing promise as an effective method for determining groups within graphs [74]. Finally, assortativity metrics were found to be strong indicators of anomalies, according to Piraveenan *et al.* [83].

Within the clustering group of metrics, both the clustering coefficient and square clustering coefficient metrics were used. Negro [75] suggests that the clustering coefficient metric is vital in fraud detection, where nodes with a lower coefficient compared to the global average indicate that the nodes are connected to nodes that belong to independent groups. Being linked to an independent group can be suspicious in some contexts, as it suggests that users might be involved in a coordinated fraudulent syndicate. Similarly, the square clustering coefficient determines the probability that two neighbours of a node share another common neighbour [60].

The clique family of network metrics were also considered, as the grouping of nodes has shown to be an essential method for determining the likeness of network participants [80]. A node clique number finds the largest clique containing a node, where a clique is a complete graph where all of the nodes are connected to each other. The number of cliques that a node belongs to is also determined. This metric determines the number of unique cliques that contain each node, which is complementary to the node clique number metric, which only captures the largest clique containing the node [76]. The larger the number of cliques or node clique number per node, the more connected the user is within the network, and ultimately more likely to act suspiciously.

In total, the overall network contains 27,361 nodes and 34,957 edges. Considering the large volume of nodes, the computed features for only the first five nodes in the network are displayed in Table 1. In addition to the network metrics described previously, there is an opportunity to compute custom network metrics using the labelled data. The first two network features listed in Table 1 are reliant on the industry organisation's internal labelling of the users in the data set as being fraudulent or having an account that is in a locked state. A user that is marked as fraudulent would always have a locked account, but a user with a locked account does not always imply that the user is indeed fraudulent. There are a number of operational reasons why a user might be locked, most of which are an indication of a previous negative event linked to the user. The assumptions with these features are that being closely linked to other users with a history of negative behaviour could potentially indicate fraud for the user in question. The proximity of all nodes to known fraudulent or locked nodes is determined by applying Dijkstra's algorithm [45] to determine the shortest path to any anomalous node.

The balance of the features in Table 1 are standard network modelling metrics related to the users' influence within the network of linked users. The network can be defined as the subset of the original industry data set where the underlying users have a connection to another user. Therefore, there is no guarantee that if a user exists in the original industry data that they will appear in the network.

In Appendix A, the network features are transformed to a $N(0,1)$ distribution via the same process, and joined to the original industry data set to form the expanded industry data set, including network metric features. The features in Appendix A represent an abstraction of the data following the standardisation process, and is taken from the industry data set including the network metrics. The network metrics set that was joined onto the

Table 1: *Raw network metrics for the first five nodes*

Feature	Node 1	Node 2	Node 3	Node 4	Node 5
Distance closest fraud user	0	3	3	1	0
Distance closest locked user	0	1	1	1	0
Degree centrality	0.000073	0.000183	0.000146	0.000256	0.000146
Eigenvector centrality	1.235e-25	6.032e-07	6.019e-07	2.927e-06	2.59e-19
Closeness centrality	0.0000	0.007691	0.007690	0.009173	0.000073
Betweenness centrality	0.0000	0.000004	0.000000	0.000103	0.000000
Load centrality	0.0000	0.000004	0.000000	0.000103	0.000000
Harmonic centrality	0.0000	251.35148	250.85148	298.61873	2.000000
Average neighbour degree	1.0000	5.400000	6.000000	9.571429	3.000000
Clustering coefficient	0.0000	0.333333	1.000000	0.600000	1.000000
Node clique number	1	3	3	4	3
Number of cliques	1	2	1	2	1
Square clustering coefficient	0.0000	0.636364	6.000000	0.074488	0.666667
PageRank	0.000037	0.000041	0.000026	0.000027	0.000037

transactional feature set end up being sparsely populated. This sparsity is due to the relatively smaller number of users in the network data set when compared with the full set of users in the transactional feature set. There are fewer users in the network data set because not all users have a connection to another user. Being connected to another user is somewhat suspicious in itself, and therefore it is expected that a small proportion of the users in the transactional features are represented as nodes in the network metric features data.

The final set of features is now available, including both the transactional features and the network metric features determined in Sections 4.1 and 4.2 respectively.

4.3 Practical experiment

In this section, the combined transactional and network metrics features constructed in Sections 4.1 and 4.2 are used to train an autoencoder model to detect anomalies in the data. The model training process involves applying the unified approach proposed by Ball *et al.* [8] and is detailed in Section 3.2. The unified approach starts by simulating 100 candidate autoencoder architectures via a neural architecture search (NAS), where the optimal architecture is selected based on a balanced score comprised of the mean of the average precision (AP), area under the ROC curve (AUC), and normalised Matthews correlation coefficient (MCC) scores.

The NAS process, as denoted by Algorithm 1, found that models trained on the combined transactional and network metrics feature set yielded an optimal architecture balanced score of 94.07%. The results of the NAS process are displayed in Table 2. The optimal architecture had nine hidden layers, with a bottleneck of three neurons. The architecture column indicates the number of neurons in each hidden layer of the autoencoder, ranging from the bottleneck through to the final hidden layer prior to the output layer, essentially representing the decoder component. Symmetric autoencoder architectures are assumed, therefore the encoder component can be considered to be the mirror image of the decoder. The "Param" column notes the number of trainable parameters that were generated, while the "Thres" column lists the optimal threshold for the architecture discovered while optimising for a maximal balanced score.

Table 2: *Details of top 20 architectures using the combined transactional and network features*

AP	AUC	MCC	Balanced score	Architecture	Param	Thres
0.885	0.997	0.938	0.940	[3, 23, 34, 41, 42]	13402	4.242
0.883	0.996	0.937	0.939	[18, 19, 22, 26, 28, 38, 43, 46]	19607	3.636
0.858	0.995	0.923	0.925	[12, 21, 24, 25, 38, 41]	13101	3.737
0.853	0.996	0.920	0.923	[5, 7, 12, 17, 19, 26, 39]	9372	4.242
0.850	0.995	0.919	0.921	[4, 16, 23, 30, 34, 41, 45]	16695	4.242
0.844	0.994	0.915	0.918	[3, 4, 27, 31, 35, 40, 43]	15994	4.343
0.835	0.996	0.910	0.914	[13, 16, 17, 22, 27, 35]	9898	3.131
0.819	0.995	0.901	0.905	[8, 10, 14, 19, 22, 27, 35]	9479	4.343
0.814	0.995	0.898	0.902	[3, 7, 11, 20, 33, 38, 46]	13946	4.444
0.812	0.995	0.897	0.902	[4, 12, 15, 22, 25, 26, 34]	9765	4.242
0.805	0.994	0.893	0.897	[4, 14, 19, 24, 29]	6723	3.939
0.791	0.995	0.885	0.890	[8, 13, 15, 17, 20, 29, 35]	9575	3.939
0.790	0.995	0.884	0.889	[3, 24, 29, 30, 43]	11418	3.737
0.786	0.993	0.882	0.887	[3, 8, 18, 23, 26, 29, 32, 42]	13956	4.343
0.782	0.994	0.879	0.885	[4, 12, 16, 18, 20, 22, 26, 32]	9725	4.040
0.774	0.994	0.875	0.881	[3, 8, 18, 22, 25, 29, 31, 39, 45]	17392	4.242
0.775	0.989	0.876	0.880	[6, 22, 33, 42]	9873	2.424
0.766	0.994	0.870	0.877	[6, 9, 15, 20, 30, 35, 36]	11547	3.535
0.765	0.994	0.870	0.876	[4, 5, 27, 32, 35, 38, 46]	16477	4.040
0.766	0.993	0.870	0.876	[5, 7, 10, 11, 12, 19, 22, 38]	8598	4.444

Once the optimal architecture has been established, the next step in the unified approach is to determine the optimal classification threshold for the autoencoder. The threshold is required in order to split the reconstruction errors produced by the trained model into the non-anomalous and anomalous groups. In the case of fraud detection, reconstruction errors above the threshold would indicate fraudulent transactions, while reconstruction errors below the threshold are deemed to be non-anomalous. Araya *et al.* [4] refer to a threshold as being typically defined by the mean square error (MSE). The optimal threshold for the autoencoder is determined by applying the threshold optimisation algorithm presented as Algorithm 2, where the optimal threshold for the autoencoder model was determined to be a MSE of 5.353.

The optimal threshold for the model is depicted in Fig. 8, where the figure indicates that the peak balanced score of just over 94% on the y-axis corresponds to a MSE threshold of 4.545 on the x-axis. Fig. 9 is a graphical representation of the reconstruction errors, where the bold line in the figure is the threshold that attempts to split the non-anomalous and fraudulent transactions.

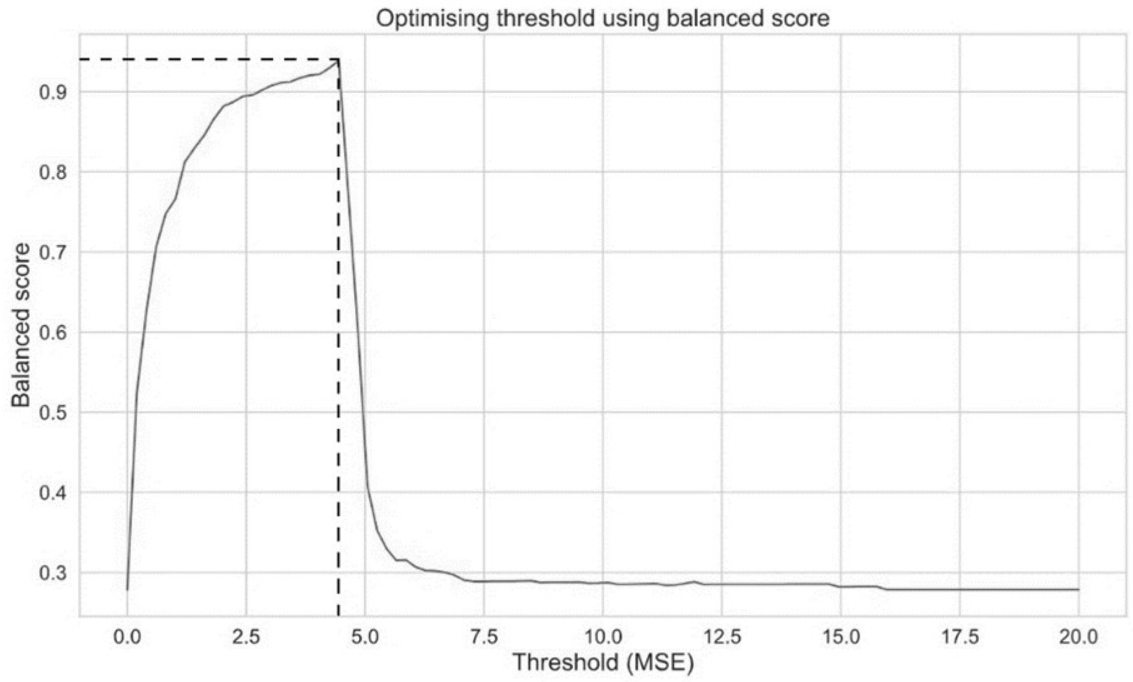

Figure 8: *Optimising combined model threshold selection using the balanced score.*

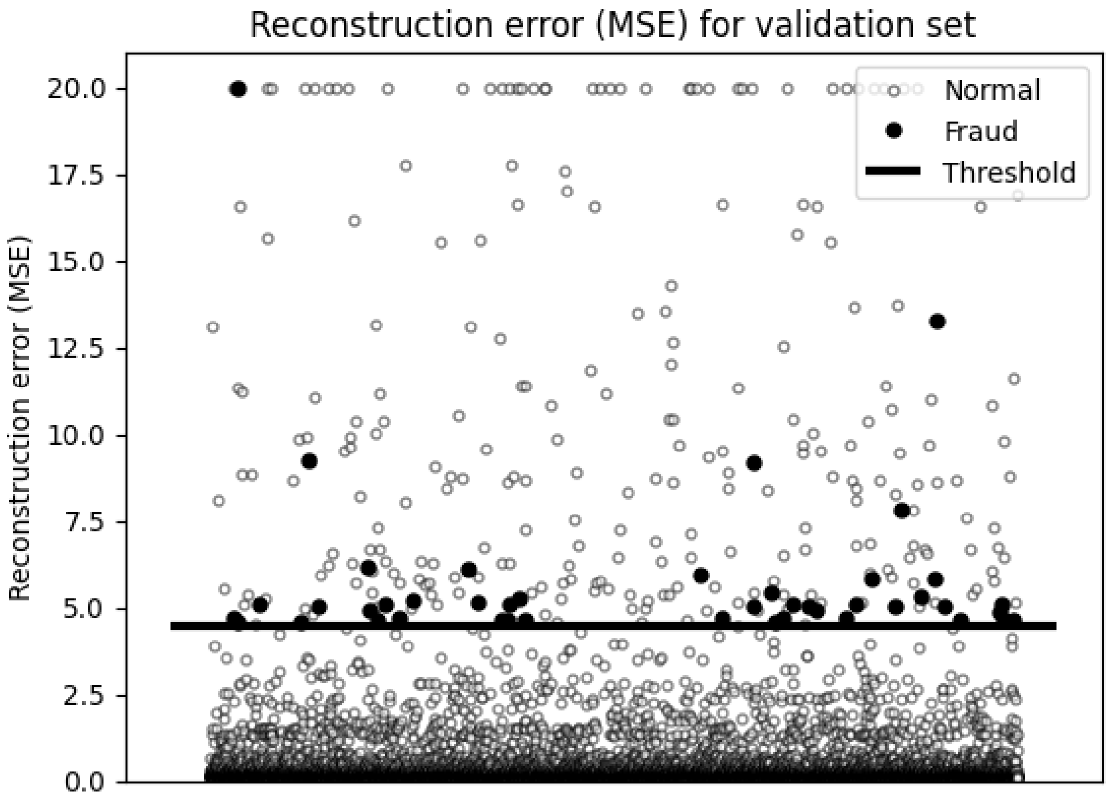

Figure 9: *Reconstruction scores (MSE) using the combined validation data.*

Following the establishment of the optimal thresholds, the third phase of the unified approach, Gaussian scaling, is applied to the raw anomalies scores, as described by Algorithm 3. This helps with scaling the raw scores into an interpretable $[0; 1]$ range, which can be used and easily understood in an operational context. The Gaussian scaled anomaly scores for the model is illustrated by Fig. 10. The darker bars indicate the non-anomalous transactions, which are expected to be scored lower in relation to the lighter bars indicating the fraudulent transactions. Fig. 10 groups the fraudulent transactions in the 95%+ bucket, whereas the non-anomalous transactions are grouped into various buckets, although the vast majority are in the desired lowest bucket.

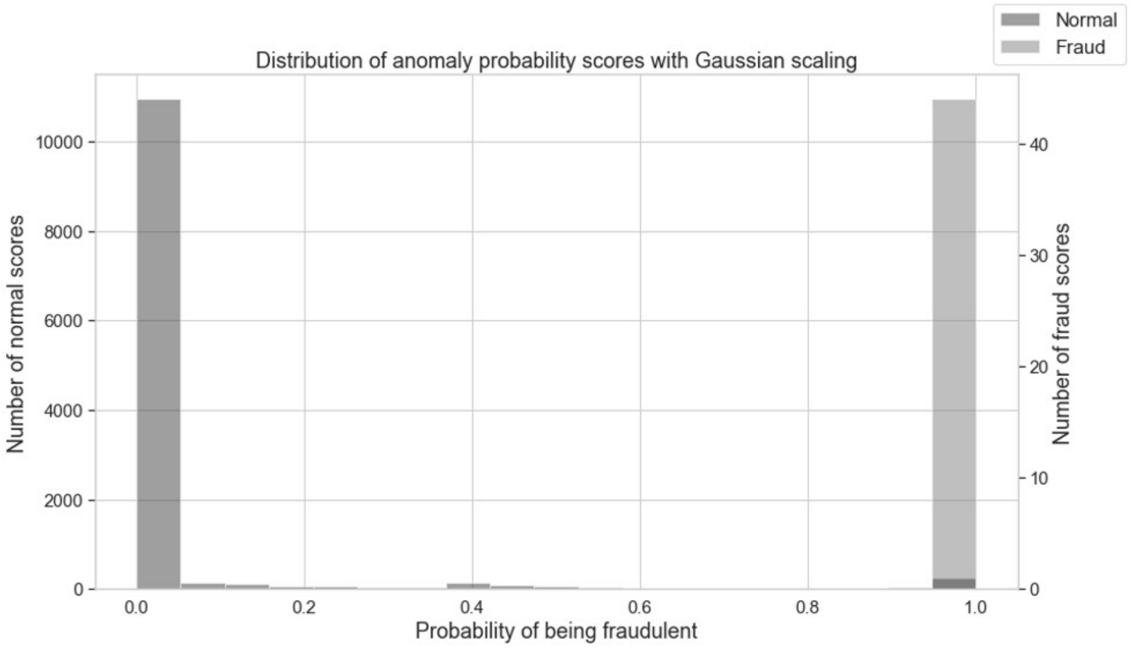

Figure 10: *Anomaly scores with Gaussian scaling.*

4.4 Experimental results

Table 3 captures the performance of the optimal autoencoder architecture on the feature hold-out test set. This test set contains 12,038 records, of which 44 are confirmed fraud cases. The optimal model correctly classified all 44 instances of fraud while misclassifying 125 non-anomalous transactions as fraudulent.

There are many performance measures that are traditionally used in the evaluation of a model. The most popular measures include average accuracy, average error rate, precision or sensitivity, recall and F_1-score. In this research, it was decided to use the precision and recall measures as the primary performance indicators. Precision is defined as the fraction of relevant instances among the retrieved instances and, for a binary classification problem, it is mathematically expressed as

$$\text{Precision} = \frac{tp}{tp + fp}$$

where tp indicates the number of true positives, or the number of confirmed anomalous instances correctly classified as such, and fp indicates the number of false positives, or the number of confirmed non-anomalous instances that were incorrectly classified as anomalous. The relevant instances in the precision formula are those predicted as anomalous; therefore, precision is the number of correctly classified anomalous instances over the total number of predicted anomalous instances.

Recall pertains to the fraction of total relevant instances that were retrieved and for a binary classification problem is formally defined as

$$\text{Recall} = \frac{tp}{tp + fn}$$

where fn is the number of false negatives, or the number of confirmed anomalous instances that are incorrectly classified as non-anomalous. Recall can therefore be considered the number of correctly classified anomalous instances over the total number of confirmed anomalous instances.

The final results obtained was a precision score of 26% and a recall score of 100%. Due to the highly imbalanced nature of the data, it is expected that the model will realise low precision scores. Al-Shabi [1] confirmed that the balance between precision and recall is a trade-off that needs to be evaluated by the organisation in question, as there are unique costs associated with both types of misclassifications. The trade-off between precision and recall can be quantified by applying a cost-based approach to capture the financial impact of fraud. This could be achieved by attaching a predetermined cost to each quadrant of the confusion matrix (see Table 3) to compute the total cost impact at differing precision-recall thresholds [6].

The F_1-score (the harmonic mean between precision and recall) is also widely acknowledged as a valuable metric in many studies [99]. However, it has faced some criticism and Hand and Christen [41], for example, criticised F_1-scores for placing equal importance on precision and recall. This becomes problematic in practice, where different misclassifications carry different cost implications. The argument is important in fraud detection, where it is often more expensive to misclassify a positive instance of fraud as negative than a non-anomalous instance as positive [94]. The accuracy measure was also not used as a performance measure as it is often not considered an effective performance metric due to the imbalance between the classes. This is known as the accuracy paradox [102].

Table 3: *Confusion matrix of optimal architecture with combined features*

		Predicted class		
		Fraud	Not fraud	Total
True class	Fraud	44	0	44
	Not fraud	125	11869	11994
	Total	169	11869	12038

Another important factor in assessing the effectiveness of a trained model is model interpretability. To evaluate the model obtained, it was decided to apply a SHAP analysis to the results. SHAP is a technique for assigning feature importance for model predictions [65], where the contribution of each feature to specific predictions are explained [72]. The technique has also been adapted to the anomaly detection domain, where Antwarg *et al.*

[3] showed success in applying SHAP on the output scores generated by autoencoder models. Applying SHAP on an autoencoder model output entails explicitly determining the connection between the features that have a high reconstruction error and those features that are most important in affecting the reconstruction error of the model.

The results of applying SHAP to the model outputs are shown in Fig. 11. The transactional features are depicted by the darker bars, while the lighter bars highlight the contributions of the network features in descending order of influence. Features $v1$ through $v47$ indicate the transactional features, whereas features $v48$ through $v61$ indicate the network metrics features. It is clear that the most important transactional and network features each have an equal share of the top ten features displayed in Fig. 11.

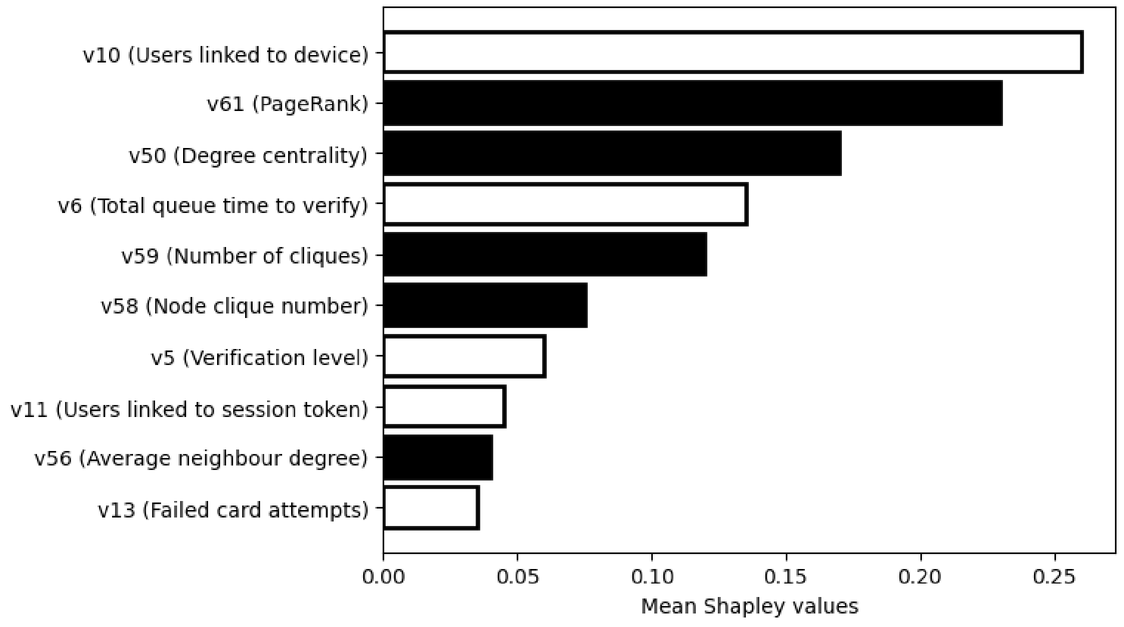

Figure 11: *Shapley values for autoencoder with combined features.*

As can be seen from Fig 11, the top ten features are comprised of five from the transactional data and five from the newly suggested network metrics. The most important transactional features (indicated by the transparent bars in Fig 11) in the final model include the following:

- *Users linked to device ($v10$):* This feature is the most important of all and indicates whether other users are using the same registered mobile device. In the context of fraud detection, it is important to know whether users share a registered device because it is assumed that non-anomalous users only use their own devices and do not allow others to transact via their devices.

- *Total queue time to verify ($v6$):* The second most important transactional feature is v6 which is also ranked fourth among the top ten features. It shows whether a user progressed through the sign-up process swiftly, which may indicate that the user has signed up with multiple fictitious accounts previously. This signifies that the user has gained a substantial amount of experience in completing the sign-up process, which is not considered non-anomalous, where a non-anomalous user should only need to sign-up once. The feature may also detect an automated approach to completing the sign-up process, such as copying and pasting data from a template, which too is considered suspicious behaviour.

- *Verification level ($v5$):* The third most important transactional feature, verification level, indicates the level of transactional restrictions placed on an account. A higher verification level denotes a lower level of restriction, as more is known about the customer from a KYC (Know Your Customer) perspective. Both non-anomalous and fraudulent users would benefit from having a high verification level so that they are able to transact with relatively large financial amounts.
- *Users linked via a session token ($v11$):* The fourth most substantial transactional feature, users linked via a session token, measures the number of different users who transacted via a uniquely generated web token. As with users linked to device, the assumption is that non-anomalous users would not transact via the same web token assigned to another user. Web tokens are generated by the organisation, whereas devices are registered using their physical MAC (media access control) address, of which the first is seen to be more accurate in detecting user devices.
- *Failed card attempts ($v13$):* The fifth and final important transactional feature, failed card attempts, indicates the number of times that a user attempted to load a card that was declined by the issuing bank. A high number of failed card attempts is suspicious behaviour, as it might suggest that a user is testing a number of stolen cards, an activity known as card testing.

The most important network metrics (indicated by the dark bars in Fig 11) that were identified by the SHAP analysis and which are proposed to augment the standard transactional variables are as follows:

- *PageRank ($v61$):* The most important network metrics feature is the PageRank metric, which determines the importance of nodes in a graph based on the structure of the incoming edges. This results in a measure that captures the influence of a node within a graph, where influence is determined as the number of high-quality connections that a given node has to other nodes. High-quality connections are links to nodes which themselves are highly connected, which can be considered suspicious behaviour within a financial services ecosystem.
- *Degree centrality ($v50$):* The second most important network metric feature is degree centrality. This feature computes how many connections nodes have compared to all other nodes within a network. Therefore, nodes connected to more users than other nodes are more likely to be anomalous. Conversely, it is assumed that non-anomalous users would not be strongly connected to other users in a financial ecosystem such as this.
- *Number of cliques ($v59$):* Cliques or subgraphs are also strong contributing features, acting as another measure for how connected a node is to large clusters within the

graph. The number of cliques determines the number of possible subgraphs that each node is connected to, and is the third most important network metrics feature for detecting anomalies in the data. The higher the number of cliques, the more connected a user is within the network, which is deemed to be suspicious.

- *Node clique number ($v58$):* The fourth most important network metrics feature is the node clique number. A larger node clique number indicates that the user is strongly connected to other users and helps to surface anomalous behaviour within the network. The node clique number represents the largest clique containing each given node.
- *Average neighbour degree ($v56$):* The fifth and final strongly contributing network metrics feature is the average neighbour degree and is simply the average of the degree (number of connections to other nodes) of the neighbours that surround each node. A node that is closely associated with other densely connected nodes would generate a high average neighbour degree, an indication of potentially suspicious behaviour.

The above results indicate that the network metrics features are important contributors to detecting anomalies when using autoencoders. The two most important network metrics features, namely PageRank and degree centrality, make logical sense as both relate to the number of important links to the node within a network, which is important in anomaly detection.

5 Discussion and reflection

In line with the paper's objective, a methodology that combines an anomaly detection autoencoder model with additional network metrics was described in the preceding sections. The results are promising, and the combined feature model showed that adding the proposed network metrics features contributes substantially to detecting anomalies. The method followed in the study is based on a unified approach that is designed to determine a best autoencoder model architecture. The model is then extended by using a combination of transactional data and network metrics, typically found in SNA, to detect anomalous transactions in a real-world financial dataset. The discussion and reflection in this section will be presented in three parts, *i.e.*, the unified approach to select a "best" model architecture, the selection and inclusion of network metrics to augment the model and general recommendations.

Construction of an anomaly detection autoencoder model using a unified approach

The unified approach used to build a suitable model was described in Section 3.2 and applied in Section 4.3. The methodology consists of 3 main phases, *i.e.*, a neural architecture search, a threshold optimisation and a Gaussian scaling phase.

The proposed unified approach provides several advantages that will help to ensure that a "best" model architecture is used. It is a substantial improvement over the simple grid search algorithm traditionally used as it finds an optimal set of hyperparameters for a model in a relatively short time. The random nature of the neural architecture search outperforms a grid search, especially in cases where the number of possible hyperparameters substantially influence the model's performance. A grid search often suffers from the curse of dimensionality where the number of possible parameters may grow exponentially as the number of hyperparameters increases. These challenges in a traditional grid search, which are addressed with the proposed unified approach, are confirmed in the literature ([11]; [59]).

In fraud and anomaly detection problems, the unified approach also provides advantages. The optimal architecture is found by validating various hyperparameter configurations against performance metrics specifically chosen due to their suitability for datasets with imbalanced classes. These imbalanced datasets are often found in fraud-related datasets as the number of fraudulent transactions usually is far less than the valid ones. Three metrics appropriate to the anomaly detection context were chosen *i.e.*, average precision, the area under the ROC curve and a normalised Mathews correlation coefficient. These metrics were then combined to simplify the underlying metrics into a single balanced score.

A typical problem encountered in anomaly detection is the subjective threshold decision to identify anomalous transactions. This is frequently determined through visual inspection of a potential threshold. The proposed unified approach provides, through its second phase, an advantage over other threshold determination approaches, such as using the maximum mean square error from the training process. The method explicitly determines a threshold that maximises a given objective, namely a balanced score. Finally, the third component of the unified approach provides scaled, universally transferable probability scores as the autoencoder model output. These scores, transformed via Gaussian scaling, remove the friction typically caused by comparing raw anomaly scores from models with different magnitudes of scores. The unified approach chains all of these different aspects of the anomaly detection process together to form a functional pipeline that can be extended for future academic and commercial applications.

Extending and incorporating appropriate network analysis metrics into the model

The network analysis metrics used in the anomaly detection model were described in Sections 3.3 and 4.2.

While the transactional features in this study capture the various transactional behaviours that users within the system exhibit, these features do not capture the relationships between users. Variables that explain relationships are essential in the context of fraud detection and often reveal additional information that may lead to the identification of anomalies. SNA provides an opportunity to transform data that captures the linked nature of relationships between users, effectively representing these relationships as a mathematical graph. The implications for this representation are twofold. Firstly, graph data allows for visualising the relationships between users in graphical form and secondly, SNA algorithms can be applied to the linked data to determine metrics that describe the connectedness of the users within the entire graph of users. Neither of these is possible without data that captures these relationships between users, therefore specific networked data was included in the unified approach to expand the set of features beyond the transactional level of aggregation.

Including network metrics in the unified approach entails representing the transactional data in graph form. The process of porting the linked data into graph form brings specific challenges. The task of creating the network metric features is important from a computational perspective, where the application of the various graph algorithms on the graph increased the feature engineering time substantially. The run-time of the model training process itself was not impacted meaningfully by the addition of the network metrics features, with training taking slightly longer due to a more extensive training data input space. The same goes for the complexity of the model, as model complexity is determined by the hyperparameters chosen during the neural architecture search process. There is no direct impact on model complexity by merely including network metric features. Another challenge is the required knowledge necessary to implement network metrics. Calculating the network metric features requires specific knowledge on network analysis and how to use it to calculate the network metrics. Should the knowledge be available to perform network analysis, it is recommended to calculate network metric features, specifically those metrics associated with centrality, and include them as additional features when training an autoencoder for anomaly detection in a transactional setting.

The network metrics selected for this study were chosen after consulting related studies that had either used manual network metrics for fraud detection purposes or used the network metrics as features to train a machine learning model. The autoencoder model trained using additional network metric features produced an output with a low number of false positives, which would lead to a low operational cost of fraud in the industry. By analysing the Shapley values for the trained model, the PageRank and degree centrality metrics showed the most substantial influence in detecting anomalies. These metrics highlight the importance of determining nodes with many links within a given network (in many cases, an indication of a possible anomalous transaction). The identified metrics are therefore helpful for anomaly detection purposes. Applying SHAP to the autoencoder output also proved to be useful as an approach to determine the strongest contributing or offsetting network metric features for any anomaly that may be detected. Another advantage in following this approach is that the method removes the need for investigation analysts to perform manual calculations to understand user relationships and interactions. Ultimately including network metrics in the unified approach brings an additional dimension to the problem of anomaly detection, as user behaviour can be modelled from both a transactional and network perspective. This was also supported by the encouraging model accuracy results, leading to a low operational cost of fraud in the industry.

General recommendations

Considering the results obtained in this study and the above discussion of the methodology, some final comments and recommendations can be summarised as follows.

- In a fraud detection scenario that is based on a transactional setting, it is recommended that an autoencoder neural network model be constructed to identify anomalous transactions. It would be advantageous to utilise the proposed unified approach to select a “best” model architecture.

- To make provision for underlying relationships and interactions of clients, which is a vital part of fraud detection techniques, the inclusion of specific SNA metrics as additional explanatory variables should be considered. This study has shown that centrality metrics are particularly important in detecting anomalies.

- Model performance metrics should be selected with care. In a fraud context, standard accuracy measures such as the average accuracy or F_1-scores may be misleading. This is due to factors such as the existence of imbalanced datasets and the critical principle of trade-offs (*e.g.*, the cost of false negatives).

- Finally, cognisance should be taken of practical considerations when considering the use of the proposed unified approach and the implementation of social network features. Some of these considerations include model construction complexity (a substantial amount of technical knowledge is required to build and implement the models); time complexity (model selection, model training and network feature engineering may take some time); model complexity (relates to the number of hyperparameters in the model); and prerequisite knowledge required (a good understanding of neural networks – autoencoders in particular – and SNA with the associated metrics are needed).

6 Conclusion

The objective of this paper was to extend a previously proposed unified approach by incorporating network metrics features. These network metrics features were calculated by applying SNA techniques on data capturing the interactions between users within a financial system.

The unified approach was applied on a combined feature set including both transactional and network metrics features. Performance analysis was conducted on the results of the unified approach, in order to determine whether the incorporation of the network metric features had a important impact on the ability of the autoencoder to detect anomalies in a transactional context.

The performance evaluation indicated that a favourable balanced score was achieved after introducing the network metric features, which assisted in the model reporting a low number of false positives. Applying SHAP to the model outputs showed that five of the network features appeared in the top ten list of contributing features. The most important of these network metric features were the PageRank, degree centrality, and number of cliques features. It was found that the network metrics features were considerably more complex to generate in comparison with the transactional features however, and this complexity needs to be catered for in any real-world application of the approach.

Appendix A

The table presented as Appendix A details the first twenty rows of the network metrics features data. A column index is used instead of a column name to conserve space in the visualisation. While the data appears to be sparsely populated, there is enough information in the network metrics features to help the autoencoder detect anomalous transactions.

Table 4: *First 20 samples of the network metrics features*

0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0.07
0	0	0	0	0	0	0	0	0	0	0	0	0	0.08
0	0	0	0	0	0	0	0	0	0	0	0	0	0.09
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0.33	0.08	0.06	0	0.72	0	0	0.63	0.30	1	0.15	0	0.002	0.01
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0.27	0.08	0.04	0	0.72	0	0	0.6	0.2	1	0.12	0	0.01	0.02
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.6
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.06
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

References

[1] Al-Shabi MA, 2019, *Credit card fraud detection using autoencoder model in unbalanced datasets*, Journal of Advances in Mathematics and Computer Science, **33(5)**, pp. 1–16.

[2] An J & Cho S, 2015, *Variational autoencoder based anomaly detection using reconstruction probability*, Special Lecture on IE, **2(1)**, pp. 1–18.

[3] Antwarg L, Miller RM, Shapira B & Rokach L, 2021, *Explaining anomalies detected by autoencoders using Shapley additive explanations*, Expert Systems with Applications, **186**, pp. 115736.

[4] Araya DB, Grolinger K, ElYamany HF, Capretz MAM & Bitsuamlak G, 2016, *Collective contextual anomaly detection framework for smart buildings*, International Joint Conference on Neural Networks, pp. 511–518.

[5] Asha RB & Suresh Kumar KR, 2021, *Credit card fraud detection using artificial neural network*, Global Transitions Proceedings, **2(1)**, pp. 35–41.

[6] Bahnsen AC, Stojanovic A, Aouada D & Ottersten B, 2013, *Cost sensitive credit card fraud detection using Bayes minimum risk*, Machine Learning and Applications (ICMLA), **12(1)**, pp. 333–338.

[7] Balakrishnan R & Ranganathan K, 2012, *A textbook of Graph Theory*, Springer Science & Business Media, New York.

[8] Ball RS, Krüger HA & Drevin L, 2020, *A unified approach to anomaly detection*, Proceedings of the Sixth International Conference on Machine Learning, Optimization, and Data Science, pp. 281–291.

[9] Beigi G, Tang J & Liu H, 2020, *Social science-guided feature engineering: A novel approach to signed link analysis*, ACM Transactions on Intelligent Systems and Technology, **11(1)**, pp. 1–27.

[10] Bender EA & Williamson SG, 2010, *Lists, Decisions and Graphs. With an introduction to probability*, San Diego, CA, Edward A. Bender & S. Gill Williamson.

[11] Bergstra J & Bengio Y, 2012, *Random search for hyper-parameter optimization*, The Journal of Machine Learning Research, **12**, pp. 281–305.

[12] Bonilla JP, 2018, *Detection of fraud in financial blockchain-based transactions through big data analytics*, Master's dissertation, Universidad Carlos III de Madrid.

[13] Boquet G, Morell A, Serrano J & Vicario JL, 2020, *A variational autoencoder solution for road traffic forecasting systems: missing data imputation, dimension reduction, model selection and anomaly detection*, Transportation Research Part C: Emerging Technologies, **115**, pp. 102622.

[14] Borner K, Eide O, Mchedlidze T, Rehbein M & Scheuermann G, 2019, *Network visualisation in the humanities*, Dagstuhl Reports, **8(1)**, pp. 139–153.

[15] Brandes U, 2020, *On variants of shortest-path betweenness centrality and their generic computation*, Social Networks, **30(2)**, pp. 136–145.

[16] Bringmann LF, Elmer T, Epskamp S, Krause RW, Schoch D, Wichers M, Wigman J & Snippe E, 2019, *What do centrality measures measure in psychological networks*, Journal of Abnormal Psychology, **128(8)**, pp. 892–903.

[17] Burcher M & Whelan C, 2018, *Social network analysis as a tool for criminal intelligence: understanding its potential from the perspectives of intelligence analysts*, Trends in Organized Crime, **21**, pp. 278–294.

[18] Camacho D, Panizo-Lledot A, Bello-Orgaz G, Gonzalez-Pardo A & Cambria E, 2020, *The four dimensions of social network analysis: an overview of research methods, applications, and software tools*, Information Fusion, **63**, pp. 88–120.

[19] Carcillo F, Borgne Y, Caelen O, Kessaci Y, Oble F & Bontempi G, 2019, *Combining unsupervised and supervised learning in credit card fraud detection*, Information Sciences, **557**, pp. 317–331.

[20] Chalapathy R, Khoa NLD & Chawla S, 2020, *Robust deep learning methods for anomaly detection*, In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3507–3508.

[21] Chawla A, Jacob P, Lee B & Fallon S, 2019, *Bidirectional LSTM autoencoder for sequence-based anomaly detection in cyber security*, International Journal of Simulation – Systems, Science & Technology, **20(5)**, pp. 1–6.

[22] Chen J, Li J, Chen W, Wang Y & Jiang T, 2020, *Anomaly detection for wind turbines based on the reconstruction of condition parameters using stacked denoising autoencoders*, Renewable Energy, **147(1)**, pp. 1469–1480.

[23] Chen T, Liu X, Xia B, Wang W & Lai Y, 2020, *Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder*, IEEE Access, **8**, pp. 47072–47081.

[24] Chen Z, Yeo CK, Lee BS & Lau CT, 2018, *Autoencoder-based network anomaly detection*, In: 2018 Wireless Telecommunications Symposium (WTS), pp. 1–5.

[25] Chicco D & Jurman G, 2020, *The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation*, BMC Genomics, **21(1)**, pp. 1–13.

[26] Chow JK, Su Z, Wu J, Tan PS, Mao X & Wang YH, 2020, *Anomaly detection of defects on concrete structures with the convolutional autoencoder*, Advanced Engineering Informatics, **45**, pp. 101105.

[27] Colladon AF & Remondi E, 2017, *Using social network analysis to prevent money laundering*, Expert Systems with Applications, **67**, pp. 49–58.

[28] Cui Y, Sun Y, Hu J & Sheng G, 2018, , *A convolutional autoencoder method for anomaly detection on system logs*, In: 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 3057–3062.

[29] Dal Pozzolo A, 2015, *Adaptive machine learning for credit card fraud detection*, PhD thesis, Université Libre De Bruxelles.

[30] Dau HA, Ciesielski V & Song A, 2014, *Anomaly detection using replicator neural networks trained on examples of one class*, Simulated Evolution and Learning, **10**, pp. 311–322.

[31] Decuypere M, 2020, *Visual network analysis: a qualitative method for researching sociomaterial practice*, Qualitative Research, **20(1)**, pp. 73–90.

[32] Del Gaudio BL, Porzio C, Sampagnaro G & Verdoliva V, 2020, *How do mobile, internet and ICT diffusion affect the banking industry? An empirical analysis*, European Management Journal, **39(3)**, pp. 327–332.

[33] Ding K, Li J, Bhanushali R & Liu H, 2019, *Deep anomaly detection on attributed networks*, In: Proceedings of the 2019 SIAM International Conference on Data Mining, Society of Industrial and Applied Mathematics, pp. 594–602.

[34] Drummond C & Holt RC, 2003, *C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling*, In: Workshop on learning from imbalanced datasets II, **11**, pp. 1–8.

[35] Fakhraei S, Foulds J, Shashanka M & Getoor L, 2015, *Collective spammer detection evolving multi-relational social networks*, In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1769–1778.

[36] Gerych W, Agu E & Rundensteiner E, 2019, *Classifying depression in imbalanced datasets using an autoencoder-based anomaly detection approach*, In: 2019 IEEE 13th International Conference on Semantic Computing (ICSC), pp. 124–127.

[37] Goodfellow I, Bengio Y & Courville A, 2016, *Deep learning*, MIT Press, Cambridge, Massachusetts.

[38] Grando F, Granville LZ & Lamb LC, 2019, *CMachine learning in network centrality measures: tutorials and outlook*, ACM Computing Surveys, **51(5)**, pp. 1–32.

[39] Guo Y, Liao W, Wang Q, Yu L, Ji T & Li P, 2018, *Multidimensional time series anomaly detection: a GRU-based Gaussian mixture variational autoencoder approach*, In: Asian Conference on Machine Learning, PMLR, pp. 97–112.

[40] Hamilton WL, 2020, *Graph representation learning*, Synthesis Lectures on Artificial Intelligence and Machine Learning, **14(3)**, pp. 1–159.

[41] Hand DJ & Christen P, 2018, *A note on using the F-measure for evaluating record linkage algorithms*, Statistics and Computing, **28(3)**, pp. 539–547.

[42] He K, Lu Y & Sclaroff S, 2018, *A Local descriptors optimized for average precision*, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 596–605.

[43] Henderson R, 2020, *Using graph databases to detect financial fraud*, Computer Fraud and Security, **7**, pp. 6–10.

[44] Homayouni H, Ghosh S, Ray I, Gondalia S, Duggan J & Kahn MG, 2020, *An autocorrelation-based LSTM autoencoder for anomaly detection on time-series data*, In: 2020 IEEE International Conference on Big Data (Big Data), IEEE , pp. 5068–5077.

[45] Hossain MA, Ahmedy I, Harith MZM., Idris MYI, Soon TK, Noor RM, and Yusoff SB, 2020, *Route Optimization by using Dijkstra's Algorithm for the Waste Management System*, In: Proceedings of the 3rd International Conference on Information Science and Systems, pp. 110–114.

[46] Howell CR, Su W, Nassel AF, Agne AA & Cherrington AL, 2020, *Area based stratified random sampling using geospatial technology in a community-based survey*, BMC Public Health, **20(1)**, pp. 1–9.

[47] Ji C, Zou X, Liu S & Li P, 2020, *ADARC: An anomaly detection algorithm based on relative outlier distance and biseries correlation*, Software: Practice and Experience, **50(11)**, pp. 2065–2081.

[48] Johnson JM & Khoshgoftaar TM, 2019, *Survey on deep learning with class imbalance*, Journal of Big Data, **6(1)**, pp. 1–54.

[49] Khan S & Taati B, 2017, *Detecting unseen falls from wearable devices using channel-wise ensemble of autoencoders*, Expert Systems with Applications, **87**, pp. 280–290.

[50] Kivela M, McGee F, Melancon G, Riche NH & Von Landsberger T, 2019, *Visual analytics of multilayer networks across disciplines*, Dagstuhl Reports, **9(2)**, pp. 1–26.

[51] Kriegel H, Kroger P, Schubert E & Zimek A, 2011, *Interpreting and unifying outlier scores*, In: Proceedings of the 2011 SIAM International Conference on Data Mining, Mesa, Arizona, pp. 13–24.

[52] Kucher K, Fatemi M & Laitinen M, 2021, *Towards visual sociolinguistic network analysis*, In: Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics, **3**, pp. 248-255.

[53] Kumari A, Behera RK, Sahoo KS, Nayyar A, Luhach AK & Sahoo SP, 2020, *Supervised link prediction using structured-based feature extraction in social network*, Concurrency and Computation: Practice and Experience, pp. 1–16.

[54] Kurka DB, Godoy A & Von Zuben FJ, 2015, *Online social network analysis: a survey of research applications in computer science*, arXiv preprint arXiv:1504.05655.

[55] Leite RA, Gschwandtner T, Miksch S, Gstrein E & Kuntner J, 2018, *Visual analytics for event detection: focusing on fraud*, Visual Informatics, **2(4)**, pp. 198–212.

[56] Lesouple J, Baudoin C, Spigai M & Tourneret J, 2021, *Generalized isolation forest for anomaly detection*, Pattern Recognition Letters, **149**, pp. 109–119.

[57] Li L, Yan J, Wang H & Jin Y, 2020, *Anomaly detection of time series with smoothness-inducing sequential variational auto-encoder*, IEEE Transactions on Neural Networks and Learning Systems, **32(3)**, pp. 1177–1191.

[58] Liang X, Duan F, Bennet I & Mba D, 2020, *A sparse autoencoder-based unsupervised scheme for pump fault detection and isolation*, Applied Sciences, **10(19)**, pp. 6789.

[59] Liashchynskyi P & Liashchynskyi P, 2019, *Grid search, random search, genetic algorithms: a big comparison for NAS*, arXiv preprint arXiv:1912.06059.

[60] Lind PD, Gonzalez MC & Herrmann HJ, 2005, *Cycles and clustering in bipartite networks*, Physical Review E, **72(5)**, pp. 056127.

[61] Lindemann B, Maschler B, Sahlab N & Weyrich M, 2021, *A survey on anomaly detection for technical systems using LSTM networks*, Computers in Industry, **131**, pp. 103498.

[62] Liu F, Ting KM & Zhou Z, 2012, *Isolation-based anomaly detection*, ACM Transactions on Knowledge Discovery from Data, **6(1)**, pp. 1–39.

[63] Lookman, S & Nurcan S, 2015, *A framework for occupational fraud detection by social network analysis*, In: CAISE 2015 Forum, 1367.

[64] Lu W, Cheng Y, Xiao C, Chang S, Huang S, Liang B & Huang T, 2017, *Unsupervised sequential outlier detection with deep architectures*, IEEE Transactions on Image Processing, **26(9)**, pp. 4321–4330.

[65] Lundberg SM & Lee S, 2017, *A unified approach to interpreting model predictions*, Advances in Neural Information Processing Systems, **30**, pp. 4765–4774.

[66] Makki S, Assaghir Z, Taher Y, Haque R, Hacid M & Zeineddine H, 2019, *An experimental study with imbalanced classification approaches for credit card fraud detection*, IEEE Access: Advanced Software and Data Engineering for Secure Societies, **7**, pp. 93010–93022.

[67] Mehrotra KG, Mohan CK & Huang H, 2017, *Anomaly Detection Principles and Algorithms*, Springer International Publishing, New York.

[68] Meidianingsih Q, Erfiani & Sartono B, 2017, *The study of safe-level SMOTE method in unbalanced data classification*, International Journal of Scientific & Engineering Research, **8(5)**, pp. 1167–1171.

[69] Mienye ID, Sun Y & Wang Z, 2020, *Improved sparse autoencoder based artificial neural network approach for prediction of heart disease*, Informatics in Medicine Unlocked, **18**, pp. 100307.

[70] Misra S, Thakur S, Ghosh M & Saha SK, 2020, *An autoencoder based model for detecting fraudulent credit card transaction*, Procedia Computer Science, **167**, pp. 254–262.

[71] Mohamed S, Ejbali R & Zaied M, 2019, *Denoising autoencoder with dropout-based network anomaly detection*, In: The 14th International Conference on Software Engineering Advances, pp. 98–103.

[72] Molnar C, 2019, *Interpretable machine learning*, https://christophm.github.io/interpretable-ml-book/, Accessed: 2021-08-17.

[73] Narayana K, Venkata K & Prasad PVGD, 2021, *A hybrid intrusion detection system based on sparse autoencoder and deep neural network*, Computer Communications, **180**, pp. 77–88.

[74] Needham M & Hodler AE, 2020, *Graph algorithms*, 3, CA: O'Reilly Media, Inc, Sebastopol, United States of America.

[75] Negro A, 2021, *Graph-powered machine learning*, Manning Publications Co., NY, United States of America.

[76] Newman MEJ, 2010, *Networks: an introduction*, 2, New York, NY: Oxford University Press.

[77] Nguyen HD, Tran KP, Thomassey S & Hamad M, 2021, *Forecasting and anomaly detection approaches using LSTM and LSTM autoencoder techniques with the applications in supply chain management*, International Journal of Information Management, **57**, pp. 102282.

[78] Nikkel B, 2020, *Fintech forensics: Criminal investigation and digital evidence in financial technologies*, Forensic Science International: Digital Investigation, **33**, pp. 200908.

[79] Nonnemacher J, Kruse F, Schumann G & Gomez JM, 2021, *Using autoencoders for data-driven analysis in internal auditing*, In: Proceedings of the 54th Hawaii International Conference on System Sciences, pp. 5748.

[80] O'Malley AJ & Onnela J, 2014, *Topics in social network analysis and network science*, arXiv preprint arXiv:1404.0067.

[81] Oskarsdottir M, Ahmed W, Antonio K, Baesens B, Dendievel R, Donas T & Reynkens T, 2021, *Social network analysis for supervised fraud detection in insurance*, FRisk Analysis: an Official Publication of the Society for Risk Analysis.

[82] Pidhorskyi S, Adjeroh DA & Doretto G, 2020, *Adversarial latent autoencoders*, In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 14104–14113.

[83] Piraveenan M, Prokopenko M & Zomaya AY, 2008, *Local assortativeness in scale-free networks*, Europhysics Letters, **84(2)**, pp. 28002.

[84] Pourhabibi T, Ong KL, Kam BH & Boo YLY, 2020, *Fraud detection: a systematic literature review of graph-based anomaly detection approaches*, Decision Support Systems, **133**, pp. 113303.

[85] Preethi D & Khare N, 2021, *Sparse autoencoder drive support vector regression based deep learning model for predicting network intrusions*, Peer-to-Peer Networking and Applications, **14**, pp. 2419–2429.

[86] Principi E, Rossetti D, Squartini S & Pizza F, 2019, *Unsupervised electric motor fault detection by using deep autoencoders*, IEEE/CAA Journal of Automatica Sinica, **6(2)**, pp. 441–451.

[87] Ren H, Ye Z & Li Z, 2017, *Anomaly detection based on a dynamic Markov model*, Information Sciences, **411**, pp. 52–65.

[88] Riquelme F, Gonzalez-Cantergiani P, Molinero X & Serna M, 2018, *Centrality measures in social networks based on linear threshold model*, Knowledge-Based Systems, **140**, pp. 92–102.

[89] Sabetti L & Heijmans R, 2021, *Shallow or deep? Training an autoencoder to detect anomalous flows in a retail payment system*, Latin American Journal of Central Banking, **2(2)**, pp. 100031.

[90] Saia R, Carta S, Recupero DR & Fenu G, 2019, *Fraud detection for e-commerce transactions by employing a prudential multiple consensus model*, Journal of Information Security and Applications, **46**, pp. 13–22.

[91] Said Elsayed M, Le-Khac N, Dev S & Jurcut AD, 2020, *Network anomaly detection using LSTM based autoencoder*, In: Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, pp. 37–45.

[92] Saito T & Rehmsmeier M, 2015, *The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets*, PLoS ONE, **10(3)**, pp. 118432.

[93] Salahuddin M, Bari MDF, Alemeddine H, Pourahmadi V & Boutaba R, 2020, *Time-based anomaly detection using autoencoder*, In: 2020 16th International Conference on Network and Service Management (CNSM), IEEE, pp. 1–9.

[94] Sammut C & Webb G, 2011, *Encyclopedia of Machine Learning*, Springer Science & Business Media, New York.

[95] Sanchez OR, Repello M, Carrega A & Bolla R, 2021, *Evaluating ML-based DDoS detection with grid search hyperparameter optimization*, In: 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), pp. 402–408

[96] Sanni L & Nurcan S, 2015, *A framework for occupational fraud detection by social network analysis*, CAISE Forum, CEUR, 1367(29), pp. 1–8.

[97] Sanvido PHM, Kurtz GB, Teixeira CRG, Wagner PP, Leuck LP, Silveira MS, Tietzmann R & Manssour IH, 2021, *PeakVis: a visual analysis tool for social data and video broadcasts*, In: 2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 418–427.

[98] Schreyer M, Sattarov T, Borth D, Dengel A & Reimer B, 2018, *Detection of anomalies in large-scale accounting data using deep autoencoder networks*, arXiv preprint arXiv:1709.05254.

[99] Schutze H, Manning CD & Raghavan P, 2008, *Introduction to Information Retrieval*, Cambridge: Cambridge University Press, **39**, pp. 234–265.

[100] Skare M & Soriano DB, 2021, *How globalization is changing digital technology adoption: An international perspective*, Journal of Innovation and Knowledge, **6(4)**, pp. 222–233.

[101] Tun MT, Nyaung DE & Phyu MP, 0000, *Network anomaly detection using threshold-based sparse*, In: Proceedings of the 11th International Conference on Advances in Information Technology, 21, pp. 1–8.

[102] Valverde-Albacete FJ, Carrillo de Albornoz J & Pelaez-Moreno C, 2013, *A proposal for new evaluation metrics and result visualization technique for sentiment analysis tasks*, In: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, Berlin, Heidelberg, pp. 41–52.

[103] Van Vlasselaer V, Bravo C, Caelen O, Eliassi-Rad T, Akoglu L, Snoeck M & Baesens B, 2015, *APATE: A novel approach for automated credit card transaction fraud detection using network-based extensions*, Decision Support Systems, **75**, pp. 38–48.

[104] Wang S, Wang X, Zhang L, & Zhong Y, 2021, *Auto-AD: Autonomous hyperspectral anomaly detection network based on fully convolutional autoencoder*, IEEE Transactions on Geoscience and Remote Sensing, **60**, pp. 1–14.

[105] Wu P, Liu J & Shen F, 2019, *A deep one-class neural network for anomalous event detection in complex scenes*, IEEE Transactions on Neural Networks and Learning Systems, **31(7)**, pp. 2609-2622.

[106] Xie L, Pi D, Zhang X, Chen J, Luo Y & Yu W, 2021, *Graph neural network approach for anomaly detection*, Measurement, **180**, pp. 109546.

[107] Yin H, Benson AR & Leskovec J, 2019, *The local closure coefficient: a new perspective on network clustering*, In: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 303–311.

[108] Zandian ZK & Keyvanpour MR, 2019, *Feature extraction method based on social network analysis*, Applied Artificial Intelligence, **33(8)**, pp. 669–688.

[109] Zavrak S & Iskefiyeli M, 2020, *Anomaly-based intrusion detection from network flow features using variational autoencoder*, journal, IEEE Access, **8**, pp. 108346–108358.

[110] Zhang J & Luo Y, 2017, *Degree centrality, betweenness centrality, and closeness centrality in social network*, In: Proceedings of the 2017 2nd International Conference on Modelling, Simulation and Applied Mathematics, **132**, pp. 300-303.

[111] Zhou Y, Ren H, Li Z & Pedrycz W, 2021, *An anomaly detection framework for time series data: An interval-based approach*, Knowledge-Based Systems, **228**, pp. 107153.