Financial Fraud Detection with Improved Neural Arithmetic Logic Units

Daniel Schlör[1(✉)], Markus Ring[2], Anna Krause[1], and Andreas Hotho[1]

[1] University of Wuerzburg, Würzburg, Germany
{schloer,anna.krause,hotho}@informatik.uni-wuerzburg.de
[2] University of Applied Sciences and Art Coburg, Coburg, Germany
markus.ring@hs-coburg.de

Abstract. Domain specific neural network architectures have shown to improve the performance of various machine learning tasks by large margin. Financial fraud detection is such an application domain where mathematical relationships are inherently present in the data. However, this domain hasn't attracted much attention for deep learning and the design of specific neural network architectures yet. In this work, we propose a neural network architecture which incorporates recently proposed Improved Neural Arithmetic Logic Units. These units are capable of modelling mathematical relationships implicitly within a neural network. Further, inspired by a real-world credit payment application, we construct a synthetic benchmark dataset, which reflects the problem setting of automatically capturing such mathematical relations within the data. Our novel network architecture is evaluated on two real-world and two synthetic financial fraud datasets for different network parameters. We compare our proposed model with several well-established classification approaches. The results show that the proposed model is able to improve the performance of neural networks. Further, the proposed model performs among the best approaches for each dataset.

Keywords: Arithmetic neural networks · iNALU · Financial data · Fraud detection

1 Introduction

Traditional machine learning models often rely on feature engineering with expert knowledge. In contrast, one benefit of neural networks is seen in the ability of the network to find and combine relevant features. Certain network architectures have proven to be well suited for different tasks capturing the characteristics of the underlying data exceptionally well (e.g. Convolutional Neural Networks for images or Recurrent Neural Networks for sequences). However, neural network architectures specifically tailored to financial feature dependencies and mathematical relationships have not been well-studied yet.

Problem. The presence of mathematical relationships between features is a well-known fact in many financial settings [2,11]. For example, PaySim [11] is

V. Bitetta et al. (Eds.): MIDAS 2020, LNAI 12591, pp. 40–54, 2021.
https://doi.org/10.1007/978-3-030-66981-2_4

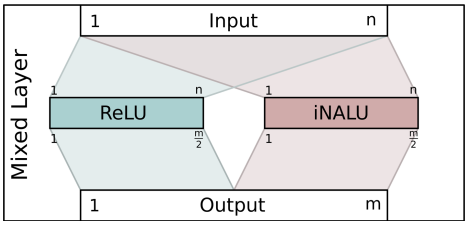

Fig. 1. Our proposed mixed layer consisting of ReLU and iNALU neurons

a mobile money simulator for fraud detection which generates bank transfers including the transmitted amount, the old account balance and new account balance as features. In this setting, these three features are highly related to each other in a mathematical sense. While neural networks are well suited for many complex data mining tasks, they often have problems with the calculation of even basic mathematical operations [18].

Objective. Although such relationships are not always directly related to the downstream-task which the machine learning model is applied to, a neural network architecture capable of capturing such relations, is able to model the data inherently better. A neural network architecture recently proposed to capture such relationships is the Improved Neural Arithmetic Unit (iNALU) [14].

In this work, we want to examine the research question if introducing iNALUs can improve the performance of neural networks on the task of financial fraud detection.

Contribution. First, we provide a short summary of existing financial fraud datasets, which are publicly available. Then, we explore the potential of enhancing neural networks using iNALUs for financial fraud detection. Since financial fraud detection is more complex than modelling mathematical relationships in the data, we propose a novel *Mixed Layer* architecture (see Fig. 1) which incorporates Rectified Linear Units (ReLUs) as general purpose neurons and iNALU neurons to capture arithmetic relationships within the data.

All approaches are evaluated on two synthetic and two real-world fraud detection datasets from the financial domain. Our findings suggest that incorporating iNALU layers significantly improves the performance on several datasets in comparison to vanilla feed forward networks with comparable network structures. Compared to several commonly used standard classifiers, our model is among the best models on each dataset and on average yields the best results.

Structure. The paper is structured as follows: The next section describes related work. Section 3 introduces the datasets used in our experiments. Our model is introduced in Sect. 4 and the experiments are described in Sect. 5. Sect. 6 discusses the results and Sect. 7 finally concludes the paper.

2 Related Work

Since this work aims to improve financial fraud detection using iNALU, this section reviews related work about modelling arithmetic relations in neural networks and financial fraud detection in general.

2.1 Modelling Arithmetic Relations

Kaiser and Sutskever [9] propose neural GPU for solving simple algorithmic tasks. Neural GPU is based on a type of Recurrent Neural Network and is able learn long binary summations and multiplications. This architecture generalizes well for long numbers, but is limited to four input symbols. Similar approaches are proposed by Kalchbrenner et al. [10] and Freivalds and Liepins [8].

Chen et al. [6] present another approach which is able to model arithmetic relations. They use reinforcement learning to solve mathematical operations such as summation, subtraction, multiplication or division. However, their approach requires the mathematical operation as an additional input to the network.

Neural Arithmetic Logic Unit (NALU) is a neural architecture designed to perform mathematical operations which is proposed by Trask et al. [18]. The NALU is not limited to certain input symbols and does not require the mathematical operation as input. The special characteristics of NALU are the restriction of the weights to the interval $[-1, 1]$ and the realization of multiplication and division in logarithmic space. The authors show in an experimental study that NALU generalizes better than traditional neurons for extrapolation tasks and achieves good results for various downstream tasks. However, NALU has some limitations as discussed in [12] and [14]. Schlör et al. [14] proposes an improved version of NALU called iNALU allowing multiplication of negative numbers and stabilization of the training.

In this work, we want to learn underlying relations within financial data implicitly. We do not want to limit ourselves to certain input symbols or explicit op-codes as input for modelling real-world datasets which generally don't fit these requirements. Therefore, we choose iNALU as basic architecture since it fits these requirements best.

2.2 Financial Fraud Detection

Financial fraud detection is very diverse and may appear in different areas such as mobile payments, credit card misuse or in ERP systems. Often, fraud represents only a very small proportion of transactions in these areas. Consequently, many financial fraud detection methods are based on anomaly detection. A comprehensive review of anomaly detection methods is given in [4]. Candola et al. [4] categorize existing approaches for anomaly detection based on their techniques and applications. Thereby, fraud detection is one of the applications being investigated. Chalapathy and Chawla [3] provide a more recent survey of deep learning for anomaly detection which among other things addresses the topic fraud detection. The authors describe existing approaches for credit card fraud detection,

mobile cellular network fraud detection, insurance fraud and healthcare fraud. This survey shows that various network architectures such as Auto-Encoders, Generative Adversarial Networks, CNNs, Restricted Boltzmann Machines, or RNNs are used for fraud detection.

Baader and Krcmar [1] present an approach for fraud detection in purchase-to-pay business processes in ERP systems. The authors combine a red flag approach with process mining and try to reduce the number of false positives. Red flags are hints or indicators for fraudulent behavior and show that something irregular has happened. Process mining entails discovering, monitoring, and improving real processes through the extraction of information from event logs of IT systems. Schreyer et al. [15] use deep autoencoders to detect anomalous journal entries in ERP systems. Journal entries are standard accounting transactions which affect multiple accounts. The authors calculate an anomaly score for each journal entry which takes into account the frequency of the values and the reconstruction error of the autoencoder. Then, anomalies are detected based on a scoring systems in combination with a user-defined threshold. Schreyer et al. evaluate their work on two private data sets which encompass the entire population of journal entries of a single fiscal year.

Many works investigate the suitability of machine learning methods for credit card misuse. Wang et al. [19] evaluate Random Forest, Support Vector Machines and Capsule Networks for credit card fraud detection. The authors use a private dataset about online credit card transactions and extract several features from each transaction. The features consider customer specific historical information like the average transaction amount over the last days. Similarly, Maes et al. [13], Shen et al. [16], or Sun and Vasarhelyi [17] evaluate different neural network architectures for credit card fraud detection.

In contrast to existing approaches our work aims at evaluating the benefit of neurons which model arithmetic relations rather than achieving the best performance on the downstream tasks or specific datasets. Instead of relying on thorough feature engineering (cf. [1,19]) or unsupervised anomaly-detection methods (cf. [15]), we explicitly include neurons (iNALU) which model arithmetic relations in our neural network architectures and evaluate in a supervised classification setting. Financial data sets often cannot be shared due to privacy concerns. As a consequence, many approaches are evaluated on non-publicly available data sets and therefore lack reproducibility and the availability for other researchers to conduct follow up studies. To address this challenge, we chose four publicly available data sets in our evaluation.

3 Datasets

In this study, we use two synthetic and two real-world datasets for financial fraud detection. All datasets are highly imbalanced and contain only few fraud cases. The main dataset characteristics are summarized in Table 1.

Table 1. Main characteristics of the datasets

Dataset	Features	Samples	Benign	Fraud	Fraud-ratio	Origin
Credit	4	100 000	98 967	1 033	0.010	Synth.
PaySim	11	6 362 620	6 354 407	8 213	0.001	Synth.
CCFraud	30	284 807	284 315	492	0.002	Real
IEEE-CIS Fraud	431	590 540	569 877	20 663	0.035	Real

3.1 Credit Payment

Peer-to-Peer credits become more and more popular and open new opportunities for financial fraud. One possibility for fraud is incorrect interest-calculation. We created the synthetic credit payment dataset referred to as *Credit*, which reflects such kind of fraud.

Each data point contains the following attributes: credit sum in month x (CS_x), interest rate (IR), payment rate (PR), credit sum in month $x+1$ (CS_{x+1}), label. The mathematical relationship between the attributes is defined as follows:

$$\mathrm{CS}_{x+1} = \mathrm{CS}_x + \frac{\mathrm{CS}_x \cdot \mathrm{IR}}{12} - \lambda \cdot \mathrm{PR} \tag{1}$$

With a probability of 99% the credit sum is calculated correctly ($\lambda = 1$) according to Eq. 1, but with a probability of 1% we simulate a fraudulent calculation of the remaining credit by only reducing the new credit sum by 95% of the payed rate ($\lambda = 0.95$). Overall, each instance contains the columns $\mathrm{CS}_{x+1}, \mathrm{CS}_x, \mathrm{IR}, \mathrm{PR}$ and the label *is Fraud*. For each instance the features are drawn randomly from a uniform probability distribution ($\mathrm{CS}_x \sim \mathcal{U}(0, 10\,000)$, $\mathrm{IR}_x \sim \mathcal{U}(0, 0.5)$, $\mathrm{PR}_x \sim \mathcal{U}(0, 5\,000)$) with the constraint, that the credit is not overpaid i.e. $\mathrm{CS}_{x+1} \geq 0$. In contrast to PaySim, fraudulent and benign transactions are modeled after the same probability distributions (or user-profiles) which means, the machine-learning model has to capture the mathematical relationship to predict correctly if a transaction is fraudulent. The dataset consists of 100 000 instances having 1033 fraudulent and 98 967 benign transactions. We make the dataset and the code publicly available[1].

3.2 PaySim

Paysim [11] is a multi-agent based simulator which can model financial mobile money transactions with fraudulent behavior. The simulator is based on simple mathematical statistics and is able to generate five types of transactions (*cash-in*, *cash-out*, *debit*, *payment* and *transfer*). From unpublished real transaction data, Lopez-Rojas et al. extracted several characteristics of the data and modelled them as user profiles. These characteristics include for example aggregated

[1] https://github.com/daschloer/inalu-finfraud.

transaction counts, amounts or initial account balances. Based on these user profiles, new synthetic data was generated.

Each transaction instance is described by the attributes old and new balance for both source and destination account, type of transaction, and amount of transferred money. Each row also includes a rule-based red-flag indicator for fraud, which is omitted for our experiments. We omit the account names, since they are picked randomly during the simulation without any underlying user-network structure or memory of fraudulent and non fraudulent users.

A dataset created with PaySim has been made publicly available on Kaggle[2]. The PaySim dataset contains 6 362 620 transactions of which 6 354 407 (99.871%) are benign and 8 213 (0.129%) are fraud.

3.3 Credit Card Fraud Detection

The Credit Card Fraud Detection (CCFraud) [7] dataset is a real data set of credit card transactions of European cardholders in the year 2013. The dataset contains normal and fraudulent credit card transactions. For anonymization, the authors applied principal component analysis (PCA) to all features except *time* and *amount* resulting in 28 continuous PCA transformed attributes. The feature *time* contains the seconds elapsed since the first transaction in the dataset and the feature *amount* the amount of money transferred. It contains 284 807 samples from which 492 are fraudulent.

3.4 IEEE-CIS Fraud Detection

The IEEE-CIS Credit Card Fraud Detection data set was released as part of a Kaggle[3] data science competition in 2019. This dataset is provided by Vesta Corporation and contains feature-rich representations of normal and fraudulent credit card transactions. It contains identity features such as network connection information and digital signature information about the device as well as transaction features, e.g., product information, card information, masked undisclosed counting and match features, address and distance features as well as the amount and time-delta from a reference date. Many features are not described in detail and remain deliberately unclear for privacy and contract reasons. The dataset contains 590 540 transactions, of which 20 663 are labeled as fraud (3.5%) and 569 877 as benign (96.5%).

4 Our Model

This section first describes the iNALU architecture shortly and then proposes our novel neuronal network architecture for financial fraud detection tasks.

[2] https://www.kaggle.com/ntnu-testimon/paysim1.
[3] https://www.kaggle.com/c/ieee-fraud-detection/data.

4.1 INALU

Improved Neural Arithmetic Unit (iNALU) [14] is a neuron specially designed for mathematical operations and it's architecture is shown in Fig. 2.

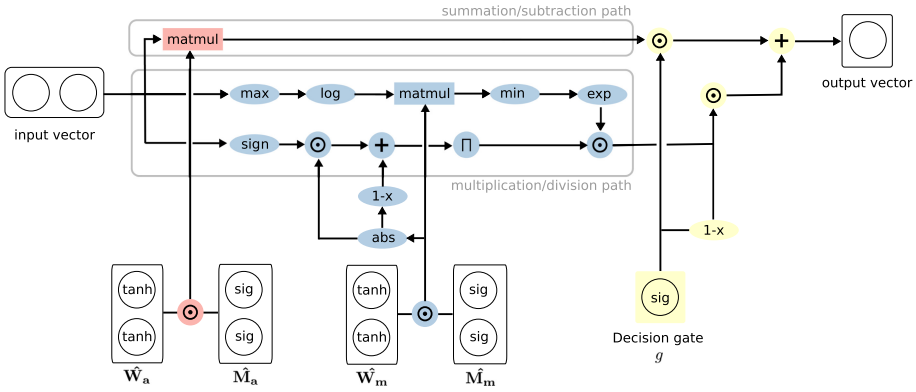

Fig. 2. iNALU architecture [14]

iNALU has one path for summation/subtraction and one path for multiplication/division. A decision gate regulates the influence of both paths. Each path has two weights matrices $\hat{W}_a$ and $\hat{M}_a$ (respectively $\hat{W}_m$ and $\hat{M}_m$). In a first step, tanh respectively sigmoid activation functions are applied to the weight matrices and then weight matrices are multiplied with each other element by element. Therefore the resulting weights are scaled to the interval $[-1, 1]$ with three plateaus at -1, 0 and $+1$. If the resulting weight is -1, iNALU performs subtractions respectively divisions. If the resulting weight is 0, iNALU ignores the input signals. If the resulting weight is $+1$, iNALU performs summations respectively multiplications. Multiplications and divisions are calculated by additions and subtractions of input signals in the logarithmic space. In addition, iNALU uses max and min functions to prevent values which are too large and calculates the resulting sign of the multiplicative path separately to allow the multiplication and division of negative numbers. We refer to Schlör et al. [14] for a more detailed description of iNALU and a discussion of stability and precision in several experimental scenarios.

4.2 Improved Neural Network Architecture

Our model relies on iNALU which by design is able to model simple arithmetic relationships. More complex relationships can be learned stacking multiple layers of iNALUs to a deeper network. However, even in the financial domain, real-world datasets generally contain more than mathematical relationships. Therefore, we propose a model with each layer containing 50% general purpose non-linear hidden units (to be precise ReLUs) and 50% iNALUs. In particular, the iNALU

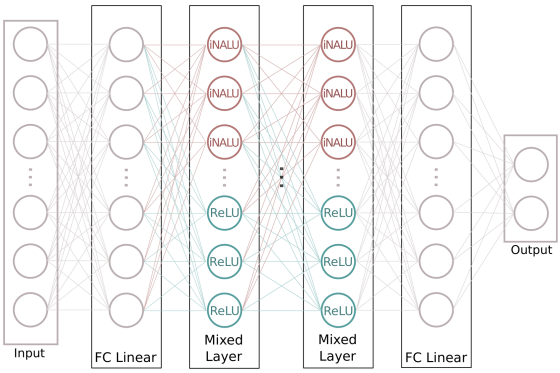

Fig. 3. Mixed Layer network model with fully connected linear input and output layers and 1 to k Mixed Layers as used in our experiments

part and the ReLU part of each layer has the full input dimension n as input, and contributes with an output dimension of $\frac{m}{2}$ concatenated to an output dimension of m for the complete layer (see Fig. 1). In combination with a linear layer as input and output layer, the network can thereby "route" and combine any input dimension to every part of each network layer by learning the weights accordingly. Therefore, the model is able to represent arithmetic as well as non-arithmetic feature relationships of varying complexity. We refer to the resulting model as shown in Fig. 3 as neuronal network with Mixed Layers (*Mixed Layers model* for short).

5 Experiments

5.1 Experimental Setup

Architecture. All experiments involve supervised training of a multi-layer-perceptron (MLP) as basic neural network architecture with a linear input, non-linear dense layers with ReLUs and a linear output layer. The number of neurons in the hidden layers varies over the experiments. To investigate the research question if introducing iNALUs can improve the performance of neural networks on the task of financial fraud detection, we use the same architecture and replace the non-linear dense layers with our Mixed Layers.

Train-Test Split. For all experiments, we use the same strategy to generate the train-test split: For training we randomly choose only few instances of the fraud class in order to keep the majority of fraudulent instances for evaluation. This approach reflects the class imbalance of the available data and emphasizes the requirement for a model to generalize from very few fraudulent samples to find new fraudulent cases when applied in a real-world scenario. In a preliminary study, we found that under-sampling the majority class to some extend didn't affect the performance negatively but reduced training-time by large margin.

Therefore only a random subset of the instances is used for training: For *Credit*, *PaySim* and *CCFraud* we use 2000 instances with a fraud-proportion of 1%, for *IEEE-CIS* we use 5000 instances with a fraud proportion of 4%. We then use the synthetic minority over-sampling technique (SMOTE) [5] to synthesize a balanced training dataset.

For the test dataset, we create a balanced split of 50% fraud and 50% benign samples, exclusively containing instances which haven't been used for training. This is motivated by the objective to study the ability of capturing mathematical relations rather than investigating effects of predicting strongly unbalanced data. To represent the variety of benign instances and avoid skewed results due to random fluctuations, we repeat this process 5 times with different random seeds and report the F1 score for the fraud class and for experiment 2 the Area Under the Receiver Operating Characteristic Curve (ROC-AUC) additionally.

Preprocessing. Each dataset is preprocessed by one-hot encoding categorical values. To assure comparability between all datasets, we follow the preprocessing strategy of the CCFraud dataset and apply PCA to all other datasets as well as min-max scaling to all datasets ensuring a valid train-test split by only fitting on training data. In a preliminary study, we verified that applying PCA to the datasets doesn't negatively impact the performance of neural networks with and without Mixed Layers. Applying PCA can also mitigate privacy issues which could possibly prevent from making a real dataset publicly available.

Training Procedure. For neural network training, we use ADAM as optimizer with a learning rate of 0.001, a weight decay of 0.0001 and Cross Entropy loss. The batch-size is set to 200 and all models are trained for 200 epochs, which have been validated as suitable training parameters in preliminary experiments.

5.2 Experiment 1

In the first experiment, we explore the influence of Mixed Layers in neural network architectures. Therefore, we construct neural networks containing Mixed Layer as well as neural networks of identical architecture exclusively with dense layers and ReLU activations. For each dataset all possible combinations between the number of input neurons (chosen from 10, 20, 30, 50 and 100) and the number of layers (chosen from 1 to 3) are evaluated to assess the performance and stability for different neural network capacities.

Results. The results are depicted in Fig. 4 and Fig. 5. The boxplots show the F1 scores on our test datasets for both neural architectures for different number of layers and varying number of hidden neurons in each layer. Each box summarizes the results of all runs for a certain parameter configuration with different random seeds. Note that both architectures share the same train- and test-splits for each run and parameter configuration. This ensures the comparability of the underlying performances and boxes for each parameter. For all dataset except IEEE-CIS, our proposed model yields very good results of F1 scores around 0.9. The performance of IEEE-CIS dataset is notably worse for both architectures

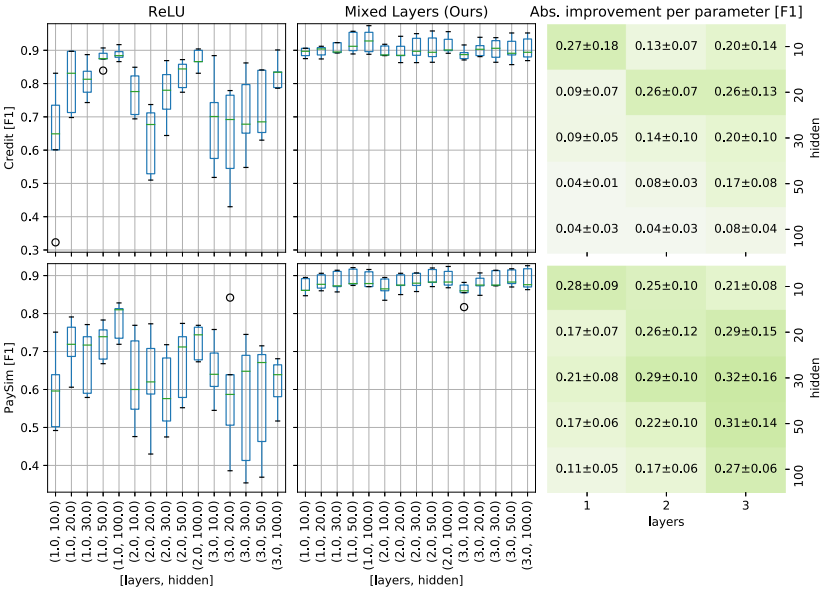

Fig. 4. F1 scores of experiment 1 on the synthetic Credit Payment and PaySim datasets. Mixed Layers describes the neural network structure as described in Sect. 4 and ReLU shows the results for the same model architecture having the Mixed Layers replaced by layers with ReLU activations. The heatmaps show the absolute improvement of Mixed Layers in comparison to the respective ReLU architecture for each parameter configuration averaged over all random seeds and their standard deviations.

and the results vary highly for different random seeds. As this dataset is much more complex regarding the number and kind of features, the task might require a different training strategy to achieve better results.

For all other datasets the performance of our model is very stable over all parameter configurations, which means that even a very small neural network consisting of one Mixed Layer and 10 hidden neurons (along with a linear input and output layer) solves the task sufficiently well. In comparison, the same architecture with one dense layer and ReLU activiations performs notably worse. To assess the actual performance improvement for each possible parameter configuration, we pairwise aligned parameters and seeds of both architectures and report the average absolute improvement per parameter configuration in Fig. 4 and Fig. 5. A positive improvement value describes a performance gain of Mixed Layers over the respective ReLU model, whereas a negative improvement means that the ReLU model performs better. The Mixed Layer model outperforms the ReLU model for all datasets except IEEE-CIS. For Credit and PaySim this observation holds for all network configurations, whereas for the CCFraud dataset large models (50 and 100 neurons) with one or two layers perform equally well.

5.3 Experiment 2

In the second experiment, we want to compare our model with several commonly used supervised classification algorithms. Precisely, we evaluate linear Support Vector Machines (SVM), Support Vector Machines with RBF kernel (SVM-RBF), k-Nearest Neighbor (kNN, $k = 3$), Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), Logistic Regression (LR) and eXtreme Gradient Boosting (XG-Boost). With Isolation Forest (IF) we also include an anomaly detection method for comparison. For all classifiers, we use the implementation from the python scikit-learn[4] library with default hyper-parameters except XG-Boost, which is evaluated using the xgboost[5] python library.

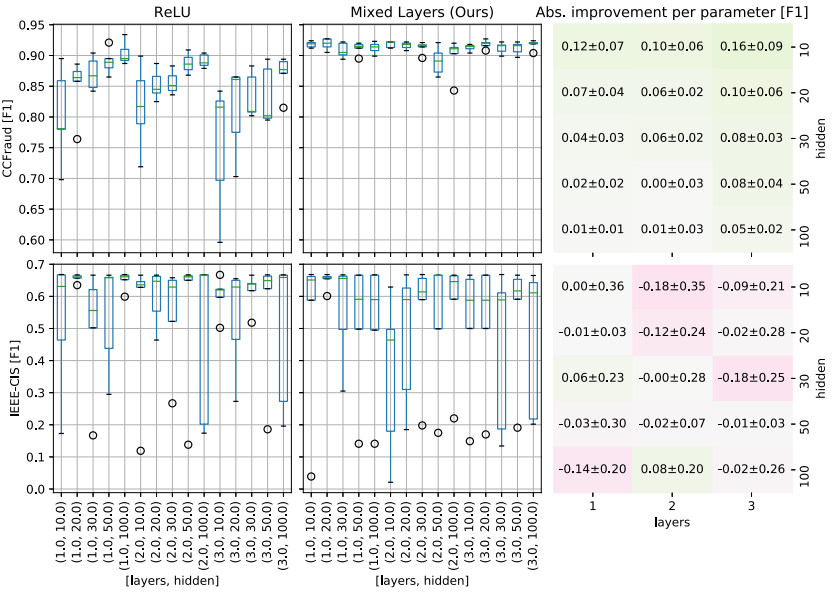

Fig. 5. F1 scores of experiment 1 on the real CCFraud and IEEE-CIS datasets. Mixed Layers describes the neural network structure as described in Sect. 4 and ReLU shows the results for the same model architecture having the Mixed Layers replaced by dense layers with ReLU activations. The heatmaps show the absolute improvement of our architecture in comparison to the respective ReLU architecture for each parameter configuration averaged over all random seeds and their standard deviations.

For the ReLU model, we use the most promising parameter configuration from experiment 1, which is one layer with 100 neurons. For the Mixed Layers model we used one layer with 20 neurons. We want to emphasize that due to the good stability of Mixed Layers over different parameter configurations, the

[4] https://scikit-learn.org v. 0.22.2.
[5] https://github.com/dmlc/xgboost v. 1.0.2.

Table 2. Average and standard deviation F1 score and ROC-AUC for several supervised classifiers in comparison to our model aggregated over different random seeds. For ReLU and our model, we conducted this experiment using the most promising parameter configuration from experiment 1, one layer with 100 neurons for ReLU and one layer with 20 neurons for our Mixed Layer model. The last column shows the average for each classifier over all datasets. The best results per dataset are printed in bold.

		Credit	PaySim	CCFraud	IEEE-CIS	Avg.
F1	SVM	0.88 ± 0.01	**0.89** ± 0.02	0.90 ± 0.01	0.41 ± 0.27	0.77
	SVM-RBF	**0.92** ± 0.03	0.85 ± 0.02	0.85 ± 0.07	0.43 ± 0.23	0.76
	kNN	0.89 ± 0.03	0.81 ± 0.01	0.91 ± 0.01	0.65 ± 0.03	0.81
	DT	0.87 ± 0.02	0.82 ± 0.03	0.80 ± 0.08	0.49 ± 0.04	0.74
	RF	0.89 ± 0.03	0.83 ± 0.03	0.88 ± 0.02	0.57 ± 0.05	0.80
	NB	0.85 ± 0.03	0.76 ± 0.05	0.91 ± 0.01	0.65 ± 0.02	0.80
	LR	0.88 ± 0.01	0.87 ± 0.02	**0.92** ± 0.01	0.55 ± 0.15	0.81
	XG-Boost	0.91 ± 0.02	0.84 ± 0.03	0.89 ± 0.01	0.62 ± 0.01	0.82
	IF	0.82 ± 0.01	0.81 ± 0.01	0.88 ± 0.01	**0.74** ± 0.01	0.81
	ReLU	0.89 ± 0.02	0.78 ± 0.04	0.90 ± 0.02	0.65 ± 0.03	0.81
	Mixed Layers	0.90 ± 0.01	0.88 ± 0.02	**0.92** ± 0.01	0.65 ± 0.02	**0.84**
ROC-AUC	SVM	0.95 ± 0.01	0.97 ± 0.01	0.95 ± 0.01	0.49 ± 0.12	0.84
	SVM-RBF	**0.98** ± 0.01	0.96 ± 0.01	0.93 ± 0.05	0.59 ± 0.09	0.86
	kNN	0.91 ± 0.02	0.86 ± 0.02	0.92 ± 0.01	0.54 ± 0.05	0.81
	DT	0.88 ± 0.02	0.85 ± 0.02	0.83 ± 0.05	0.52 ± 0.08	0.77
	RF	**0.98** ± 0.01	0.95 ± 0.01	**0.97** ± 0.00	0.48 ± 0.04	0.85
	NB	0.93 ± 0.02	0.92 ± 0.01	0.94 ± 0.03	0.50 ± 0.01	0.82
	LR	0.96 ± 0.01	0.96 ± 0.01	0.96 ± 0.01	0.45 ± 0.09	0.83
	XG-Boost	**0.98** ± 0.02	**0.98** ± 0.01	**0.97** ± 0.00	0.54 ± 0.01	0.87
	IF	0.95 ± 0.01	0.90 ± 0.01	0.95 ± 0.00	**0.74** ± 0.01	**0.89**
	ReLU	0.96 ± 0.01	0.88 ± 0.02	0.94 ± 0.01	0.64 ± 0.02	0.85
	Mixed Layers	0.97 ± 0.00	0.96 ± 0.01	0.96 ± 0.01	0.57 ± 0.12	0.87

parameter choice for Mixed Layer in this experiments is arbitrary and other parameter configurations perform comparably.

Results. The results of the second experiment are presented in Table 2. Our model is among the best four classifiers for all datasets. All classifiers perform well on each dataset except IEEE-CIS on which the best models except IF only achieve F1 scores of 0.65. Isolation Forest performs best on this dataset with a F1 score of 0.74 which suggests, that methods specifically tailored to anomaly detection can capture the characteristics of this dataset better. Comparing the F1 score averaged over all datasets (see Table 2, column Avg.), the Mixed Layer architecture yields significantly[6] better results compared to the best ReLU architecture. The results evaluated with the ROC-AUC metric support our findings with the exception of IEEE-CIS, where our Mixed Layers performed worse than

[6] $p = 0.00932$, Wilcoxon Signed-Rank Test [20] over all datasets and repetitions.

the ReLU layer. An in-depth analysis showed that two of the five repetitions with different random seeds yield notably worse results (0.41 and 0.44) which leads to the performance drop for the mean and the high standard deviation.

6 Discussion

In our experiments, we show that neural networks benefit from including Mixed Layers when applied to the task of fraud detection on four different datasets. This finding suggests that Mixed Layers improve the capability of neural networks to model mathematical relationships within the data. The neural network architectures containing our Mixed Layers have a good stability over different number of hidden neurons and layers. Even small networks yield competitive results with several well established supervised classification algorithms over different synthetic and real-world datasets. Overall the ReLU architecture seems much less stable regarding the different random seeds and parameter configurations.

Although our model was among the best classifiers for the IEEE-CIS dataset in experiment 2, all methods performed notably worse compared to the other datasets. This shows that the dataset is hard to predict in our evaluation setting and suggests that the unstable performance observed in experiment 1 is presumably not related to our model but rather to the dataset itself and might be explained by different aspects: On the one hand the dataset includes many features which might require intensive prepossessing and feature engineering. On the other hand the dataset is larger than all other datasets with respect to the number of features and fraud cases. This might require a different training procedure than the other datasets for example regarding the train-test split, the network architecture or the number of epochs for training. The observation, that Isolation Forest yields notably better results on IEEE-CIS also suggest that for more complex data sets methods adapted for anomaly detection should be used instead of standard classifiers. Since our study is not primarily conducted to achieve best performances on each dataset but rather to examine the research question, if neural network architectures benefit from iNALU neurons applied to the financial domain, our experiments focused on a fair comparison instead of thorough hyper-parameter tuning on individual datasets. However, tuning our proposed model to certain datasets in comparison to hyperparameter-tuned state-of-the-art classifiers may be interesting to investigate as future work.

On our synthetic Credit Payment dataset, some supervised classifiers performed surprisingly well, which by design of our dataset we didn't expect. Since solving the task correctly is fully dependent on capturing the correct mathematical relationship, we expected for example kNN to perform worse. We suspect the good performance is a result of applying PCA as preprocessing step, which might contribute to modeling the correct relation within the data. Examining the influence of different preprocessing steps e.g. training embedding layers end to end instead of PCA might be an interesting task for future work.

Both experiments show, that our model outperforms the respective ReLU baseline models. However, Mixed Layers with iNALUs contain more trainable

parameters than linear neurons with ReLU activations. One Mixed Layer in our experiments contains 50% ReLUs and 50% iNALUs and a iNALU has 4 times more trainable parameters. For a comparison of two architectures with an equal number of trainable parameters an architecture with Mixed Layers with a hidden dimension of 20 can be compared with an architecture with ReLU Layers with a hidden dimension of 50. In this comparison (see Fig. 4 and Fig. 5) the Mixed Layers model still outperforms the ReLU based model with an F1-score of 0.837 over 0.759 averaged over all datasets.

7 Conclusion

This work examined the question if a neural network architecture which includes hidden units specifically tailored to capturing mathematical relations is beneficial for supervised classification tasks on datasets in the financial fraud domain.

We designed a new Mixed Layer for neuronal networks which incorporates iNALUs and ReLUs. Further, we constructed a synthetic benchmark dataset specifically with the difficultly of modeling a mathematical relationship, which is inspired by a real-world credit payment application. We evaluated Mixed Layer based neural networks on two real-world and two synthetic datasets and compared it to a neuronal network with ReLU activations. The experiments show that Mixed Layers are able to improve the performance of neuronal networks on financial fraud data sets. We compare our proposed model with several well-established classification approaches in a supervised evaluation setting and perform among the best approaches for each dataset.

As we designed our baseline architecture as well as our proposed model to solve a supervised task, we constructed our experiments accordingly and evaluate on a balanced dataset. However, in a real-world setting fraud and benign transaction will not occur equally frequent and the actual financial prejudice of having more false positives or more false negatives will be task depending and require a more detailed evaluation including metrics to reflect these circumstances. Moreover many approaches applied to recognize financial fraud rely on anomaly-detection or novelty-detection techniques which often use one-class or even unsupervised approaches. As we generally showed the benefit of our proposed model in the supervised setting, for future work we plan to introduce it in other evaluation settings and compare it to unsupervised approaches, as well as approaches specifically designed for anomaly or novelty-detection. Other ideas for future work include optimizing and evaluating different network architectures and to vary the proportions of iNALU and ReLU neurons in the mixed layers as well as inspecting and refining preprocessing steps.

Acknowledgements. This work was partly funded by the Federal Ministry of Education and Research of Germany as part of the DeepScan project (01IS18045A). M. R. was supported by the BayWISS Consortium Digitization.

References

1. Baader, G., Krcmar, H.: Reducing false positives in fraud detection: combining the red flag approach with process mining. Int. J. Account. Inf. Syst. **31**, 1–16 (2018)
2. Bolton, R.J., Hand, D.J.: Statistical fraud detection: a review. Stat. Sci. **17**, 235–249 (2002)
3. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: a survey. arXiv preprint arXiv:1901.03407 (2019)
4. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. (CSUR) **41**(3), 1–58 (2009)
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
6. Chen, K., Dong, Y., Qiu, X., Chen, Z.: Neural arithmetic expression calculator. arXiv preprint arXiv:1809.08590 (2018)
7. Dal Pozzolo, A., Caelen, O., Johnson, R.A., Bontempi, G.: Calibrating probability with undersampling for unbalanced classification. In: IEEE Symposium Series on Computational Intelligence, pp. 159–166. IEEE (2015)
8. Freivalds, K., Liepins, R.: Improving the neural GPU architecture for algorithm learning. In: Workshop on Neural Abstract Machines & Program Induction (NAMPI), International Conference on Machine Learning (ICML) (2018)
9. Kaiser, L., Sutskever, I.: Neural GPUs learn algorithms. In: International Conference on Learning Representations (2016)
10. Kalchbrenner, N., Danihelka, I., Graves, A.: Grid long short-term memory. arXiv preprint arXiv:1507.01526 (2015)
11. Lopez-Rojas, E., Elmir, A., Axelsson, S.: PaySim: a financial mobile money simulator for fraud detection. In: European Modeling and Simulation Symposium (EMSS), pp. 249–255. Dime University of Genoa (2016)
12. Madsen, A., Rosenberg Johansen, A.: Measuring Arithmetic Extrapolation Performance (2019)
13. Maes, S., Tuyls, K., Vanschoenwinkel, B., Manderick, B.: Credit card fraud detection using Bayesian and neural networks. In: International Naiso Congress on Neuro Fuzzy Technologies, pp. 261–270 (2002)
14. Schlör, D., Ring, M., Hotho, A.: iNALU: improved neural arithmetic logic unit. arXiv e-prints arXiv:2003.07629, March 2020
15. Schreyer, M., Sattarov, T., Borth, D., Dengel, A., Reimer, B.: Detection of anomalies in large scale accounting data using deep autoencoder neural networks. In: GPU Technology Conference - Silicon Valley (2018)
16. Shen, A., Tong, R., Deng, Y.: Application of classification models on credit card fraud detection. In: International Conference on Service Systems and Service Management, pp. 1–4. IEEE (2007)
17. Sun, T., Vasarhelyi, M.A.: Predicting credit card delinquencies: an application of deep neural networks. Intell. Syst. Account. Finance Manag. **25**(4), 174–189 (2018)
18. Trask, A., Hill, F., Reed, S.E., Rae, J., Dyer, C., Blunsom, P.: Neural arithmetic logic units. In: Advances in Neural Information Processing Systems, pp. 8035–8044 (2018)
19. Wang, S., Liu, G., Li, Z., Xuan, S., Yan, C., Jiang, C.: Credit card fraud detection using capsule network. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 3679–3684 (2018)
20. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics Bull. **1**(6), 80–83 (1945)