PAYMENTS CANADA

SHALLOW OR DEEP? DETECTING ANOMALOUS FLOWS IN THE CANADIAN AUTOMATED CLEARING SETTLEMENT SYSTEM USING AN AUTOENCODER

2020-02-25

payments.ca

Shallow or deep? Detecting anomalous flows in the Canadian Automated Clearing and Settlement System using an autoencoder*

Leonard Sabetti[a] and Ronald Heijmans[b,c]

[a]*Bank of Canada*
[b]*Payments Canada*
[c]*De Nederlandsche Bank*

February 2020

Abstract

Financial market infrastructures and their participants play a crucial role in the economy. Financial or operational challenges faced by one participant can have contagion effects and pose risks to the broader financial system. Our paper applies (deep) neural networks (autoencoder) to detect anomalous flows from payments data in the Canadian Automated Clearing and Settlement System (ACSS) similar to Triepels et al. (2018). We evaluate several neural network architecture setups based on the size and number of hidden layers, as well as differing activation functions dependent on how the input data was normalized. As the Canadian financial system has not faced bank runs in recent memory, we train the models on "normal" data and evaluate out-of-sample using test data based on historical anomalies as well as simulated bank runs. Our out-of-sample simulations demonstrate the autoencoder's performance in different scenarios, and results suggest that the autoencoder detects anomalous payment flows reasonably well. Our work highlights the challenges and trade-offs in employing a workhorse deep-learning model in an operational context and raises policy questions around how such outlier signals can be used by the system operator in complying with the prominent payment systems guidelines and by financial stability experts in assessing the impact on the financial system of a financial institution that shows extreme behaviour.

*This project was carried out while Sabetti worked at Payments Canada. Sabetti and Heijmans can be reached at lsabetti@bank-banque-canada.ca and ronald.heijmans@dnb.nl, respectively. We are grateful for helpful comments from seminar participants at the Canadian Economics Association Annual Conference (June 2019), the Bank of Finland Payment and Settlement System Simulator workshop (August 2019), the joint conference by the Bank of England and King's College on "Modelling with Big Data and Machine Learning: Interpretability and Model Uncertainty" (November 2019) and the Payments Canada and Bank of Canada Joint Payments Research Symposium (November 2019). We also thank Timothy Aerts and Youcef Touizrar for providing useful comments on earlier drafts of the paper. The views expressed in the paper are solely those of the authors and do not represent the views of the authors' affiliations.

Keywords: Anomaly Detection, Autoencoder, Neural Network, Articial intelligence, ACSS, Financial Market Infrastructure, Retail Payments.
JEL classifications: C45, E42, E58.

1 Introduction

Financial market infrastructures (FMIs) play a crucial role in our economy. They facilitate the clearing and settlement of financial obligations between financial institutions and their clients. An FMI that does not function properly can pose significant risks to the financial system and become a potential source of contagion, especially, in times of increased market stress. Due to their importance, FMIs have to live up to high international standards, called the Principles for Financial Market Infrastructures (CPSS, 2012).

The automated clearing settlement system (ACSS) facilitates the clearing and settlement of primarily low-value electronic and paper-based payment items on behalf of financial institutions and their customers, underpinning much of day-to-day economic activity in Canada.[1] [2] [3] Because the majority of payments cleared in the ACSS are driven by recurring business and economic activity in Canada between member financial institutions, payment flows tend to follow well-behaved, partly predictable patterns that also correlate with economic conditions. As a result, sudden changes or diverging trends in these flows could represent unusual or suspect conditions for a financial institution (whether due to liquidity, operational or other issues). The emergence of financial problems for a participating financial institution (referred to hereafter as a participant) could result in decreased trust and trigger a sudden increase in the redemption of deposits, which is known as a bank run.

The aim of this paper is to detect anomalous payment flows in ACSS automatically by applying an unsupervised anomaly detection method. Anomaly detection aims to detect patterns in data that do not conform with expected behaviour (Chandola et al., 2009). Our data is unlabeled as we do not have previously identified examples. The unsupervised learning problem we study is a potentially challenging and highly relevant one in light of the legacy design of ACSS that does not make use of real-time risk controls relative to other payment systems such as large value payment systems or automated clearing houses. Our method can be used by: 1) the ACSS system operator to comply with

[1] Credit card payments as well as Interac e-transfer's P2P application clear through separate rails and settle in the large-value system.

[2] ACSS does not fall under the PFMIs but under the Bank of Canada Prominent Payment Systems (PPS), which are derived from the PFMIs. Therefore, we still link the risks to the PFMIs as a guideline known world wide.

[3] In 2018, the ACSS cleared over $6.4 trillion in value, representing roughly 7 billion individual payments.

the Canadian prominent payment systems guidelines; 2) prudential banking supervisors in detecting potential liquidity problems at an early stage; and 3) financial stability experts assessing the impact of a large financial institution showing strange behaviour in the financial system either by looking at historical data or by running simulations.

Our paper is closely related to Triepels et al. (2018). Their anomaly detection method tries to identify starting bank runs from TARGET2 transaction data.[4] There is a few differences in our approach, however, compared to Triepels et al. (2018). First, we implement the autoencoder approach for a retail payment system (ACSS) which processes different types of retail payment instruments. Liquidity problems for a participant could unfold through a single or multiple payment stream(s), such as those payment streams that offer faster funds availability or typically process relatively large dollar amounts. Second, we work with daily aggregate bilateral payment flows instead of intraday ones given the batch-entry nature of ACSS. Lastly, we also investigate the potential of a deeper autoencoder, which includes more than one hidden layer.

We add to the growing literature on outlier detection and finding potential risks in financial market infrastructures (FMIs). Petrunia et al. (2018) study a credit risk (collateral) management scheme for the Canadian ACSS designed to cover the exposure of a defaulting member. Heijmans and Heuver (2014) identify several elements that may show signs of liquidity stress at the participant level in a large value payment system. Berndsen and Heijmans (2020) develop different types of risk indicator based on FMI transaction data. Their indicators are linked to the PFMIs. In order to set the threshold to raise an alarm for medium or high risk, they use the univariate method developed by Timmermans et al. (2018). Triepels and Heuver (2019) develo a neural network to identify stress by banks months before they actually go bankrupt. Chakraborty and Joseph (2017) introduce machine learning in the context of central banking and policy analysis. Beutel et al. (2019) compare the out-of-sample predictive performance of different early-warning models for systemic banking crises using a sample of advanced economies covering the past 45 years. They compare a benchmark logit approach to several machine learning approaches recently proposed in the literature. They find that while machine

[4]TARGET2 is the largest real time gross settlement system in the European Union for euro denominated payments.

learning methods often attain a very high in-sample fit, they are outperformed by the logit approach in recursive out-of-sample evaluations.

Stock exchanges also make use of anomaly detection to detect trades in which brokers try to manipulate the stock prices in which they may have a strong position Kim and Sohn (2012), sometimes referred to as spoofing. Ferdousi and Maeda (2006) used anomaly detection to detect cases in which the brokers did not act in their customers' best interest. Also Credit card companies have an interest in detecting fraudulent use of their clients' credit cards as they may have to cover the losses of this fraudulent use. Ghosh and Reilly (1994) or Maes et al. (2002) use anomaly detection to detect such fraud at the level of individual clients.

Our paper is also linked to papers that look at different types of risks defined by the PFMIs such as systemic or credit risks. Li and Perez-Saiz (2018) develop a method to measure systemic risk across financial market infrastructures in Canada. Avdjiev et al. (2019) develop a new network centrality measure that can be interpreted in terms of a banking system's credit risk or funding risk. Boyd et al. (2019) construct theory-based measures of systemic bank shocks. The authors find, consistent with the result in the previous literature, that deposit insurance and safety net guarantees do not affect the probability of a systemic bank shock, but do increase the probability of a policy response to such a shock.

The organization of the paper is as follows. Section 2 describes the ACSS and its corresponding data. Section 3 describes the setup of the autoencoder. Section 4 assesses how well our autoencoder setup works to detect liquidity problems in ACSS data and evaluates the detection of outliers using in-sample outliers and a simulated bank run. Section 5 summarizes and provides insights in how to use it as an operator.

2 ACSS

2.1 ACSS Overview

The ACSS facilitates the clearing of many paper and electronic payment instruments, such as cheques, automated funds transfer (AFT) and electronic data interchange (EDI). Payments require clearing and settlement when a transfer of funds occurs between accounts held at different financial institutions, which are either direct participants or have tiering arrangements as indirect participants. These payment types include online bill payments, direct deposits such as payroll, business-to-business payments, pre-authorized debits such as mortgage payments and point-of-sale debit card purchases, as well as cross institution ABM cash withdrawals. On average, the ACSS clears about 30 million transactions per day worth roughly $30 billion.

By comparison, the large-value transfer system (LVTS) clears and settles roughly 30,000 payments per day, worth on average CA$150 billion. Wire payments in LVTS are typically large in value that require immediate finality and irrevocability. Wire payments often settle obligations from other FMIs outside LVTS or relate to cross-border transactions. Because LVTS operates under a collateralized hybrid DNS-RTGS model with real-time risk controls, a participant experiencing an outflow of payment requests would be required to obtain sufficient liquidity in advance, such as by pledging additional collateral to its settlement account held at the central bank. In contrast, payments exchanged and cleared through ACSS do not pass any similar real-time liquidity risk controls.

For historical reasons, including the accommodation of physical paper items that require manual processing, ACSS operates under a deferred net settlement model. Participants settle their multilateral net positions arising from the exchange of payments across multiple payment streams over a business day the following morning via wire payment in the LVTS. Credit risk materializes when a participant is incapable of meeting its final settlement obligation (a debit, or payable, multilateral net position); that is, at the time of settlement it is unable to deliver funds to its ACSS settlement account at the Bank of Canada in an amount equivalent to its final multilateral net debit position. Furthermore, in anticipation of settlement, participants may have already made funds available to recipient clients

from the defaulting participant following exchange the previous day.[5] Historically, a loss sharing agreement ensured certainty of settlement in case of a default; more recently, ACSS was designated a prominent payments system (PPS) and Payments Canada instituted a cover-one collateral pool made up of proportional contributions from participants to ensure that financial resources are available in case of a participant default. The level of the collateral pool is set to the 99^{th} largest observed MNDP over a rolling two-year time period. While multilateral netting lowers settlement risk exposure between participants, there is no practical way by which a participant can limit its credit exposure in the event of a default, nor can the operator implement risk controls to constrain exposures.

The next section describes some statistical features (or empirical regularities) of ACSS participant bilateral payment flows and settlement balances from recent history. Deviations from normal or usual patterns could indicate the emergence or development of impending risks, whether financial, operational or other. An anomaly detection tool may prove effective at uncovering such instances in light of historical statistical patterns that have emerged from the data. The application of an autoencoder approach is motivated by these considerations to inform and monitor ongoing risks.

2.2 ACSS Data and Statistics

While exchange of payment files compromising client payment instructions for each payment instrument occurs over the course of the day with three exchange window cutoff times, the ACSS application only receives data on payment flows overnight when entries are made by participant institution's back offices. As a result, from an operational perspective Payments Canada does not observe payment flows for a given day until the following morning. For this reason, we work with data aggregated to the daily level reflecting gross bilateral payments sent between participants, either by payment instrument or in aggregate. Figure 1 provides a visual depiction of average daily bilateral flows between participants; we observe that among the 11 participants, most of the value is skewed towards the larger ones.

Table 1 provides a high-level description of the data stemming from the ACSS for 2019. The largest

[5] Some payment instruments, such as AFT are governed by a two-hour funds availability rule; however, in general ACSS payments are not necessarily irrevocable, unlike wire transfers.

payment streams by value are AFT credit and debit. These are electronic credit push as well as debit pull payments, typically for direct deposits and pre-authorized debits. Furthermore, Payments Canada instituted a two-hour funds availability rule following the end of each exchange window. As a result, a client could send an AFT payment during the morning exchange window and the recipient would receive funds by close of business. This type of transfer has some features of a wire payment while being cheaper and easier to process from the sender's perspective. As a result, payment flows in AFT could particularly show early signs of dislocation for a participant experiencing stress or other challenges.

Table 1: ACSS facts in 2018.

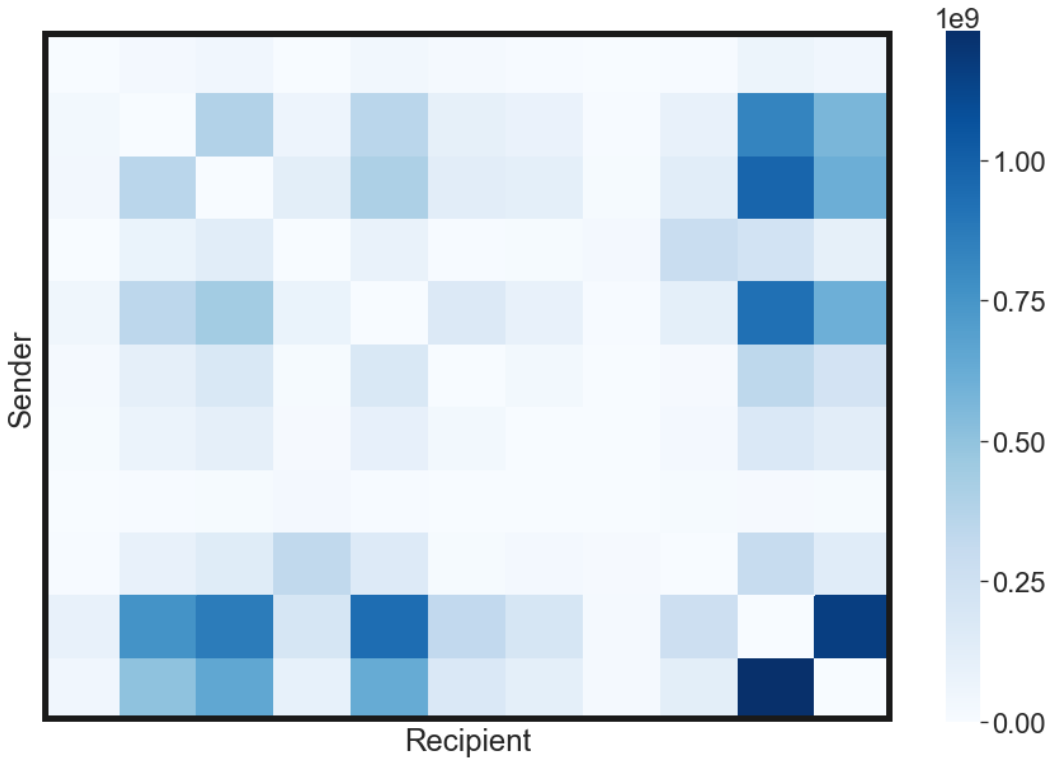
Number of participants	ACSS has 11 direct participants, (excluding the central bank)
Daily averages	ACSS processed a daily average of 30 , million payments, representing a daily average value of CA$ 35 billion
Median transaction value	CA$ 2000
Top 5 payment instruments by value	Automated funds transfer (AFT) debits and , credits (mainly direct deposits), paper-based and imaged cheques and remittances, point-of-sale debits and credits
Value share of largest five participants	90%

Source: Payments Canada.

Figure 1 plots historical daily multilateral net positions (MNPs) for select participants for illustration. While MNPs typically centre around zero, we observe some days with large outliers with reversion to the mean and little serial correlation. The empirical densities of settlement positions are highly skewed, which poses challenges for inferring VaR measures that rely on distributional or normal assumptions.

Overall, the data derive from normal and benign conditions, except for a short period of financial stress during the financial crisis; further, we are unaware of any major or prolonged operational incidents. In light of the data, our objective is to ascertain to what extent an algorithm such as a neural

Figure 1: Heat map of bilateral flows between participants.

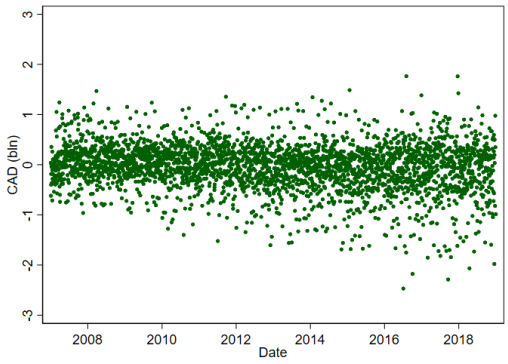


Note: Heatmap is based on average daily bilateral gross value sent in dollars from sending participant to recipient over the time period used in our training sample from 2007 to 2018. Diagonal elements are zero.

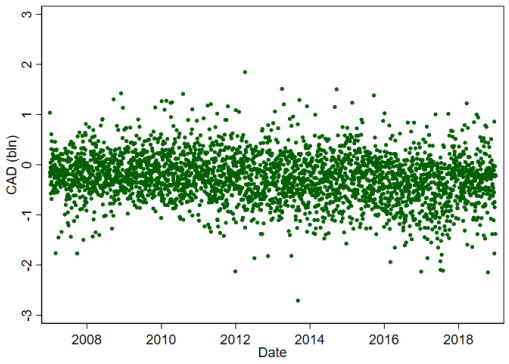
network autoencoder could detect anomalous payment flows in a relatively short period of time, alerting potential emerging risks concerning a participant. Timely signals from such a model could provide useful information to the operator as well as other financial-system stakeholders. In the next section, we describe our modeling approach in detail.

Figure 2: Daily individual participant settlement obligations of the six largest participants (referred to as A to F) in ACSS.

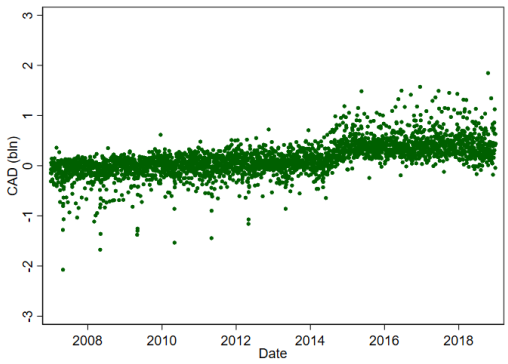
(a) Participant A.



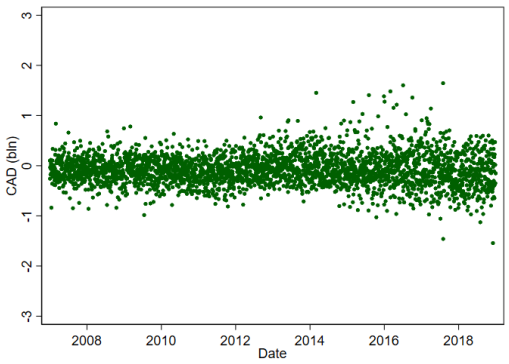
(b) Participant B.



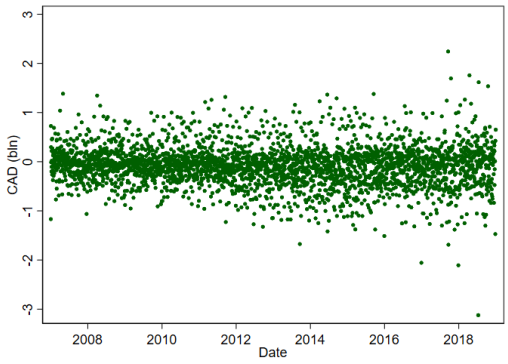
(c) Participant C.



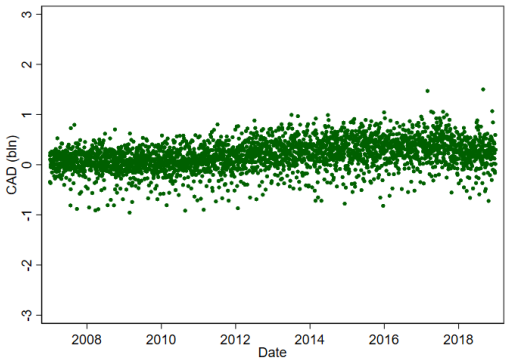
(d) Participant D.



(e) Participant E.



(f) Participant F.



Note: Scatterplots represent observed daily settlement obligations, or multilateral net positions, in ACSS for six participants. A positive value reflects a debit position while a negative value reflects a credit position.

3 Anomaly detection methodology

This section describes the main concepts of our methodology (autoencoder) for detecting anomalous liquidity flows from aggregate payment streams of the ACSS. The setup and used notation is similar to Triepels et al. (2018) and for a more in depth description of the methodology we refer to their paper.

3.1 ACSS payment flows as input of our methodology

In ACSS payment flows between (direct and group) clearers, which are aggregated by the system at daily level, can be observed. In contrast Triepels et al. (2018) use 15-minute aggregations as they were able to make use of the individual underlying transactions. However, the liquidity matrix will be the same in our setup and is defined as:

$$\mathbf{A}^{(t)} = \begin{bmatrix} a_{1,1}^{(t)} & \cdots & a_{1,11}^{(t)} \\ \vdots & \ddots & \vdots \\ a_{11,1}^{(t)} & \cdots & a_{11,11}^{(t)} \end{bmatrix} \tag{1}$$

Each element $a_{ij}^{(t)}$ of matrix $A^(t)$ is a daily flow clearer (or bank) i sends to clearer j at a certain day t. The diagonal elements of this matrix are the total amounts of liquidity a clearer in ACSS would send to itself, which is zero in our case.[6] As ACSS has eleven clearers $A^{(t)}$ will be an 11 by 11 matrix. Figure 1 in section 2.2 provided a visual representation of the elements in this matrix in the form of a heat map. While ultimately the payment system operator or overseer is concerned with the potential for unusually large settlement positions that could trigger a default, multilateral net positions are derived from these bilateral liquidity flows.

3.2 General concepts of an autoencoder

An autoencoder (AE) is an unsupervised artificial neural network that learns efficient data representations (encoding) by training the network to ignore signal noise. It consists of three types of layers:

[6] In the European interbank payment system TARGET2 for example there are flows between accounts of the same participant known as liquidity transfers, see e.g. Berndsen and Heijmans (2020).

1) the input, 2) one or more hidden (encoding) layers and 3) the output (decoding) layer. Figure 3 graphically depicts the architecture of a classic three-layered AE (Figure 3a) and a four-layered (Figure 3b) deep AE. A greater number of hidden-layers is associated with a deep(er) network. The hidden layer(s) in an autoencoder can be seen as the bottle neck through which the data has to go. The input and the output layer of an autoencoder have by definition the same number of nodes and represent the bilateral flows of the clearers in our analysis.

In the encoding stage the AE takes the input x and maps it to the hidden layer h using the encoder:

$$h = \sigma(Wx + b) \tag{2}$$

where σ is an element-wise activation function, W is a weight matrix and b is a bias vector. As activation function we use the functions tanh and rectified linear unit (ReLu). The starting point of the weights and bias terms is done randomly using an are updated iteratively by a process that is called back propagation.

After encoding the input data to the hidden layer, the decoder stage of the autoencoder takes the compressed representation and reconstructs it to $\hat{x}$ of the same shape as the input vector x. The decoder is defined as:

$$\hat{x} = \sigma'(W'h + b') \tag{3}$$

where σ' ,W' ,b' are the activation function, weights and bias terms of the decoder. However, the value can by certainly do not have to be the same as the ones of the encoder.

AEs are trained to minimise reconstruction errors or "loss". We use the mean squared error (MSE) of the difference between the input and output vector as the loss and is defined as:

$$\mathscr{L}(x, \hat{x}) = \|x - \hat{x}\|^2 \tag{4}$$

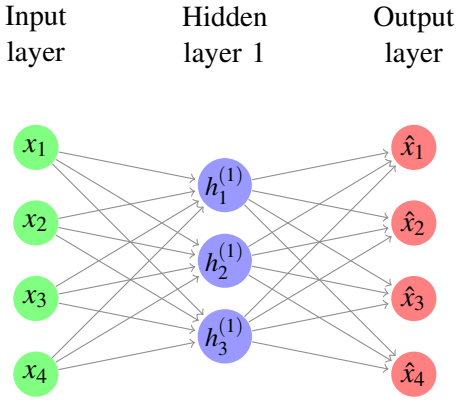
A crucial aspect in defining the optimal model is the number of nodes in the hidden layer(s) In the event that the number of neurons is too low, the AE may be unable to learn the main features from

the data and fails in reconstructing the compressed representation to the output layer. However, if the number of neurons is very large (close to the number of neurons in the input layer) it will not really learn any features from the data and just copy the input to the output layer. In order for an AE to learn something from the data it is essential that there are some general features that can be learnt. The model setup that shows the lowest reconstruction error on the validation data is considered the best performing model theoretically.

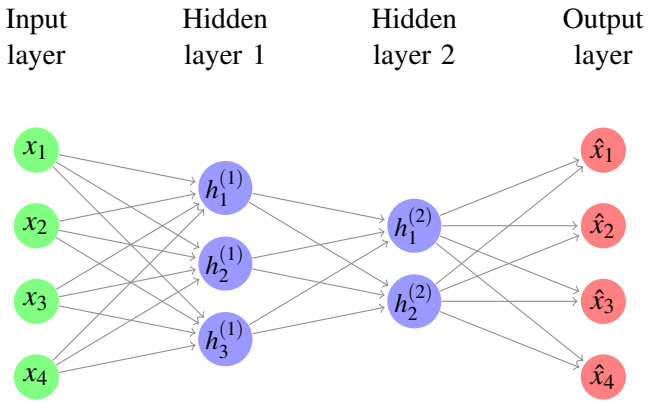
To avoid over-fitting, we split our data into training and validation sub-samples. The training set contains 80% of the data sample and the validation set 20%. Our model is trained using the training set and evaluated on the validation (or hold-out) set. Finally, the optimal model is evaluated on test data, which contains the outliers in our data sample. In order to minimize the potential for over-fitting to the validation set, we use k-fold cross validation where we partition the training set into five partitions of equal size. For a given model choice, we train the model in five separate instances using one of the distinct validation partitions. The final model score for a particular specification is based on an average across the five validation sets.

Figure 3: The architecture of an autoencoder. The input layer and hidden-layer(s) can also have a bias term.

(a) One hidden-layer.



(b) Two hidden-layers.



3.3 Detection and explanation of Anomalies

In line with Hawkins et al. (2002), Tóth and Gosztolya (2004), Dau et al. (2014) and Triepels et al. (2018) we also use an AE for anomaly detection. To assess the performance of a trained model, it is evaluated on validation data never seen before. A network that performs equally or almost as well on validation data as on training data is indicative of a model that generalizes well out-of-sample and will not overfit the training data. In our case, we would like our trained AE to have a low reconstruction error on data that is considered to reflect normal behaviour but a high reconstruction error for payment flows that are unusual (defined as anomalous). Consequently, we train an AE on historical data that excludes daily observations where any participant experienced an end of day settlement obligation

(or multilateral net debit position) that exceeded its 99^{th} empirical quantile. We draw our test data from our historical sample with the remaining observations containing bilateral flows generating a settlement position among the largest percentile for at least one participant.

The AE is trained on the training data set, which consists of daily liquidity flows covering roughly 10 years. In the training phase, the AE learns the main features of the aggregated payment flows between the participants. After the network has been trained it can be used to check whether new data is normal (similar to the ones it was trained on) or anomalous. In case the reconstruction error is high it is considered anomalous and normal otherwise. What is high or low has to be determined by the end user as there is no theoretical way of determining the cut off value between high and low. The expert in charge has to trade-off between the false negatives and false positives it is willing to accept. In the event that the reconstruction error value is chosen relatively low, it is likely that the number of false positives is high or when the value is chosen very high the number of false negative will increase.

As the reconstruction errors is the MSEs of the difference between the data in the input and output layer, it is possible to analyse who is causing the level of the reconstruction error. In case of a detected anomaly the expert in charge can see which flows between which banks are causing it. It is also possible to see whether it is for example an outflow from one bank to others or that one bank has increased inflows from all others.

4 Model selection and evaluation

This section describes the model performance for different setups of the neural network we implement. Table 2 shows the different setups for the one and two hidden layer AEs. For the one hidden layer network, the number of nodes ranges from 10 to 70 in increments of 10. The number of nodes in the two layer network ranges from 10 to 70 in increments of 10 for the first hidden layer and has 8, 16 and 32 nodes for the second hidden layer. The two hidden layer network attempts to obtain better generalizations than the single hidden layer setup. The number of nodes in both the input and the

output layer is 110.[7]

Table 2: Different model setup of for the one and two hidden layer AEs.

Number of layers	nodes in 1st layer	nodes in 2nd layer	activation functions
1	10,20,30,40, 50,60, 70	NA	ReLu, Tanh
2	10,20,30,40, 50,60, 70	8,16, 32	ReLu, Tanh

For the activation function we either use the ReLu or the tanh.[8] The ReLu activation function is used when the input data is normalized to be mean 0 with standard deviation 1 and as a result is treated as a continuous variable. The tanh activation function is used when we normalize the input data by maximum values to lie in the interval 0 and 1. We measure the model performance by examining the overall loss (mean square error, MSE) on the validation sets. The lower this value, the better the model performs.

Two important hyper-parameters when training a learning algorithm using stochastic gradient descent are the batch size and the number of epochs. The batch size reflects the number of observations passed through the network prior to updating the weights. We set the batch size to 16 throughout. The number of epochs defines the number of instances the entire training set will pass through the learning algorithm. A plot of the model error by epoch is known as a learning curve.

This remainder of this section will first look at the performance of the one and two hidden layer networks (section 4.1). Section 4.2 describes the best performing model and the trade offs in the selection of that model. Section 4.3 and 4.4 show the evaluation of the best model on the test set and on artificial data.

[7]There are 11 participants in ACSS. Each participant can make payments to ten other participants. There are no payments from a participant to itself.

[8]ReLu stands for rectified linear unit and is mathematically defined as $y = max(0,x)$. Tanh is hyperbolic tangent function.

4.1 One vs two hidden layers

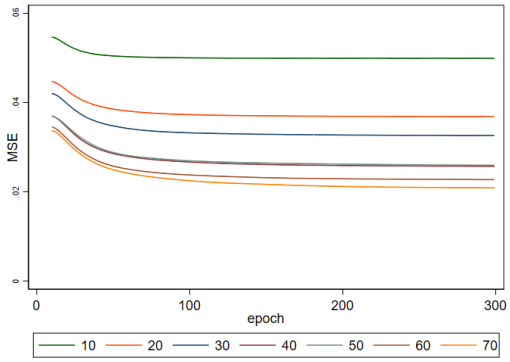
Figure 4 shows the model performance on training and validation sets in terms of mean square error (MSE) for the one hidden layer autoencoder using the ReLu (Figure 4a and 4b) and tanh (Figure 4c and 4d) activation functions. The number of nodes in the hidden layer varies between 10 and 70 and we train up to 300 epochs.

Overall, we observe that MSE scores from the training data tend to improve rapidly over the first 50 epochs but gradually stabilize with only marginal improvement as the number of epochs increases beyond 200. The MSE using the tanh activation function shows much lower values overall than the ReLu. The MSEs for the ReLu activation function range from roughly 0.01 to 0.05, and for the tanh from 0.0002 to 0.0018. If we only look at the performance on the training set suggests that the tanh outperforms the ReLu setup. The MSEs on the validation set tend to exhibit some variance as the number of epochs increases, but on average tend to decline as well, allthough they trend at permanently higher levels when compared to the training data results. In the case of the tanh results, there is greater variance in MSEs by epoch on the validation data and this tends to increase with the number of nodes, which may be caused by overfitting. Overall, the best performing one hidden layer setup for the tanh activation function has a smaller hidden layer while the ReLu performs better on a larger hidden layer. However, in evaluating model performance there is a trade-off between lower overall MSEs versus the variance of the MSEs with changing epochs.

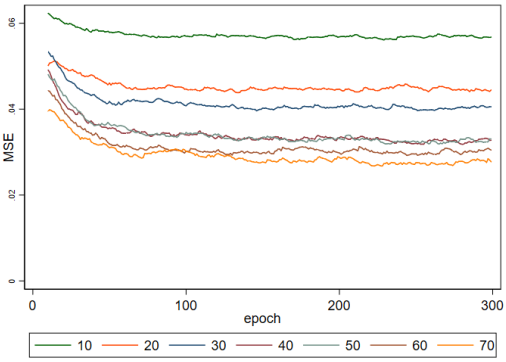
Figure 5 shows the model performance in terms of the MSE for the different setups of the two hidden layer (deep) autoencoder also using the ReLu and tanh activation function. Similar to the one hidden layer, the tanh also shows a better performance than the ReLu activation function. The performance on the validation data also shows higher variance than the training data. However, the variance of the validation data using both the tanh and ReLu activation function is lower than observed in the one hidden layer autoencoder. This suggests that the main advantage of utilizing a deeper network (two hidden layers) is to decrease the variance.

Figure 4: Model performance for different number of nodes in the hidden layer.

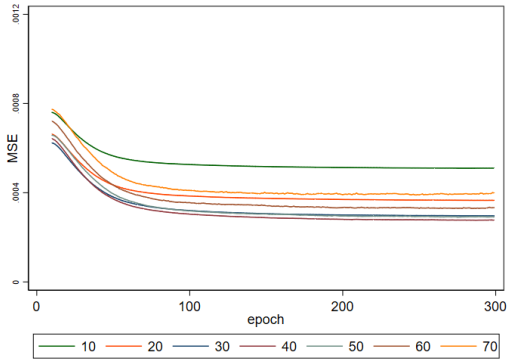
(a) ReLu training set.



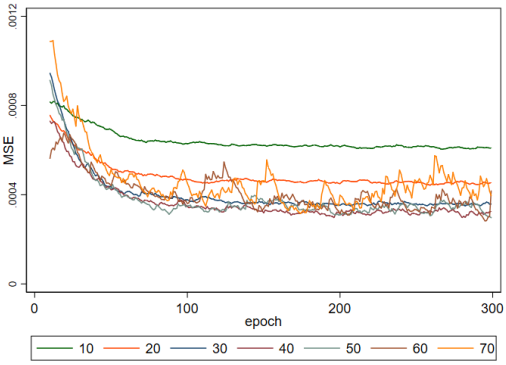
(b) ReLu validation set.



(c) Tanh training set.



(d) ReLu validation set.



4.2 Model selection

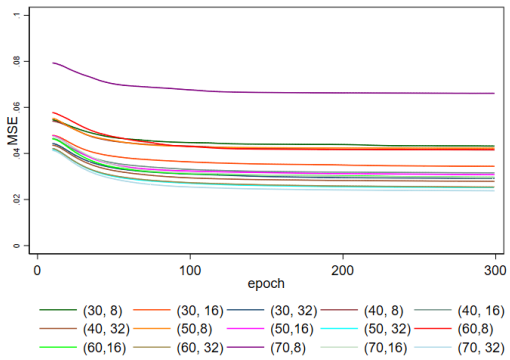
In order to select the best performing model an epoch has to be chosen, which we set to 200.[9] Model setups showing large variance in the MSE when increasing the epochs (of the validation set) do not seem to be good candidates for a model. This informed our modeling choice to constrain training the one hidden layer setup on 70 nodes or fewer.

Figure 6 shows the performance of the one and two hidden layer networks using the activation function ReLu (Figure 6a) and tanh (Figure 6b) respectively, with epoch 200. These figures allow for selecting the best model setup based on the MSE of the validation set. We have already seen in section 4.1 that the tanh activation function performs better than the ReLu based on its MSE values.

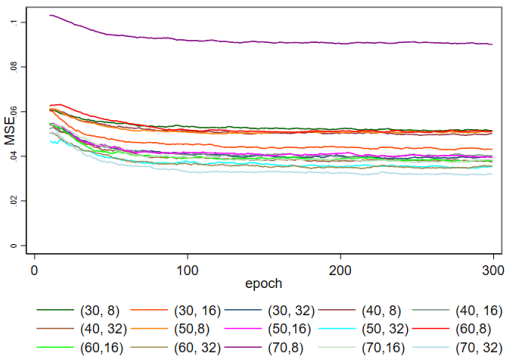
[9]However, the choice of epoch is somewhat arbitrary and a different epoch may lead to a difference in the selection of the best performing model.

Figure 5: Model performance for different number of nodes combinations in the two hidden layers.

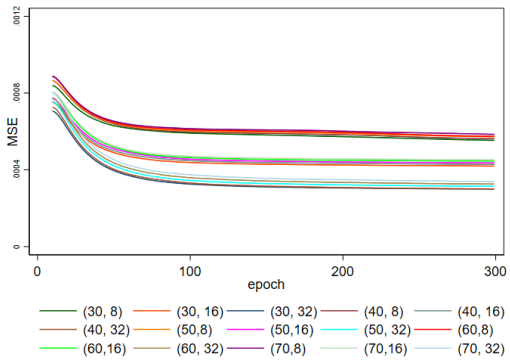
(a) ReLu training set.



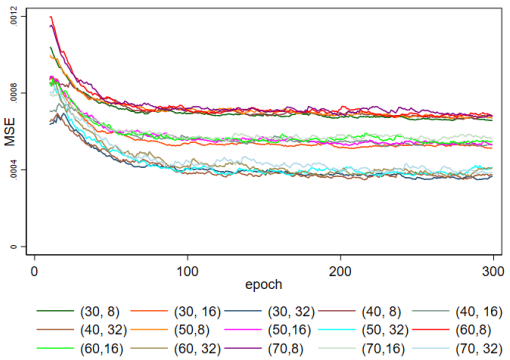
(b) ReLu validation set.



(c) Tanh training set.



(d) ReLu validation set.



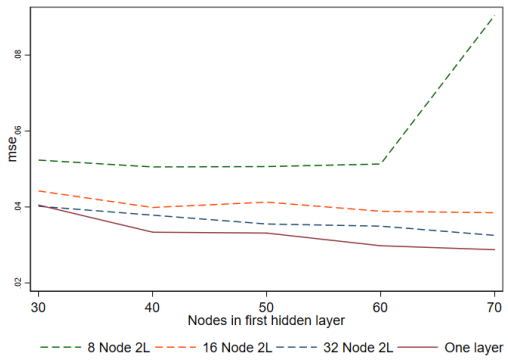
Therefore, the tanh would be the preferred activation function. Zooming into the difference between the one and two hidden layer setups, Figure 6b shows that the one hidden layer performance dominates the two hidden layer up until 70 nodes in the first layer. The figure shows that the one hidden layer approach with 40 nodes performs best overall, although not much better than a two hidden layer with 40 nodes in the first layer and 32 nodes in the second layer. Therefore, we will discuss the model performance on the test and artificial data in the next sections using the best one and two hidden layer setups.

4.3 Evaluation on test data

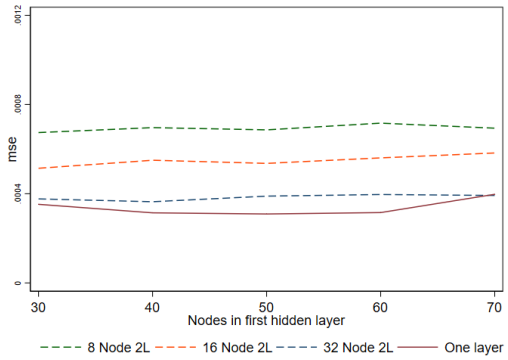
Figure 7 evaluates the optimal one layer and two layer networks derived in the previous section on the training and test data for both activation functions. The model error is displayed at the daily level.

Figure 6: MSE at epoch 200: one versus two hidden layers.

(a) ReLu.



(b) Tanh.



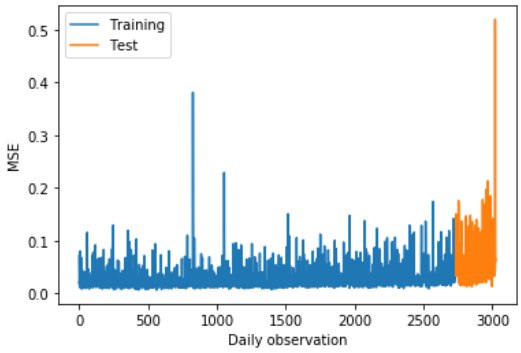
note: The solid line shows the performance (low MSE is better) of the autoencoder with one hidden layer (solid line). The dashed lines show the performance of the 2 hidden layers autoencoder for different number of nodes in the both the first (x-axis) and second hidden layer (different dashed lines).

The test data is drawn from historical daily observations where at least one participant exhibited a settlement position greater than its ex post 99th quantile. While we observe that the daily reconstruction error displays, on average, a level shift higher on the test data than on the training data along with larger spikes, there is no clear demarcation between the two reconstruction error sets. In other words, there are daily observations from the training data where the reconstruction error is greater than those observed from the test data. Furthermore, despite the fact that the neural networks were optimized to fit the training data, we do observe outliers at the daily level where the network does a poor job. The two hidden layer autoencoder has an overall lower MSE and fewer spikes compared to the one layer autoencoder.

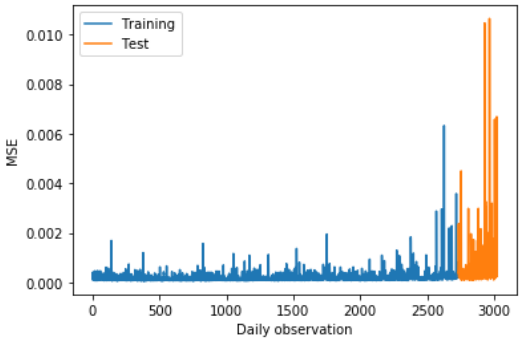
The outlier scores (MSE) on the test data are quite similar between the networks. Therefore, given the lower overall MSE levels and fewer spikes, the deeper network will pick up outliers more quickly. This may be due to the fact that a deeper network is more capable of generalizing the data. A deeper network can also learn correlations between flows of participants. This may also be the reason why some normal days in the training days show high MSE values, as there may be uncommon outflow and/or inflow combinations between participant pairs. The tanh activation function shows a better performance on our data than the ReLu.

Figure 7: MSE levels of the training and test set using the optimal autoencoder.

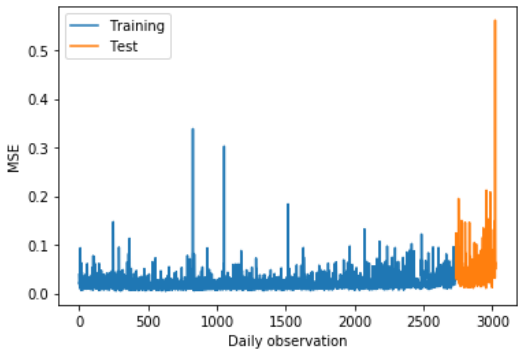
(a) ReLu 1 hidden layer.



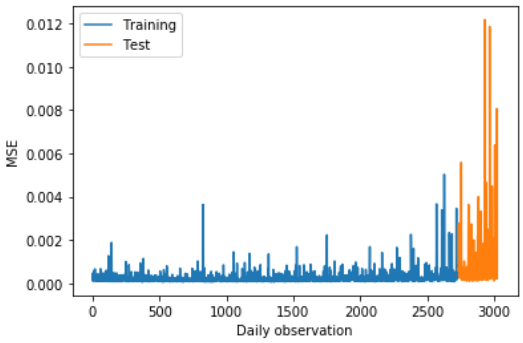
(b) Tanh 1 hidden layer.



(c) ReLu 2 hidden layers.



(d) Tanh 2 hidden layers.



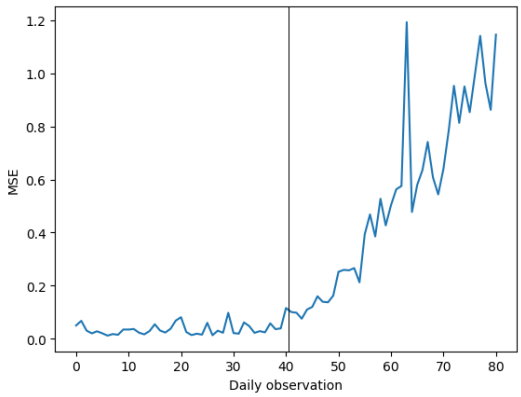
note: The blue bars show the MSE levels for the different days in the training set. The orange bars shows the MSE levels of the test set which contains outliers.

4.4 Evaluation on artificial data

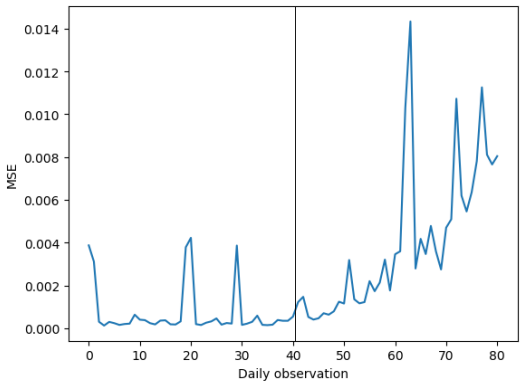
During the time period under study, there have not been any defaults in ACSS nor any substantive periods of financial stress for any participant. Therefore, we are not able to test whether a scenario such as a bank run would be flagged by our autoencoder. To combat that, we simulate an artificial bank run and evaluate the performance of our chosen optimal autoencoders; defined as a scenario where a participant's bilateral outflows increase proportionally to all other participants by $\frac{1}{2}$ standard deviation on a weekly basis; similarly, its bilateral inflows from all other participants decrease at the same aggregate rate. As a result, after one month of simulation the participant is exhibiting gross bilateral inflows and outflows of 2 standard deviations from actual levels. The "bank run" starts at business day 41 in the graph, which is the start of the ninth business week. Figure 8 displays the reconstruction error when applying each two-layer neural network to one month of normal data followed by one month of artificial data that intends to mimic a bank run. We observe that the reconstruction error rises rapidly in both cases, but the potential detection of anomalous activity occurs more quickly in the case of the ReLu activation function. This suggests that a bank run in which the outflows increase consistently will be picked up by our autoencoder setup after a few days. Of course, bank runs will have different speeds at which the outflow increase, and may even have gaps in the outflow due to insufficient liquidity, see Triepels et al. (2018).

Figure 8: MSE levels of a simulated bank run of a large participant in ACSS.

(a) ReLu.



(b) Tanh.



5 Conclusions

This paper investigates whether a neural network (autoencoder) can detect anomalous flows in the Canadian retail payments data (ACSS). It describes the process of setting up the best model and the choices that have to be made by the system operator. It is a first attempt to apply a neural network model to these data. The most optimal setup of the autoencoder, i.e. the setup that has the lowest MSE of all models on the validation set, is a one hidden layer autoencoder with 40 nodes using the tanh activation function. However, we observe multiple incidents of large MSEs on the training data indicating potential anomalous flows that are in fact normal (false positives). This may be due to several flows between participants being unusual, but not exceptionally so. In addition, on the test data which we consider to be a sample of all outlier days, we observe cases of lower than average MSEs (false negatives). In contrast, on the training sample we observe fewer spikes (false positives) in the two hidden layer setup with 40 nodes in the first hidden layer and 32 nodes in the second. This suggests that the two hidden layer setup better generalizes the flows between the participants. In particular, on the cross correlation between flows of different participants. Even though the two hidden layer autoencoder is not the best model looking at the MSE of the validation set, it does a better job in detecting the type of outliers we aim for in this paper.

The simulated bank run shows that if there are continuously increasing outflows from the problem bank to all other participants, the algorithm will show substantially increased MSEs within a week. One direction for improving the model would be to investigate whether the sequence or ordering of the data (e.g. weekly or monthly patterns) matter. Recurring peak values by day of the week or month could be accounted for, which might help reduce the number of incorrectly identified outliers. Examples of such models would be a Long Short-Term Memory (LSTM) or a Gated Recurrent Units (GRU).

To conclude, we summarize how the output of the types of models we work with in this paper can be used by the operator to monitor unusual flows in its system. In the case an outlier is detected by the model, it could trigger an analysis on the underlying vcauses; for instance, whether the outlier was caused by: 1) just one bank having a very large in or outflow; 2) one bank having unusual in and

outflows to many other banks; or 3) many banks having unusual in and outflows to many other banks. In addition, the occurrence of an outlier or multiple outliers could warrant an investigation so as to untangle whether the underlying cause was driven by financial stress, operational or otherwise as well as monitoring activity closely in the short-term. Depending on the application, a single outlier or a sequence of outliers can be shown to the operator. In some cases, the outlier could be linked to an operational problem known by the operator. The output can also be used by financial stability experts in assessing the impact of strange behaviour or liquidity problems at the level of one or more financial institutions participating in ACSS. This can either be done by looking at historical data or by running simulations.

References

Adams, M., Galbiati, M., and Giansante, S. (2010). Liquidity costs and tiering in large-value payment systems. *Bank of England Working Papers*.

Avdjiev, S., Giudici, P., and Spelta, A. (2019). Measuring contagion risk in international banking. *Journal of Financial Stability*, 42:36 – 51. Financial Services Indices, Liquidity and Economic Activity.

Berndsen, R. and Heijmans, R. (2020). Near real-time monitoring in a real-time gross settlement system: A traffic light approach. *Journal of Risk*, 22:39–64.

Beutel, J., List, S., and von Schweinitz, G. (2019). Does machine learning help us predict banking crises? *Journal of Financial Stability*, 45:100693.

Bottou, L. (2004). Stochastic Learning. In *Advanced lectures on machine learning*, pages 146–168. Springer.

Boyd, J. H., Nicolò, G. D., and Rodionova, T. (2019). Banking crises and crisis dating: Disentangling shocks and policy responses. *Journal of Financial Stability*, 41:45 – 54.

Chakraborty, C. and Joseph, A. (2017). Machine learning at central banks. *Bank of England Working Paper*, (674).

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys (CSUR)*, 41(3):15:1–15:58.

CPSS (2012). Principles for Financial Market Infrastructures: Disclosure Framework and Assessment Methodology. *Bank for International Settlements*.

Dau, H. A., Ciesielski, V., and Song, A. (2014). *Anomaly Detection Using Replicator Neural Networks Trained on Examples of One Class*, pages 311–322. Springer International Publishing, Cham.

Ferdousi, Z. and Maeda, A. (2006). Unsupervised Outlier Detection in Time Series Data. In *22nd International Conference on Data Engineering Workshops (ICDEW'06)*, pages 51–56. IEEE.

Ghosh, S. and Reilly, D. (1994). Credit card fraud detection with a neural-network. In *System Sciences, 1994. Proceedings of the Twenty-Seventh Hawaii International Conference on*, volume 3, pages 621–630. IEEE.

Hawkins, S., He, H., Williams, G., and Baxter, R. (2002). Outlier Detection Using Replicator Neural Networks. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 170–180. Springer.

Heijmans, R. and Heuver, R. (2014). Is this Bank Ill? The Diagnosis of Doctor TARGET2. *Journal of Financial Market Infrastructures*, 2(3):3–36.

Kim, Y. and Sohn, S. Y. (2012). Stock fraud detection using peer group analysis. *Expert Systems with Applications*, 39(10):8986–8992.

Li, F. and Perez-Saiz, H. (2018). Measuring systemic risk across financial market infrastructures. *Journal of Financial Stability*, 34:1–11.

Maes, S., Tuyls, K., Vanschoenwinkel, B., and Manderick, B. (2002). Credit Card Fraud Detection Using Bayesian and Neural Networks. In *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, pages 261–270.

Petrunia, R., Sabetti, L., and Voia, M. (2018). Tail risk in the automated clearing settlement system (acss). *Payments Canada Discussion Papers*, (7).

Timmermans, M., Heijmans, R., and Daniels, H. (2018). Cyclical patterns in risk indicators based on financial market infrastructure transaction data. *Quantitative Finance and Economics*, 2(3):615–636.

Tóth, L. and Gosztolya, G. (2004). Replicator neural networks for outlier modeling in segmental speech recognition. *Advances in Neural Networks*, pages 996–1001.

Triepels, R., Daniels, H., and Heijmans, R. (2018). Detection and explanation of anomalous payment behaviour in real-time gross settlement systems. In *Enterprise Information Systems. ICEIS 2017. Lecture Notes in Business Information Processing*, volume 321. Springer, Cham.

Triepels, R. and Heuver, R. (2019). Liquidity stress detection in the european banking sector. *DNB Working Paper*, (642).

Werbos, P. J. (1982). Applications of Advances in Nonlinear Sensitivity Analysis. In *System modeling and optimization*, pages 762–770. Springer.