Hybrid approach for Anomaly Detection in Time Series Data

1st Zeineb Ghrib
Devoteam Research, France
University of Paris VIII
Paris, France
zeineb.ghrib@devoteam.com

2nd Rakia Jaziri
University of Paris VIII
Paris, France
rakia.jaziri@univ-paris8.fr

3rd Rim Romdhane
Devoteam Research, France
Paris, France
rim.romdhane@devoteam.com

***Abstract*—Anomaly detection is an active research field which attracts the attention of many business and research actors. It has led to several research projects depending on the nature of the data, the availability of labels on normality, and domains of application that are diverse such as fraud detection, medical domains, cloud monitoring or network intrusions detection, etc. However, dealing with effective anomaly detection for complex and high-dimensional time series data remains a challenging task. In this work, we propose hybrid approach composed of an LSTM Autoencoder trained on normal records to learn efficient normal sequence representations combined with an SVM classifier for anomaly detection. Experimental results show that by encoding time series via a pretrained LSTM encoder allows efficient representation of data so that we can accurately detect abnormal records. In fact, the encoded representation reduces significantly the correlations between normal and abnormal records and allows us to have an efficient latent data representation that separates consistently the two classes. The proposed hybrid approach outperforms state-of-the art approaches [1], [2], [3], [4].**

***Index Terms*—Anomaly detection · Supervised Classification · Recurrent Neural Network · LSTM Autoencoder · Latent**

I. Introduction

It has been shown that the performance of any machine learning algorithm applied on complex tasks, such as images classification or sequence prediction, is highly correlated with the quality of the extracted features. Handcrafted feature extraction is tedious and usually inefficient. Many researches have been conducted to overcome this challenge by proposing different approaches to extract efficient features for time series classification [5].
In our case, we address the anomaly detection issue as a fraud detection application. We work on a public data set composed of time-stamped banking transactions. This type of data raises several challenges: (1) First, the sequence length is variable, which is problematic because most of machine learning algorithms, are designed to process fixed length data inputs.(2)Second, we have to find an efficient representation that captures chronological sequencing across transaction records: which often requires deep domain expertise or complex signal processing. The model that would be the most appropriate to learn the sequencing of transactions representation would be the Long Short Term Memory Model (LSTM) [6] which is a Recurrent Neural Network (RNN), that has shown excellent results in applications involving sequencing in the data such as : machine translation (or sequence to sequence embedding) [7], image captioning [8], hand writing generation [9]. LSTMs [6] are based on sophisticated cells allowing the model to capture long-term dependencies and patterns in time series. Furthermore, another powerful model called Autoencoder [10], is used for reconstructing data input and more specifically, for learning efficient representations of the input data, called encodings. These models turned to be very powerful feature detectors. Many recent researches has been conducted around this topic [11], [12]. Authors in [13] have used Autoencoders called Generative Adversarial Networks to detect efficient features, they were able to generate new realistic data that have the same distribution as the training data. This Autoencoder model was able to generate synthesized pictures that are very realistic people faces. In a later work [14] , authors have used generative Autoencoders to make novel molecular structures with desirable pharmacological and physio-chemical properties. In our work, we propose to investigate the power of a merged LSTM Autoencoder to tackle the problem of sequence representations of banking transactions to adress anomaly detection problem. Our model inherits from the Autoencoders the ability of learning efficient representations, and from the LSTM Neural Networks the sequence embedding.

II. RELATED WORK

Anomaly detection or outlier detection, aims to identify rare items, events or observations that significantly deviate from the normal data distribution. Typically, anomalous data can be connected to some kind of problem or rare event such as bank fraud [15], medical problems [16], structural defects or malfunctioning equipment [11] etc. Many anomaly detection algorithms have been proposed [1], [2], [3], [4] to address outlier detection issues. Among the most used ones we can mention one-class support vector machine (OCSVM) [17], [1], which is trained only on examples belonging to one of the two classes, and proceeds by constructing an hyper-sphere around the observed data samples so that new observations are classified as normal if they get through the sphere, outliers otherwise.
However, OCSVMs are very heavy in term of computational complexity, which makes it unsuitable for large volume of

data or if we have limited resources. Another wellknown algorithm used for outlier detection with linear complexity is the isolation forest [3], it consists of building an ensemble of trees for a given data set, and then anomalies are those instances which have short average path lengths on the trees. Recent works [18], [19] have applied more advanced techniques such as Autoencoders, which are composed of pairs of Neural Network models that learn to reconstruct the training inputs making the Autoencoders very efficient in learning consistent latent representations. For a well-trained Autoencoder the encodings are so representative that the decoder is able to rely on it to reconstruct the input data without much loss. Most of the work have used the reconstruction error as a key indicator: the model is trained on "normal" records aiming to minimize the reconstruction error (which is the loss function). The main assumption in using Autoencoders is that anomalies would suffer from a remarkable high reconstruction error, that can be used as an outlier score.

In our approach, we proceed in a different way by using the Autoencoder to learn efficient latent representation then we keep only the trained encoder model part and we combined it with a binary supervised classifier, in our work we used different implementations of SVM classifier to compare the classification performances.

Very recently, [20] have applied a neural network as an outlier classifier following a basic pretrained encoder. The use of a Neural Network assumes that the Autoencoder can fail to produce enough representational encoding so that the following Neural Network can compensate this lack of efficiency due to its ability to extract progressively new features. However in our work we will prove that using the adequate type of Autoencoder for time series with the suitable parameters can be efficient enough that a simple classifier is quite sufficient to conduct accurate classification, without need to deploy complex and costly neural network model.

The type of the deployed Autoencoder, which is the keystone of our model, is LSTM Neural Networks [6]. This type of Neural Network have basically been designed to overcome the limitation of RNN (Recurrent Neural Network) vanishing gradient problem [21] by employing elaborated recurrent cells that are able to capture long-range dependencies. However, to the best of our knowledge the power of using the LSTM Autoencoder to learn efficient latent representation in time series combined with basic supervised classifier for predicting time series and using it for anomaly detection task have not been proposed before.

This paper is organised as follows: Section 3 describes our approach. In Section 4, we present temporal anomaly detection results on real-world dataset using our combined LSTM autoencoder approach as well the comparison with state of art approaches.

III. The proposed Hybrid Approach for Anomaly Detection

a) ***Long Short-Term Memory Network (LSTM):*** is a type of Recurrent Neural Network with sophisticated memory cells allowing the model to remember long term dependencies and forget insignificant information. As mentioned in the previous section, LSTM was proposed to overcome RNNs limits [21]. RNNs are a class of ANN models that are recurrent in the sense that they possess an internal state or short-term memory with recurrent feedback connections. This implies maintaining for each time step an activation parameters vector. However, if they are trained with SGD (stochastic gradient descent iterative method), it would be more difficult to learn long-term dependencies due to the vanishing gradient problem [21]. To address this issue, a more sophisticated neuron (called also "cell") was proposed. It integrates regulatory structures called **gates** that gives to the network the ability to remove or add information to the cell state, allowing it to learn long-term dependencies.

Given a sequence with a fixed T-length time window: $X = [x_1, .., x_t, .., x_T]$, where $x_t \in R_n$ is an n-featured input at time step t, a Recurrent Network maintains a hidden state vector $h_t \in R_h$ that resumes the sequence information from the current input x_t and the previous hidden state h_{t-1}. LSTM cells integrate new components called "gates" to produce a more refined hidden state. We have the input gate, the output gate and the forget gate. The hidden state vector h_t is computed as follows:

$$i_t = \sigma(W_{xi}^T * x_t + W_{hi}^T * h_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(W_{xf}^T * x_t + W_{hf}^T * h_{t-1} + b_f) \tag{2}$$

$$o_t = \sigma(W_{xo}^T * x_t + W_{ho}^T * h_{t-1} + b_o) \tag{3}$$

$$g_t = tanh(W_{xg}^T * x_t + W_{hg}^T * h_{t-1} + b_g) \tag{4}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes g_t \tag{5}$$

$$\Longrightarrow h_t = o_t \otimes tanh(c_t) \tag{6}$$

Where :

- W are the Neural Network weights, b are the bias terms, σ is the sigmoid function and $\otimes$ represents element-wise product operator. operator
- For a given time-step t we have: i_t: input gate, f_t: forget gate, o_t: output gate and c_t: cell activation vector.

LSTM is able to recognize important incoming patterns thanks to the *input gate*, besides, it detects noisy information that will be omitted from the cell state via *forget gate*, and finally the *output gate* is associated with filtering operations to obtain more relevant result. With such structure, LSTM cells outperform regular cells at capturing long-term patterns in time series.

b) ***LSTM Autoencoder*** *::* Similarly to the word embedding, in which words are represented in a continuous space where semantic relationships between words are preserved, the LSTM Autoencoder realizes the sequence-embedding: in fact, the Encoder learns an internal representation of the input sequence that captures the latent information within a fixed-sized vector called encoding-vector. This last will be used as input to the Decoder that will try to reproduce the input

sequence with a reconstruction error. This model has shown good results in video representation [22], machine translation [23]. In short, the overall process of an LSTM Autoencoder is designed to read the input sequence, encode it, decode it and recreate it. The performance of the model is evaluated based on its ability to recreate the input sequence.
Considering a time widow T, a sequence of T transactions is represented by a vector X= $\{x_1,..,x_T\}$ where $x_t \in R_n$ is the transaction record at time step t.
1. Encoder: It encodes the transaction sequence X with an LSTM Neural Network: for a given time-step t the corresponding hidden vector of the encoder h_t^{en} is a function of x_t, h_{t-1}^{en}:

$$h_t^{en} = f_{en}(x_t, h_{t-1}^{en}) \tag{7}$$

The sequence input will be truncated with a fixed time window T so h_T^{en} captures the information of whole sequence.

2. Decoder: it is a Neural Network that adopts the T-length sequence representation h_T^{en} as the input to reconstruct the original sequence:

$$h_t^{dec} = f_{dec}(h_T^{en}, h_{t-1}^{dec}) \tag{8}$$

where h_t^{dec} is the hidden state vector of the decoder at time step t.
We consider the combination of both the encoder and the decoder as an overall Neural Network with an activation function F so the reconstructed transaction $\hat{x}_t$ at time step t is given as follows:

$$\hat{x}_t = F(h_t^{dec}) \tag{9}$$

The loss function of our model is expressed by the Euclidean distance between the generated vector and the original sequence input:

$$Loss = \sum_{t=1}^{T} (\hat{x}_t - x_t)^2 \tag{10}$$

In this work, we omit the decoder once the whole model has reached the desired performance, and keep the encoder part which encodes input sequences into a fixed length vector. The resulting vectors are used as input, to another supervised learning model, as an efficient representation of the input sequence. Normal and fraudulent transaction sequences would be encoded in separate regions in the internal latent continuous space.

A. *LSTM Autoencoder for Anomaly Detection*

In the raw form of our input data, normal and fraudulent transaction sequences are highly correlated, which makes the classification task very hard for classic classifiers based approaches to distinguish between the two classes. To deal with this problem, we propose to train beforehand an LSTM Autoencoder on normal sequences in order to capture efficient latent representations. And then, we propose to keep the pretrained encoder part to map normal and fraudulent transactions to separate regions in the latent continuous feature space. This representation is then provided as input to a binary classifier that would easily distinguish between the two classes. The overall process can be described in two main steps (See Fig.1):

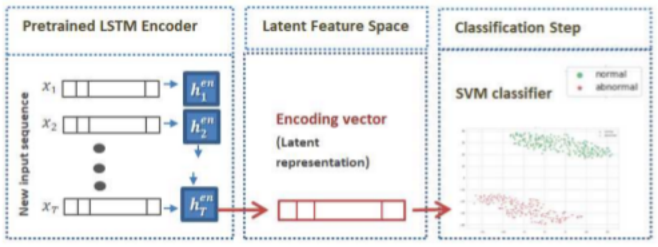

Fig. 1. The overall process : latent representations computation with LSTM Encoder and SVM based classification.

Step 1:
Given a training data set E_{normal} that contains labeled normal and abnormal transaction sequences, we first train the LSTM Autoencoder model with normal data sequences (Fig. 2) as detailed in the algorithm 1. For comparison, We have implemented both a dense Neural Network and an LSTM Autoencoders, and as expected the LSTM Autoencoder has learned much better representation encoding.

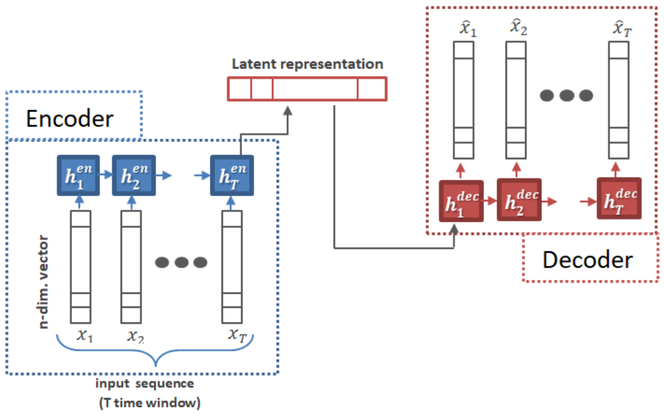

Fig. 2. Time Series Data Encoding based LSTM Autoencoder.

Step 2: Once the Autoencoder finished the training phase, we omit the decoder and we keep the encoder to make sequence representation of new records, that will be fed as input for a binary classifier. In our work, we tested several implementations of SVM classifiers as detailed in the pseudo-code of the algorithm 2.

IV. Experiments

A. *Experimental Dataset*

We have evaluated the proposed hybrid model on a labeled dataset containing time series records[1] composed of time-stamped credit card transactions which have occurred

[1]https://www.kaggle.com/mlg-ulb/creditcardfraud

Algorithm 1:

```
Result: Trained LSTM-Autoencoder
Inputs:
  • Supervised Training dataset E=($\{x_1,..,x_N\},\{y_1,..y_N\}$):
    with $y_i$=0 if $x_i$ is a normal transaction, $y_i$=1 otherwise
Parameters:
  • Training time window T
  • Training epoch for LSTM-Autoencoder epochs
Begin
  Initialize LSTM parameters
  Extract normal subset $E_{normal}$= $\{x_i$, for i in $[1, N]$ ,
  if $y_i$=0$\}$
  $i \leftarrow 1$ while $i \leq epochs$ do
      Select random T-length sequence from E$_{normal}$
      Forward propagation : reconstruct the sequence
      Backward propagation algorithm minimizing the
      loss function (10)
end
End
```

Algorithm 2:

```
Result: Trained classifier
Inputs:
  • Supervised Training dataset $E$=($\{x_1,..,x_N\}$,$\{y_1,..y_N\}$):
    with $y_i$=0 if $x_i$ is a normal transaction, $y_i$=1 otherwise
  • Trained LSTM Autoencoder
Begin:
  Randomly split $E$ to $trainSubset$ and $testSubset$
  $U_{train}$=$\{\}$
  for each $x$ in $trainSubset$ do
    • compute the latent representation vector $v_x$ of $x$ using
      the encoder
    • $U_{train}$ = $U_{train}$+ $v_x$
end
  Initialize classifier
  Train the classifier on $U_{train}$
  $U_{test}$=$\{\}$
  for each $y$ in $testSubset$ do
    • compute the latent representation vector $u_y$ of $y$ using
      the encoder
    • $U_{test}$ = $U_{test}$+ $u_y$
end
  Test the classifier on $U_{test}$
  Evaluate classification performances
  End
```

in September 2013 by european cardholders. It contains only numerical input variables which are the result of a PCA transformation for confidentiality issues. But the original values of the time and the Amount of transactions were preserved. The target feature takes value 1 in case of fraud and 0 otherwise. This dataset is highly skewed with only 0,1% of ”1” tagged data inputs.

B. Experimental Setup

We have implemented both dense (classic fully connected NN) and LSTM Autoencoder via Keras framework with tensorflow back-end using Python 3.5.6 by optimizing the networks using *Adelta*. For the LSTM Autoencoder the chosen dimension of the hidden layer (which is also the latent feature space dimension) is set to 150, the time window length T is set to 10, the chosen value for training epoch is 20 and the batch size is set to 256. We carry out extensive experiments to set empirically these values which maximize our model performance. The main difference between dense and LSTM Autoencoder at the implementation level, is that in the second one, the input has to be 2D-reshaped, in fact, it takes a sequence of $T * n - dim$ transaction records, whereas for the dense Autoencoder, each input is $1 * n$ dimensional record.

C. Experimental Results

We have trained both dense Autodencoder and LSTM Autoencoder on our training dataset. Fig. 3. shows the 2D visualisations using T-distributed Stochastic Neighbor Embedding (t-SNE) on raw data, we notice that the two classes are highly correlated, and applying a binary classifier on that form of data is doomed to failure, fig. 4. corresponds to the latent representation of the data obtained with a dense Neural Network Encoder: we note that the two classes are more separated, in fact, we can distinguish more clearly the two classes and finally fig. 5. shows the latent representation of the sequences obtained with the LSTM Encoder: the efficient LSTM data encoding has distinguish normal and fraudulent transactions, in two diagonally-opposite regions which makes the two classes straightly separable. This last encoding do best discriminative encoding power compared to the dense Neural Network representations.

For the classification step, we tried several SVM classifiers implementations with three different optimizers provided by sklearn library which are:

- SVC-Support Vector Classification : The implementation is based on libsvm.
- LinearSVC which is based on liblinear library: it constitute a better choice for large numbers of samples than the first implementation.
- Stochastic Gradient Descent (SGD) is a general optimization method which can optimize many different convex-optimization problems.

Furthermore, we have trained several classifiers on different representations of our data. To evaluate the different performances, all the experimented models has been trained on 80% of the dataset and evaluated on the remaining

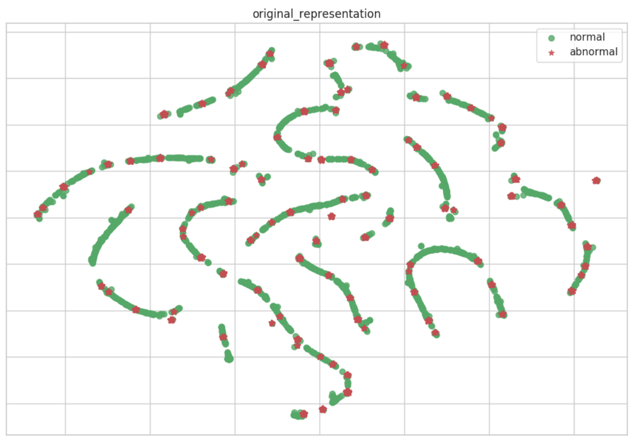

Fig. 3. Original data representation : strong correlation between the two classes (normal and fraud)

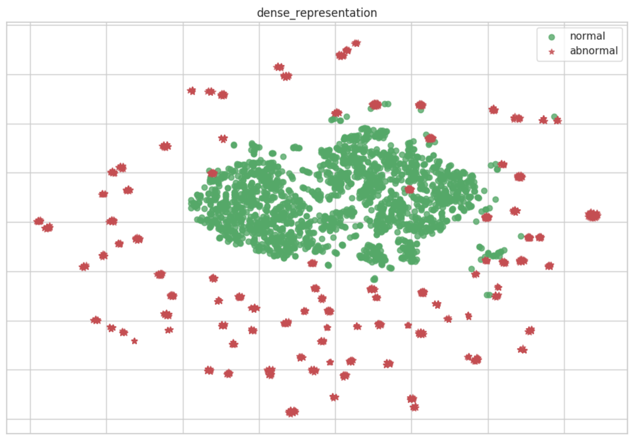

Fig. 4. Data Encoded with a pre-trained dense neural network encoder.

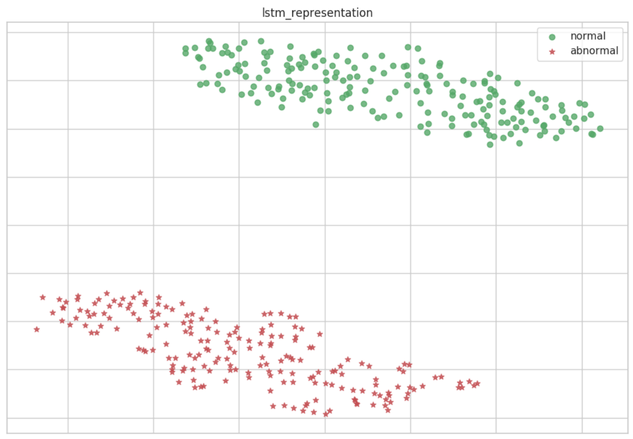

Fig. 5. Data Encoded with a pretrained LSTM Encoder

20%. The train/test sampling was stratified along the target variable in such a way that the fraudulent and normal transactions proportions are kept in both train and test samples. We compute the Precision, Recall and F1 score to evaluate classification performances, applied on the different representations of our data : the overall results are resumed in table I. We can notice that there is a significant improvement with the second representation (b) : f1 score increased from 0.25 to 0.9 for linear SVC optimizer and from 0.61 to 0.9 for SGD. However, the best representation in terms of classification performances is given by the LSTM encoding (c), in fact, the SVM classification score has increased from 0.61 to 0.98. But the difference was more significant with the use of SVM classifier based on SGD optimizer: f1 score measure has increased to 0.99 with the last encoding: it was impossible for the classifier with SGD optimizer to dicriminate between classes at the raw form of our data (a), meanwhile it has been much more easier at the LSTM encoded representation.

The obtained results show that the transactions sequences representations given by LSTM autoencoder was able to capture the salient latent information about transactions sequences, which improved the performances of basic classifiers.

TABLE I
CLASSIFICATION ERRORS OF THREE CLASSIFIERS APPLIED SEPARATELY AT THE ORIGINAL DATA INPUT (A) THEN AT THE SAME DATA ENCODED WITH A CLASSIC PRE-TRAINED NEURAL NETWORK ENCODER(B) AND FINALLY AT THE ENCODING OBTAINED THROUGH THE PRE- TRAINED LSTM ENCODER(C)

Input Representation	**Metric**	**SGD**	**LinearSVC**	**SVC**
(a) Original data	Precision	0	0.45	0.49
	Recall	0	0.17	0.80
	F1-score	0	0.25	0.61
(b) Dense encoder	Precision	0.84	0.90	0.90
	Recall	0.93	0.91	0.912
	F1-score	0.88	0.9	0.90
(c) LSTM encoder	Precision	0.99	0.99	1
	Recall	0.995	0.96	0.97
	F1-score	0.997	0.98	0.98

D. Comparison with state-of-the-art outlier detection algorithms

The proposed method is compared with several state-of-the-art models (One Class SVM ”OCSVM”, Isolation Forest and Local Outlier Factor ”LOF”). Table II shows that we have obtained poor performances with isolation forest (f1-score of 0.66) and with LOF (f1-score of 0.35) whereas with our hybrid model using SGD optimizer for the SVM classifier we have obtained a score of 0.99.
We also evaluated other non outlier detection oriented algorithms, the experimented models are the following:

- **Logistic regression** classifier: [24] with L2 norm penalty and an inverse regularization strength constant set to 2
- **Random forest** [25] with 50 estimators and maximum depth equals to 3. this model has been implemented with a class weighting technique [26] to overcome the heavy data skewness by giving weights inversely proportional to each class. This technique changes the weight that each

class has when the Random forest algorithm calculates the gini score of a chosen split point. Fraudulent transactions were weighted by 10 against 1 for normal entries.
- **LightGBM** [2] model which is basically a gradient boosting framework based on tree learning algorithms [27] with 100 estimators

The used hyper-parameters of each model are fixed with random search technique [28]. Table III illustrates the obtained performances. It shows that Random forest coupled with weighting method has shown better results than the others. This is due to the "gini" scores calculation at each split, is biased in favor of the minority class (fraudulent transactions), allowing with this way some false positives for the majority class (normal transactions).

TABLE II
COMPARISON WITH STATE-OF-THE-ART METHODS: THE PROPOSED HYBRID MODEL (LSTM & SVC) AND (LSTM & SGD) HAVE SHOWN BETTER RESULTS COMPARED STATE-OF-THE-ART METHODS.

Metric	Isolation Isolation	LOF	OCSVM	Hybrid (LSTM + SVC)	Hybrid (LSTM + SGD)
Precision	0.50	0.35	0.82	1	0.99
Recall	0.95	0.35	0.42	0.97	0.995
F1-Score	0.66	0.35	0.56	**0.98**	**0.997**

TABLE III
RESULTS OF OTHER NON OUTLIER DETECTION ORIENTED MODELS

Algorithm	LogiticRegression	RandomForest	LightGBM
Precision	0.006	0.70	0.12
Recall	0.125	0.88	0.50
F1-Score	0.012	0.77	0.19

V. CONCLUSIONS

In this paper, we proposed an hybrid approach for anomaly detection in time series by combining an LSTM Autoencoder with a linear classifier. Our model inherits from the Autoencoders the ability of learning efficient representations, and from the LSTM Neural Networks the sequence embedding. The latent representation based on LSTM Autoencoder was so efficient that we don't need any complex model with high computational complexity for the classification task, a linear classifier is sufficient to achieve high classification performances. Experiments were conducted on real world dataset, and the comparison with the state-of-the-art methods showed that our hybrid model outperforms the others in term of classification rate. In future work, we plan to tackle another complex issue in anomaly detection task which is the problem of the unbalanced dataset between normal and abnormal class consisted of rare outliers. The source code of our approach is publicly available on github[3]. It was furthermore extended as a web application that can also be provided.

[2]https://github.com/microsoft/LightGBM

[3]https://github.com/Athena75/hybrid_lstm_ijcnn

REFERENCES

[1] M. Y. Larry M. Manevitz, "One-class document classification via neural networks," *Neurocomputing*, vol. 70, p. 1466–1481, 2007. [Online]. Available: https://doi.org/10.1016/j.neucom.2006.05.013

[2] R. T. N. J. S. L. Markus M. Breunig†, Hans-Peter Kriegel, "Lof: Identifying density-based local outliers, in acm sigmod record," *Procedia Manufacturing*, vol. 29, p. 93–104, 2000.

[3] F. T. L. . K. M. T. . Z.-H. Zhou, "Isolation forest," *Eighth IEEE International Conference on Data Mining*, 2008.

[4] G. S. P. A. Pankaj Malhotra, Lovekesh Vig, "Long short term memory networks for anomaly detection in time series," *In European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015.

[5] M. A. T. M. J. Leonard, "Time series feature extraction," 2018.

[6] J. S. Sepp Hochreiter, "Long short-term memory," *Neural Computation*, vol. 9, p. 1735–1780, 1997.

[7] Q. V. L. Ilya Sutskever, Oriol Vinyals, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*. NIPS Montreal, 2014.

[8] S. O.Vinyals, A.Toshev and D.Erhan, "Show and tell: A neural image caption generator," 2015.

[9] G. Alex., "Generating sequences with recurrent neural network," 2014.

[10] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," *JMLR: Workshop and Conference Proceedings*, vol. 27, p. 37–50, 2012.

[11] J. L. J. H. G. L. Changfan Zhang, Xiang Cheng, "Deep sparse autoencoder for feature extraction and diagnosis of locomotive adhesion status," *Procedia Manufacturing*, 2018. [Online]. Available: https://doi.org/10.1155/2018/8676387

[12] A. B. G. A. S. Marco Maggipinto, Chiara Masiero, "A convolutional autoencoder approach for feature extraction in virtual metrology," *Procedia Manufacturing*, vol. 17, pp. 126–133, 2018.

[13] S. O. Ian J. Goodfellow, Jean Pouget-Abadie, "Generative adversarial nets," 2014.

[14] O. E. J. B. H. C. Thomas Blaschke, Marcus Olivecrona, "Application of generative autoencoder in de novo molecular design," *Journal of Cheminformatics*, 2017. [Online]. Available: https://doi.org/10.1186/s13321-018-0286-7

[15] L. T. U. M. R. I. Mohiuddin Ahmed, Abdun Mahmood, "A survey of anomaly detection techniques in financial domain," 2015.

[16] S. S. Girik Pachauri, "Anomaly detection in medical wireless sensor networks using machine learning algorithms," *Procedia Computer Science 70*, vol. 70, p. 325 – 333, 2015.

[17] B. S. R. W. A. S. J. S.-T. J. Platt, "Support vector method for novelty detection," *In NIPS'99 Proceedings of the 12th International Conference on Neural Information Processing Systems*, 1999.

[18] A. B. Andrea Borghesi, "Anomaly detection using autoencoders in high performance computer systems," 2018.

[19] G. A. L. V.-P. A. G. S. Pankaj Malhotra, Anusha Ramakrishnan, "Lstm-based encoder-decoder for multisensor anomaly detection," *TCS Research*, 2016.

[20] A. K. M. S. C. Raghavendra Chalapathy, "Anomaly detection using one-class neural networks," 2019.

[21] P. F. R. C.-A. K. M. S. C. Y. Bengio, P. Simard, "Learning long-term dependencies is difficult," *IEEE*, vol. 5, no. 2, p. 15 –166, 1997.

[22] E. M. Nitish Srivastava and R. Salakhutdinov, "Unsupervised learning of video representations using lstms," *ICML'15 Proceedings of the 32nd International Conference on International Conference on Machine Learning*, vol. 37, pp. 843–852, 2015.

[23] e. a. Kyunghyun Cho, et al., "Learning phrase representations using rnn encoder–decoder for statistical machine translation," *Association for Computational Linguistics*, p. 1724–1734, 1997.

[24] J. Brzezinski, "Logistic regression modeling for context-based classification," *Proceedings. Tenth International Workshop on Database and Expert Systems Applications. DEXA 99*, 1999.

[25] L. Breiman, *Random Forests*, 2001, vol. 45.

[26] A. L. Chao Chen and L. Breiman, "Using random forest to learn imbalanced data," 2004.

[27] G. K. Q. M. T. F. T. W. W. C. W. M. Q. Y. T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems 30 (NIP 2017)*, 2017.

[28] Y. B. James Bergstra, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research 13*, 2012.