Dual Sequential Variational Autoencoders for Fraud Detection

Ayman Alazizi[1,2(✉)], Amaury Habrard[1], François Jacquenet[1], Liyun He-Guelton[2], and Frédéric Oblé[2]

[1] Univ. Lyon, Univ. St-Etienne, UMR CNRS 5516, Laboratoire Hubert-Curien, 42000 Saint-Etienne, France
{ayman.alazizi,amaury.habrard,francois.jacquenet}@univ-st-etienne.fr
[2] Worldline, 95870 Bezons, France
{ayman.alazizi,liyun.he-guelton,frederic.oble}@worldline.com

Abstract. Fraud detection is an important research area where machine learning has a significant role to play. An important task in that context, on which the quality of the results obtained depends, is feature engineering. Unfortunately, this is very time and human consuming. Thus, in this article, we present the DuSVAE model that consists of a generative model that takes into account the sequential nature of the data. It combines two variational autoencoders that can generate a condensed representation of the input sequential data that can then be processed by a classifier to label each new sequence as fraudulent or genuine. The experiments we carried out on a large real-word dataset, from the Worldline company, demonstrate the ability of our system to better detect frauds in credit card transactions without any feature engineering effort.

Keywords: Anomaly detection · Fraud detection · Sequential data · Variational autoencoder

1 Introduction

An anomaly (also called outlier, change, deviation, surprise, peculiarity, intrusion, etc.) is a pattern, in a dataset, that does not conform to an expected behavior. Thus, anomaly detection is the process of finding anomalies in a dataset [4]. Fraud detection, a subdomain of anomaly detection, is a research area where the use of machine learning can have a significant financial impact for companies suffering from large frauds and it is not surprising that a very large amount of research has been conducted over many years in that field [1].

At the Wordline company, we process billions of electronic transactions per year in our highly secured data centers. It is obvious that detecting frauds in that context is a very difficult task. For many years, the detection of credit card frauds within Wordline has been based on a set of rules manually designed by experts. Nevertheless such rules are difficult to maintain, difficult to transfer to other business lines, and dependent on experts who need a very long training

M. R. Berthold et al. (Eds.): IDA 2020, LNCS 12080, pp. 14–26, 2020.
https://doi.org/10.1007/978-3-030-44584-3_2

period. The contribution of machine learning in this context seems obvious and Wordline has decided for several years to develop research in this field.

Firstly, Worldline has put a lot of effort in feature engineering [3,9,12] to develop discriminative handcrafted features. This improved drastically supervised learning of classifiers that aim to label card transactions as genuine or fraudulent. Nevertheless, designing such features requires a huge amount of time and human resources which is very costly. Thus developing automatic feature engineering methods becomes a critical issue to improve the efficiency of our models. However, in our industrial setting, we have to face with many issues among which the presence of highly imbalanced data where the fraud ratio is about 0.3%. For this reason, we first focused on classic unsupervised approaches in anomaly detection where the objective is to learn a model from normal data and then isolate non-compliant samples and consider them as anomalies [5,17,19,21,22].

In this context, Deep autoencoder [7] is considered as a powerful data modeling tool in the unsupervised setting. An autoencoder (AE) is made up of two parts: an encoder designed to generate a compressed coding from the training input data and a decoder that reconstructs the original input from the compressed coding. In the context of anomaly detection [6,20,22], an autoencoder is generally trained by minimizing the reconstruction error only on normal data. Afterwards, the reconstruction error is applied as an anomaly score. This assumes that the reconstruction error for a normal data should be small as it is close to the learning data, while the reconstruction error for an abnormal data should be high.

However, this assumption is not always valid. Indeed, it has been observed that sometimes the autoencoder generalizes so well that it can also reconstruct anomalies, which leads to view some anomalies as normal data. This can also be the case when some abnormal data share some characteristics of normal data in the training set or when the decoder is "too powerful" to properly decode abnormal codings. To solve the shortcomings of autoencoders, [13,18] proposed the *negative learning technique* that aims to control the compressing capacity of an autoencoder by optimizing conflicting objectives of normal and abnormal data. Thus, this approach looks for a solution in the gradient direction for the desired normal input and in the opposite direction for the undesired input.

This approach could be very appealing to deal with fraud detection problems but we found that it is sometimes not sufficient in the context of our data. Indeed, it is generally almost impossible to obtain in advance a dataset containing all representative frauds, especially in the context where unknown fraudulent transactions occur on new terminals or via new fraudulent behaviors. This has led us to consider more complex models with variational autoencoders (VAE), a probabilistic generative extension of AE, able to model complex generative distributions that we found more adapted to efficiently model new possible frauds.

Another important point for credit card fraud detection is the sequential aspect of the data. Indeed, to test a card for example, a fraudster may try to make several (small) transactions in a short time interval, or directly perform an abnormally high transaction with respect to existing transactions of the true card holder. In fact this sequential aspect has been addressed either indirectly via aggregated features [3], that we would like to avoid designing, or directly by sequential models such as LSTM, but [9] report nevertheless that the LSTM did not improve much the detection performance for e-commerce transactions. One of the main contribution of this paper is to propose a method to identify fraudulent sequences of credit transactions in the context of highly imbalanced data. For this purpose, we propose a model called DuSVAE, for Dual Sequential Variational Autoencoders, that consists of a combination of two variational autoencoders. The first one is trained from fraudulent sequences of transactions in order to be able to project the input data into another feature space and to assign a fraud score to each sequence thanks to the reconstruction error information. Once this model is trained, we plug a second VAE at the output of the first one. This second VAE is then trained with a negative learning approach with the objective to maximize the reconstruction error of the fraudulent sequences and minimize the reconstruction error of the genuine ones.

Our method has been evaluated on a Wordline dataset for credit card fraud detection. The obtained results show that DuSVAE can extract hidden representations able to provide results close to those obtained after a significant work of feature engineering, therefore saving time and human effort. It is even possible to improve the results when combining engineered features with DuSVAE.

The article is organized as follows: some preliminaries about the techniques used in this work are given in Sect. 2. Then we describe the architecture and the training strategy of the DusVAE method in Sect. 3. Experiments are presented in Sect. 4 after a presentation of the dataset and useful metrics. Finally Sect. 5 concludes this article.

2 Preliminaries

In this section, we briefly describe the main techniques that are used in DuSVAE: vanilla and variational autoencoders, negative learning and mixture of experts.

2.1 Autoencoder (AE)

An AE is a neural network [7], which is optimized in an unsupervised manner, usually used to reduce the dimensionality of the input data. It is made up of two parts linked together: an encoder $E(x)$ and a decoder $\mathcal{D}(z)$. Given an input sample x, the encoder generates z, a condensed representation of x. The decoder is then tuned to reconstruct the original input x from the encoded representation z. The objective function used during the training of the AE is given by:

$$\mathcal{L}_{AE}(x) = \|x - \mathcal{D}(E(x))\| \tag{1}$$

where $\| \cdot \|$ denotes an arbitrary distance function. The ℓ_2 norm is typically applied here. The AE can be optimized for example using stochastic gradient descent (SGD) [10].

2.2 Variational Autoencoder (VAE)

A VAE [11,16] is an attractive probabilistic generative version of the standard autoencoder. It can learn a complex distribution and then use it as a generative model defined by a prior $p(z)$ and conditional distribution $p_\theta(x|z)$. Due to the fact that the true likelihood of the data is generally intractable, a VAE is trained through maximizing the evidence lower bound (ELBO):

$$\mathcal{L}(x;\theta,\phi) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - D_{\mathrm{KL}}\left(q_\phi(z|x)\|p(z)\right) \tag{2}$$

where the first term $\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]$ is a negative reconstruction loss that enforces $q_\phi(z|x)$ (the encoder) to generate a meaningful latent vector z, so that $p_\theta(x|z)$ (the decoder) can reconstruct the input x from z. The second term $D_{\mathrm{KL}}\left(q_\phi(z|x)\|p(z)\right)$ is a KL regularization loss that minimizes the KL divergence between the approximate posterior $q_\phi(z|x)$ and the prior $p(z) = \mathcal{N}(\mathbf{0}, \mathbf{I})$.

2.3 Negative Learning

Negative learning is a technique used for regularizing the training of the AE in the presence of labelled data by limiting reconstruction capability (LRC) [13]. The basic idea is to maximize the reconstruction error for abnormal instances, while minimizing the reconstruction error for normal ones in order to improve the discriminative ability of the AE. Given an input instance $x \in \mathbb{R}^n$ and $y \in \{0,1\}$ denotes its associated label where $y = 1$ stands for a fraudulent instance and $y = 0$ for a genuine one. The objective function of LRC to be minimized is:

$$(1-y)\mathcal{L}_{AE}(x) - (y)\mathcal{L}_{AE}(x) \tag{3}$$

Training LRC-based models has the major disadvantage to be generally unstable due to the fact that the anomaly reconstruction error is not upper bounded. The LRC approach tends then to maximize the reconstruction error for known anomalies rather than minimizing the reconstruction error for normal points leading to a bad reconstruction of normal data points. To overcome this problem, [18] has proposed Autoencoding Binary Classifiers (ABC) for supervised anomaly detection that improves LRC by using an objective function based on a bounded reconstruction loss for anomalies, leading to a better training stability. The objective function of the ABC to be minimized is:

$$(1-y)\mathcal{L}_{AE}(x) - y\log_2(1-e^{-\mathcal{L}_{AE}(x)}) \tag{4}$$

2.4 Mixture-of-Experts Layer (MoE)

In addition to the previous methods, we now present the notion of MoE layer [8] that will be used in our model.

The MoE layer aims to combine the outputs of a group of n neural networks called experts $EX_1, EX_2, ..., EX_n$. The experts have their specific parameters but work on the same input, their n output are combined linearly with the

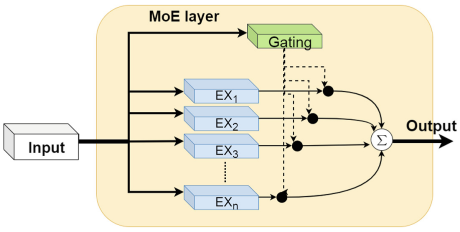

Fig. 1. An illustration of the MoE layer architecture

outputs of the gating network G which weights the experts according to the input x. See Fig. 1 for an illustration. Let $E_i(x)$ be the output of expert EX_i, and $G(x)_i$ be the i^{th} attribute of $G(x)$, then the output y of the MoE is defined as follows:

$$y = \sum_{i=1}^{n} G(x)_i EX_i(x). \tag{5}$$

The intuition behind MoE layers is to train different network experts that can focus on specific peculiarities of the data and then choose an appropriate combination of experts with respect to the input x. In our industrial context, such a layer would help us to take into account different behaviors from millions of cardholders, which results in a variety of data distributions. The different expert networks can thus model various behaviors observed in the dataset and be combined adequately in function of the input data.

3 The DuSVAE Model

In this section, we present our approach to extract a hidden representation of input sequences to be used for anomaly/fraud detection. We first introduce the model architecture with the loss functions used, then we describe the learning procedure used to train the model.

3.1 Model Architecture

We assume in the following that we are given as input a set of sequences $\mathcal{X} = \{x \mid x = (t^1, t^2, ..., t^m) \text{ with } t^i \in \mathbb{R}^d\}$, every sequence being composed of m transactions encoded by numerical vectors. Each sequence is associated to a label $y \in \{0, 1\}$ such that $y = 1$ indicates a fraudulent sequence and $y = 0$ a genuine one. We label a sequence as fraudulent if its last transaction is a fraud.

As illustrated in Fig. 2, our approach consists of two sequential variational autoencoders. The first one is trained only on fraudulent sequences of the training data. We use the generative capacity of this autoencoder to generate diverse and representative instances of fraudulent instances with respect to the sequences given as input. This autoencoder has the objective to prepare the data for the

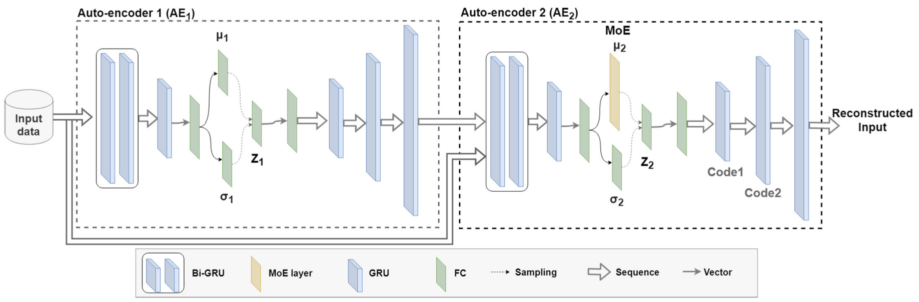

Fig. 2. The DuSVAE model architecture

second autoencoder and to provide also a first anomaly/fraud score with the reconstruction error.

The first layers of the autoencoders are bi-directional GRU layers allowing us to handle sequential data. The remaining parts of the encoder and the decoder contain GRU and fully connected (FC) layers, as shown in Fig. 2. The loss function used to optimize the reconstruction error of the first autoencoder is defined as follows:

$$\mathcal{L}_{rec}(x, \phi_1, \theta_1) = mse(x, \mathcal{D}_{\theta_1}(E_{\phi_1}(x))) + D_{\text{KL}}\left(q_{\phi_1}(z|x) \| p(z)\right), \tag{6}$$

where mse is the mean square error function and $p(z) = \mathcal{N}(\mathbf{0}, \mathbf{I})$. The encoder $E_{\phi_1}(x)$ generates a latent representation z according to $q_{\phi_1}(z|x) = \mathcal{N}(\mu_1, \sigma_1)$. The decoder $\mathcal{D}_{\theta_1}$ tries to reconstruct the input sequence from z. In order to avoid mode collapse between the reconstructed transactions of the sequence, we add the following loss function to control the reconstruction of individual transactions with respect to relative distances from an input sequence x:

$$\mathcal{L}_{trx\,AE}(x, \phi_1, \theta_1) = \sum_{i=1}^{m} \sum_{j=i+1}^{m} \frac{1}{d} \| abs(t^i - t^j) - abs(\overline{t}^i - \overline{t}^j) \|_1 \tag{7}$$

where $\overline{t}^i$ is the reconstruction obtained by the AE for the i^{th} transaction of the sequence and $abs(t)$ returns a vector where the features are the absolute values of the original input vector t.

So, we train the parameters (ϕ_1, θ_1) of the first autoencoder by minimizing the following loss function over all the fraudulent sequences of the training samples:

$$\mathcal{L}_1(x, \phi_1, \theta_1) = \mathcal{L}_{rec}(x, \phi_1, \theta_1) + \lambda \mathcal{L}_{trx}(x, \phi_1, \theta_1), \tag{8}$$

where λ is a tradeoff parameter.

The second autoencoder is then trained over all the training sequences by negative learning. It takes as input both a sequence x and its reconstructed version from the first autoencoder $AE_1(x)$ that corresponds to the output of its last layer. The loss function considered to optimize the parameters (ϕ_2, θ_2) of the second autoencoder is then defined as follows:

$$\mathcal{L}_2(x, AE_1(x), \phi_2, \theta_2) = (1-y)\mathcal{L}_1(x, \phi_2, \theta_2) \\ -y(\overline{\mathcal{L}_1}(x, \phi_1, \theta_1) + \epsilon)\log_2(1 - e^{-\mathcal{L}_1(x,\phi_2,\theta_2)}), \quad (9)$$

where $\overline{\mathcal{L}_1}(x, \phi_1, \theta_1)$ denotes the reconstruction loss $\mathcal{L}_1$ rescaled in the $[0, 1]$-interval with respect to all fraudulent sequences and ϵ is a small value used to smooth very low anomaly scores. The architecture of this second autoencoder is similar to that of the first one, except that we use a MoE layer to compute the mean of the normal distribution $\mathcal{N}(\mu_2, \sigma_2)$ defined by the encoder. As said previously, the objective is to take into account the variety of the different behavior patterns found in our genuine data. The experts used in that layer are simple one-layer feed-forward neural networks.

3.2 The Training Strategy

The global learning algorithm is presented in Algorithm 1. We have two training phases, the first one focuses on training the first autoencoder AE_1 as a backing model for the second phase. It is trained only on fraudulent sequences by minimizing Eq. 8. Once the model has converged, we freeze its weights and start the second phase. For training the second autoencoder AE_2, we use both genuine and fraudulent sequences and their reconstructed versions given by AE_1. We then optimize the weights of AE_2 by minimizing Eq. 9. To control the imbalance ratio, the training is done at each iteration by sampling n examples from fraudulent sequences and n from genuine sequences. We repeat this step iteratively by increasing the number n of sampled transactions for each novel iteration until the model converges.

Algorithm 1. Dual sequential variational autoencoder (DuSVAE)

```
1: **Input:** $\mathcal{X}_g$ genuine data, $\mathcal{X}_f$ fraudulent data.
2: **Parameters:** $n$ number of sampled examples; $h$ increment step.
3: **Output:** $AE_1$ Autoencoder, $AE_2$ Autoencoder.
4: repeat
5:     Train $AE_1$ on $\mathcal{X}_f$ by minimizing Equation 8
6: until convergence
7: Freeze the weights of $AE_1$
8: repeat
9:     $\mathcal{X}_1 \leftarrow Sample(\mathcal{X}_f, n) \cup Sample(\mathcal{X}_g, n)$
10:    $\mathcal{X}_2 \leftarrow AE_1(\mathcal{X}_1)$
11:    Train $AE_2$ on $(\mathcal{X}_1, \mathcal{X}_2)$ by minimizing Equation 9
12:    if $n \leq |\mathcal{X}_f|$ then
13:        $n \leftarrow n + h$
14:    end if
15: until convergence
```

Table 1. Properties of the Worldline dataset used in the experiments.

	Train (01/01-21/03)	Validation (22/03-31/03)	Test (01/04-30/04)
# of genuine	25,120,194	3,019,078	9,287,673
# of fraud	88,878	9,631	29,614
Total	25,209,072	3,028,709	9,317,287
Imbalance ratio	0.003526	0.00318	0.003178

4 Experiments

In this section, we provide an experimental evaluation of our approach on a real-world dataset of credit card e-payment transactions provided by Worldline. First, we present the dataset, then we present the metrics used to evaluate the models learned by our system and finally, we compare DuSVAE with other state-of-the-art approaches.

4.1 Dataset

The dataset provided by Wordline covers 4 months of credit card e-payment transactions made by European cardholders in e-commerce mode that has been splitted into **Train**, **Validation** and **Test** sets used respectively to train, tune and test the learned models. Its main challenges have been studied in [2], one of them being the imbalance ratio as we can see on Table 1 that presents the main characteristics of this dataset.

Each transaction is described by 12 features. A Boolean value is assigned to each transaction to specify whether it corresponds to a fraud or not. This labeling is handled by a team of human experts.

Since most features have a large number of values, using brute one-hot encoding would generate a huge number of features. For example the "Merchant Category Code" feature has 283 possible values and one-hot encoding would produce 283 new features. That would make our approach inefficient. Thus, before using one-hot encoding, we transform each categorical value of each feature by a score which is its risk to be associated with a fraudulent transaction. Let's consider for example a categorical feature f. We can compute the probability of the j^{th} value of feature f to be associated with a fraudulent transaction, denoted as β_j, as follows: $\beta_j = \frac{N^+_{f=j}}{N_{f=j}}$, where $N^+_{f=j}$ is the number of fraudulent transactions where the value of feature f is equal to j and $N_{f=j}$ is the total number of transactions where the value of feature f is equal to j. In order to take into account the number of transactions related to a particular value of a given feature, we follow [14]. For each value j of a given feature, the fraud score S_j for this value is defined as follows:

$$S_j = \alpha'_j \beta_j + \left(1 - \alpha'_j\right) \text{AFP} \tag{10}$$

This score computes a weighted value of β_j and the probability of having a fraud in a day (Average Fraud Probability: AFP). The weight α'_j is a normalized value of α_j in the range [0, 1], where α_j is defined as the proportion of the number of transactions for that value on the total number N of transactions: $\alpha_j = \frac{N_{f=j}}{N}$.

Having replaced each value for each feature by its score, we can then run one-hot encoding and thus significantly reduce the number of features generated. For example, the "Merchant Category Code" feature has 283 possible values and instead of generating 283 features, this technique produces only 29 features.

Finally, to generate sequences from transactions, we grouped all the transactions by cardholder ID and we ordered each cardholder's transactions by time. Then, with a sliding window over the transactions we obtained a time-ordered sequence of transactions for each cardholder. For each sequence, we have assigned the label *fraudulent* or *genuine* of its last transaction.

4.2 Metrics

In the context of fraud detection, fortunately, the number of fraudulent transactions is significantly lower than the number of normal transactions. This leads to a very imbalanced dataset. In this situation, the traditional performance measures are not appropriate. Indeed, with an overall fraud rate of 0.3%, classifying each transaction as normal leads to an accuracy of 99.7%, despite the fact that the model is absolutely naive. That means we have to choose appropriate performance measures that are robust in the case of imbalanced data. In this work we rely on the area under the precision-recall curve (AUC-PR) as a robust and clear measure of the accuracy of the classifier in an imbalanced setting. Each point of the precision-recall curve corresponds to the precision of the classifier at a specific recall level.

Once an alert is raised after a fraud has been detected, fraud experts can contact the card-holder to check the validity of suspicious transactions. So, within a single day, the fraud experts have to check a large number of transactions provided by the fraud detection system. Consequently, the precision of the transactions highlighted as fraud is an important metric because that can help human experts at Worldline to focus on the most important frauds and leave aside minor frauds due to lack of time to process them. For this purpose, we rely on the $P_{@K}$ as a global metric to compare models. It is the average of the precision of the first K transactions which are calculated according to the following equation.

$$\text{Average}P_{@K} = \frac{1}{K}\sum_{i=1}^{K} P_{@i} \tag{11}$$

4.3 Comparison with the State of the Art

We compare our approach with the following methods: variational autoencoder [11,16] trained on fraudulent or genuine data only (VAE(F) or VAE(G) respectively); limiting reconstruction capability (LRC) [13] and autoencoding binary classifiers for supervised anomaly detection (ABC) [18]. It is important to note

Table 2. AUC-PR achieved by CatBoost using various autoencoder models

Models	Raw		Reconstructed		Reconstruction error	Code1	Code2
	Trx	Seq	Trx	Seq			
VAE (F)	0.19	0.40	0.36	0.38	0.29	0.30	0.27
VAE (G)			0.42	0.43	0.31	0.32	0.33
LRC			0.46	0.46	0.17	0.28	0.13
ABC			0.48	0.50	**0.37**	0.32	0.3
DuSVAE			**0.51**	**0.53**	0.36	**0.50**	**0.49**

that ABC and LRC are not sequential models by nature. So, to make our comparison more fair, we adapted their implementation to allow them to process sequential data. As a classifier, we used CatBoost [15] which is robust in the context of imbalanced data and efficient on GPUs.

First, as we can observe in Table 2, the AUC-PR values obtained by running CatBoost directly on transactions and sequences of transactions are respectively equal to 0.19 and 0.40. If we look at the AUC-PR values obtained by running CatBoost on the reconstructed transactions and sequences of transactions, we can observe that the results are always greater than those obtained by running CatBoost on raw data. Moreover it is interesting to note that DuSVAE achieved the best results (0.51 and 0.53) compared to other state-of-the-art systems.

Now, if we look at the performance obtained by CatBoost on the hidden representation vectors Code1 and Code2, we observe that DuSVAE outperforms the results obtained by other state-of-the-art systems and those results are quite similar to the ones obtained on the reconstructed sequences of transactions. This is interesting because it means that using DuSVAE a condensed representation of the input data can be obtained, which still gives approximately the same results as on the reconstructed sequences of transactions but that are of higher dimensionality (about 10 times more) and can be less efficiently processed by the classifier. Finally, when using the reconstruction error as a score to classify fraudulent data, as done usually in anomaly detection, we can observe that DuSVAE is competitive with the best method. However, the performance level of Code1 and Code2 with CatBoost being significantly better makes the use of the hidden representations a better strategy than using the reconstruction error.

We then evaluated the impact of handcrafted features built by Worldline on the classifier performance. As we can see on the first two lines of Table 3, adding handcrafted features to the original sequential raw dataset leads to much better results both from the point of view of AUC-PR measure and P@K measure.

Now if we consider using DuSVAE (rows 3 and 4 of Table 3), we can also notice a significant improvement of the results obtained on the raw dataset of sequences augmented by handcrafted features compared to the results obtained on the original one without these additional features. This is observed for both the AUC-PR measure and the P@K measure. We see that, for the moment, by using a classifier on the sequences reconstructed by DuSVAE on just the

Table 3. AUC-PR and P@K achieved by CatBoost for sequence classification.

Input	AUC-PR	P@100	P@500
Raw data	0,40	0.43	0.11
Raw data + Handcrafted features	0,60	0.62	0.938
DuSVAE (The input:raw data)	0,53	0.88	0.72
DuSVAE (The input: raw data + Handcrafted features)	0,65	0.85	0.941

raw dataset (AUC-PR = 0.53), we cannot reach the results obtained when we use this classifier on the raw dataset augmented by handcrafted features (AUC-PR = 0.60). This can be explained by the fact that those features are based on history and profiling techniques that embed information covering a period of time larger than the one used for our dataset. Nevertheless we are not so far and the fact that using DuSVAE on the dataset augmented by handcrafted features (AUC-PR = 0.65) leads to better results than using the classifier without DuSVAE (AUC-PR = 0.60) is promising.

Table 3 also shows that the very good P@K values obtained when running the classifier on the sequences of transactions reconstructed by DuSVAE mean that DuSVAE can be a very significant help for experts to focus on real fraudulent transactions and not waste time on fake ones.

5 Conclusion

In this paper, we presented the DuSVAE model which is a new fraud detection technique. Our model combines two sequential variational autoencoders to produce a condensed representation vector of the input sequential data that can then be used by a classifier to label new sequences of transactions as genuine or fraudulent. Our experiments have shown that the DuSVAE model produces much better results, in terms of AUC-PR and $P_{@K}$ measures, than state-of-the-art systems. Moreover, the DuSVAE model produces a condensed representation of the input data which can replace very favorably the handcrafted features. Indeed, running a classifier on the condensed representation of the input data built by the DuSVAE model leads to outperform the results obtained on the raw data, with or without handcrafted features.

We believe that a first interesting way to further improve our results will be to focus on attention mechanisms to better take into account the history of past transactions in the detection of present frauds. A second approach will be to better take into account the temporal aspects in the sequential representation of our data and to reflect it in the core algorithm.

References

1. Abdallah, A., Maarof, M.A., Zainal, A.: Fraud detection system: a survey. J. Netw. Comput. Appl. **68**, 90–113 (2016)
2. Alazizi, A., Habrard, A., Jacquenet, F., He-Guelton, L., Oblé, F., Siblini, W.: Anomaly detection, consider your dataset first, an illustration on fraud detection. In: Proceedings of ICTAI 2019. IEEE (2019)
3. Bahnsen, A.C., Aouada, D., Stojanovic, A., Ottersten, B.: Feature engineering strategies for credit card fraud detection. Expert Syst. Appl. **51**, 134–142 (2016)
4. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: a survey. ACM Comput. Surv. **41**(3), 15:1–15:58 (2009)
5. Golan, I., El-Yaniv, R.: Deep anomaly detection using geometric transformations. In: Proceedings of NIPS, pp. 9758–9769 (2018)
6. Hasan, M., Choi, J., Neumann, J., Roy-Chowdhury, A.K., Davis, L.S.: Learning temporal regularity in video sequences. In: Proceedings of CVPR, pp. 733–742 (2016)
7. Hinton, G.E.: Connectionist learning procedures. Artif. Intell. **40**(1–3), 185–234 (1989)
8. Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E., et al.: Adaptive mixtures of local experts. Neural Comput. **3**(1), 79–87 (1991)
9. Jurgovsky, J., et al.: Sequence classification for credit-card fraud detection. Expert Syst. Appl. **100**, 234–245 (2018)
10. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv:1412.6980 (2014)
11. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Proceedings of ICLR (2014)
12. Lucas, Y., et al.: Towards automated feature engineering for credit card fraud detection using multi-perspective HMMs. Future Gener. Comput. Syst. **102**, 393–402 (2020)
13. Munawar, A., Vinayavekhin, P., De Magistris, G.: Limiting the reconstruction capability of generative neural network using negative learning. In: Proceedings of the International Workshop on Machine Learning for Signal Processing, pp. 1–6 (2017)
14. Pozzolo, A.D.: Adaptive machine learning for credit card fraud detection. Ph.D. thesis, Université libre de Bruxelles (2015)
15. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: Catboost: unbiased boosting with categorical features. In: Proceedings of NIPS, pp. 6638–6648 (2018)
16. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. arXiv:1401.4082 (2014)
17. Sabokrou, M., Khalooei, M., Fathy, M., Adeli, E.: Adversarially learned one-class classifier for novelty detection. In: Proceedings of CVPR, pp. 3379–3388 (2018)
18. Yamanaka, Y., Iwata, T., Takahashi, H., Yamada, M., Kanai, S.: Autoencoding binary classifiers for supervised anomaly detection. arXiv:1903.10709 (2019)
19. Zhai, S., Cheng, Y., Lu, W., Zhang, Z.: Deep structured energy based models for anomaly detection. arXiv:1605.07717 (2016)
20. Zhao, Y., Deng, B., Shen, C., Liu, Y., Lu, H., Hua, X.S.: Spatio-temporal autoencoder for video anomaly detection. In: Proceedings of the ACM International Conference on Multimedia, pp. 1933–1941 (2017)

21. Zimek, A., Schubert, E., Kriegel, H.P.: A survey on unsupervised outlier detection in high-dimensional numerical data. Stat. Anal. Data Mining: ASA Data Sci. J. **5**(5), 363–387 (2012)
22. Zong, B., et al.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: Proceedings of ICLR (2018)