Unsupervised Anomaly Detection in Time Series Using LSTM-Based Autoencoders

Oleksandr I. Provotar
Faculty of Computer Science and Cybernetics
Taras Shevchenko National University of Kyiv
Kyiv, Ukraine
aprowata@unicyb.kiev.ua

Yaroslav M. Linder
Faculty of Computer Science and Cybernetics
Taras Shevchenko National University of Kyiv
Kyiv, Ukraine
yaroslav.linder@gmail.com

Maksym M. Veres
Faculty of Computer Science and Cybernetics
Taras Shevchenko National University of Kyiv
Kyiv, Ukraine
veres@unicyb.kiev.ua

***Abstract* — Automatic anomaly detection in data mining has a wide range of applications such as fraud detection, system health monitoring, fault detection, event detection systems in sensor networks, and so on. The main challenge related to such problem is unknown nature of the anomaly. Therefore, it is impossible to use classical machine learning techniques to train the model, as we don't have labels of time series with anomaly. For periodic time series it is advisable to use STL decomposition of the signal. In such case anomaly detection task is reduced to residuals peak detection. If time series is not periodic (for example, forex price or sound) the only way is using machine learning methods. One of the best machine learning methods is autoencoder-based anomaly detection. An autoencoder is a type of artificial neural network used to learn efficient data encodings in an unsupervised manner. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore signal "noise". Unsupervised anomaly detection method based on autoencoders was tested on two types of data: various artificial signal datasets and detection of rare sound events dataset.**
(*Abstract*)

Keywords— machine learning, anomaly detection, neural network, autoencoders (*key words****)***

I. INTRODUCTION

Also known as outlier detection, anomaly detection is a data mining process used to determine types of anomalies found in a data set and to determine details about their occurrences. Automatic anomaly detection is critical in today's world where the sheer volume of data makes it impossible to tag outliers manually. Auto anomaly detection has a wide range of applications such as fraud detection, system health monitoring, fault detection, and event detection systems in sensor networks, and so on.

The main challenge related to such problem is unknown nature of the anomaly. Therefore, it is impossible to use classical machine learning techniques to train the model, as we don't have labels of time series with anomaly.

There are several classical methods of anomaly detection, such as

- Clustering-Based Anomaly Detection;
- Isolation Forests;
- Support Vector Machine;
- Anomaly Detection using Gaussian Distribution.

However, all these methods choose the data point as an outlier only based on its value, but not based on values of previous points. In other words, such methods don't take into consideration temporal nature of the data.

Therefore, classical methods often don't give good results. Among algorithms that are used to detect anomalies in time-series data one can mention

- Anomaly detection based on signal decomposition (classical decomposition, STL) [1]
- State space models: exponential smoothing, Holt-Winters, ARIMA
- Deep learning: autoencoders based on feedforward, recurrent and LSTM neural network layers [2]
- Dimensionality reduction: RPCA, SOM, discords, piecewise linear

In case when time series is periodic, the best method is decomposition method: first, we remove seasonal and trend component from the signal, and then use one of the classical methods for outlier detection. For example, such method works fine on hotel room price data which has clear yearly fluctuations. Also, prices slowly rise year from year because of the inflation. Thus, after removing seasonal and trend component we can consider data point as an outlier if it lays far enough from the zero line.

But in case when time series does not have a clear period (for example, forex price or sound) the only way is using machine learning methods. One of the best machine learning methods is autoencoder-based anomaly detection.

An autoencoder is a type of artificial neural network used to learn efficient data encodings in an unsupervised manner. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore signal "noise". Along with the reduction side, a reconstructing side is learnt, where the autoencoder tries to generate from the reduced encoding a representation as close as possible to its original input, hence its name.

Autoencoders are often trained with only a single layer encoder and a single layer decoder, but using deep encoders and decoders offers many advantages.

- Depth can exponentially reduce the computational cost of representing some functions.
- Depth can exponentially decrease the amount of training data needed to learn some functions

IEEE ATIT 2019 ISBN 978-1-72-81-61-44-0/19/$31.00©2019 IEEE

• Experimentally, deep autoencoders yield better compression compared to shallow or linear autoencoders.

An autoencoder is made of two modules: encoder and decoder. The encoder learns the underlying features of a process. These features are typically in a reduced dimension. The decoder can recreate the original data from these underlying features.

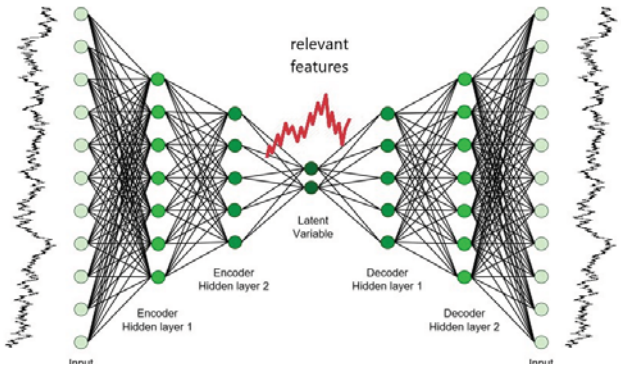


Fig. 1. Conceptual example of autoencoder

It is possible to build autoencoder based on feedforward neural network. However, to take into account temporal data we will build autoencoder based on LSTM layers. Unlike feedforward neural network, we put information into LSTM consequentially, one number at a time. Each LSTM unit is a generalization of RNN unit, so part of information about previous time series values is stored into the neural network.

Each LSTM unit has three gates

- forget gate, which is responsible for keeping the fraction of information from previous states;
- output gate, which is responsible for choosing how much of an information we output;
- input gate, which is responsible for getting new information.

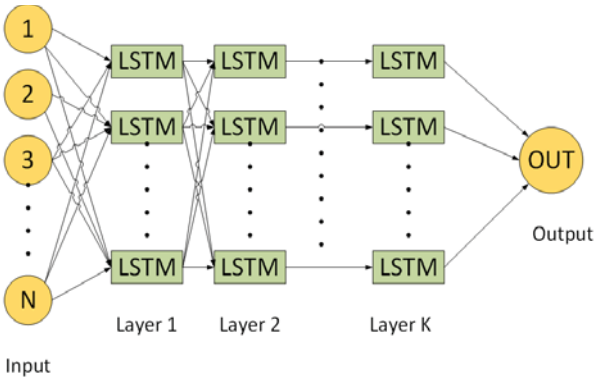


Fig. 2. Neural network architecture based on LSTM layers

Input sequence goes into neural network consequentially, one number at a time. In our case dense layer consists only of one neuron. Neural network keeps running during N+K time steps (to allow the information from last input reach the output neuron).

II. ANOMALY DETECTION BASED ON ONE CLASS SVM ALGORITHM

Predicted	Actual	
	Anomaly	*No anomaly*
Anomaly	44	38
No anomaly	6	12

To cope with the problem of anomaly detection, one-class classification problems (and solutions) are introduced. By just providing the normal training data, an algorithm creates a (representational) model of this data. If newly encountered data is too different, according to some measurement, from this model, it is labeled as out-of-class. One class SVM is an algorithm that builds "boundary" around the majority of the samples. Samples that are located outside of the boundary are labelled as anomalies [5].

We test this algorithm on audio rare sound test event database. As raw audio has too high dimension (one second of audio consists of 24000 samples), it is impossible to use one class SVM on this data without its conversion (dimensionality reduction). So, we calculate the spectrum of each chunk of the data using n-point discrete Fourier transform. Then we pick from the transform those amplitudes that corresponds to the frequency range that we are interested in (frequencies of anomalies). After that we use this numbers to train one class SVM model

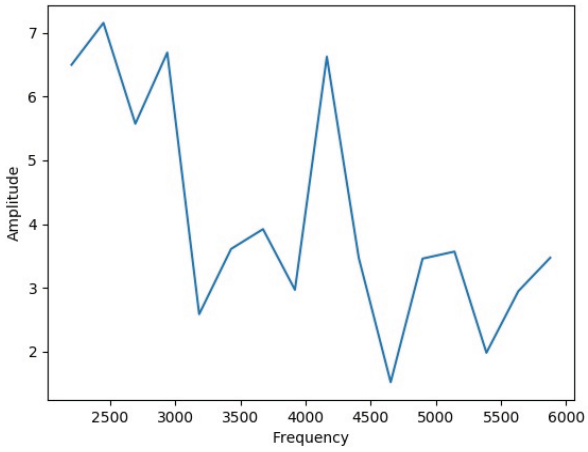


Fig. 3. Example of discrete Fourier transform result

We trained the data on 500 audio samples with 30 sec length and 50% probability of anomaly in it. After the training we tested the trained model on 100 files that were generated from different background files and event files. We used only one type of anomaly, which is gunshot.

We used audio down sampling to 24000 samples per second. We used the moving window of size 12000 with moving step is also 12000 (no overlap). We applied 50-point discrete Fourier transform with target frequency range from 2KHz to 6KHz.

Each 30-sec test file consists of 60 parts, each with length of 12000 samples, or 0.5 sec. We say that the file has an anomaly if maximum anomaly score for all these 60 parts is bigger than threshold. The optimal value of threshold can't be selected based on training data (because we have

unsupervised learning and cannot use the actual positions of anomalies). So, the optimal threshold was selected based on the same testing data.

TABLE I. CONFUSION MATRIX OF ONE CLASS SVM ANOMALY DETECTOR

Accuracy on the training set varies from 55% to 65% depending on target frequency range, window width and number of points that are used in discrete Fourier transform. With such a low accuracy we even didn't check if the exact positions of actual and predicted anomalies coincides. It is obvious that the result of one class SVM anomaly detector in case of audio data is only slightly better than random coin toss.

III. ANOMALY DETECTOR BASED ON LOESS DECOMPOSITION

Time series are usually decomposed into:

T_t – the trend component at time t, which reflects the long-term progression of the series (secular variation). A trend exists when there is a persistent increasing or decreasing direction in the data. The trend component does not have to be linear.

S_t – the seasonal component at time t, reflecting seasonality (seasonal variation). A seasonal pattern exists when a time series is influenced by seasonal factors. Seasonality occurs over a fixed and known period (e.g., the quarter of the year, the month, or day of the week).

I_t – the irregular component (or "noise") at time t, which describes random, irregular influences. It represents the residuals or remainder of the time series after the other components have been removed.

Hence a time series using an additive model can be thought of as

$$y_t = T_t + S_t + I_t$$

The algorithm that is based on loess decomposition is as follows

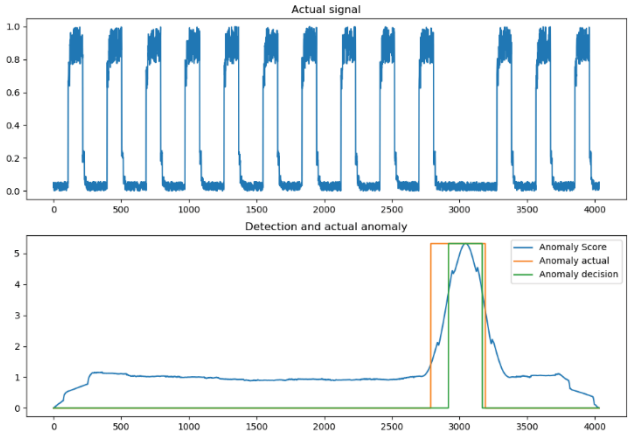
1) We determine the dominant frequency of the signal using one of the two methods: autocorrelation or Welch's power spectral density estimation
2) Then we decompose signal into seasonal, trend and residual component
3) We calculate absolute value of residual and smooth it with the moving window parameter equal to obtained dominant frequency.
4) We normalize the result by dividing it by its standard deviation, thus we got vector P of transformed and smoothed residuals
5) We decide whether we have an anomaly at time t if

$$P(t) - \frac{1}{T}\sum_{i=1}^{T} P(i) > threshold$$

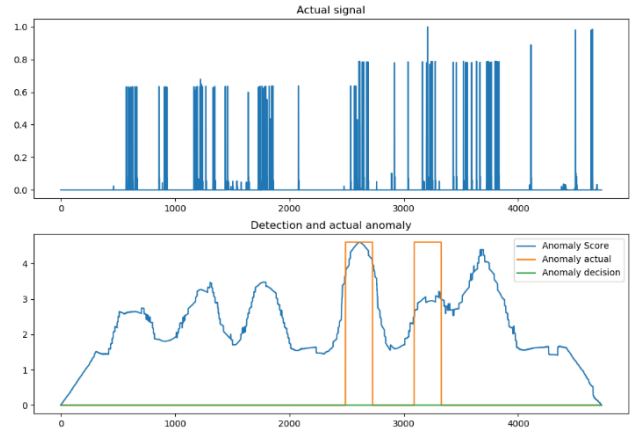
This algorithm is useless on audio data, because it has no trend, and it consists of multiple different frequencies, thus it is often impossible to extract the one "dominant" frequency. Therefore, we tested the algorithm on artificial data which consists of 58 files with time series data of different nature.

On some files algorithm gave decent results, while on other completely fails. Also, there is a lot of internal parameters that is needed to be chosen manually to make it work. So, if the task is to find the anomaly on one file, it is easy to manually select parameters that allows us to correctly detect the anomaly. However, it is impossible to find set of parameters that will work equally well on all files. For example, with frequency method set to autocorrelation, we have such result on two different files (first file correct detection, second file incorrect detection)

{'correctly detected:': 1, 'missed': 0, 'false detection': 0}

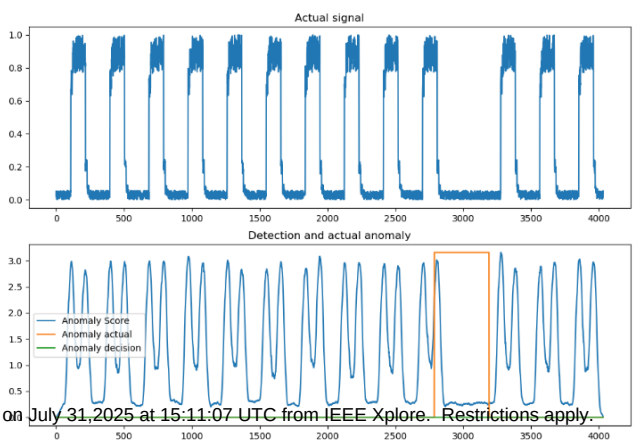


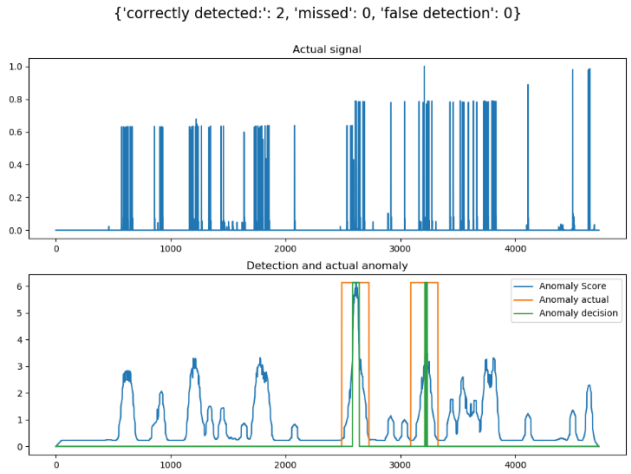
{'correctly detected:': 0, 'missed': 2, 'false detection': 0}



When we set method to Welch power spectral density, results are opposite: first file is detected incorrectly, while second is detected correctly.

{'correctly detected:': 0, 'missed': 1, 'false detection': 0}





So, we can choose this method of anomaly detection, only if

1) Signal structure doesn't change a lot (for example, we don't want a situation when initially the signal looks like right subfigure, and then switch to left subfigure)

2) Signal dominant frequency doesn't change over time

3) Signal anomalies have specific properties, that doesn't change over time.

Among the benefits of this algorithm one can mention its relative simplicity comparing to machine learning algorithm based on autoencoders. Among the disadvantages one can mention necessity of manual parameters choosing.

IV. Unsupervised Learning Autoencoder Algorithm

Proposed algorithm is based on the autoencoder machine learning method, but has several post-processing features that increases its detection abilities.

1) Split the training signals into smaller parts, using sliding window algorithm. Feed the signal into autoencoder and train it with labels equal to the same signal. During the training stage several variables need to be selected

a. Sliding window size to feed to autoencoder

b. Sliding window step. Increasing the step will lead to decrease of training set size and vice versa. So, we take slices from the signal [0…window size], [sliding step…window size + sliding step], [2*sliding step…window size+2*sliding step] etc.

c. Number of layers in neural network

d. Number of LSTM neurons in each layer

2) Split the testing signal into smaller parts, using sliding window algorithm. Feed the signal into trained autoencoder and observe its output. Combine the output into one time series.

3) (Optional) Adjust signal and output so that they fit each other in the best way (using cross correlation function). Then shift the signal and output so the peaks of signal are exactly above the peaks of the output.

4) Compute absolute value of difference between signal and output

5) (Optional) As each anomaly have some duration, we expect that residual peak will not be one-point. So, we use smoothing to get rid from accidental residual one-point peaks.

6) Training data set is used to adjust threshold, above which signal part is considered as anomaly. In case training data set is not labeled, we use 99.9% quantile for detecting anomaly threshold. Threshold method can be reviewed and updated based on actual data.

7) All residuals above the threshold are considered as anomalies.

V. Results

Unsupervised anomaly detection method based on autoencoders was tested on two datasets:

1) Various artificial signal datasets [3]

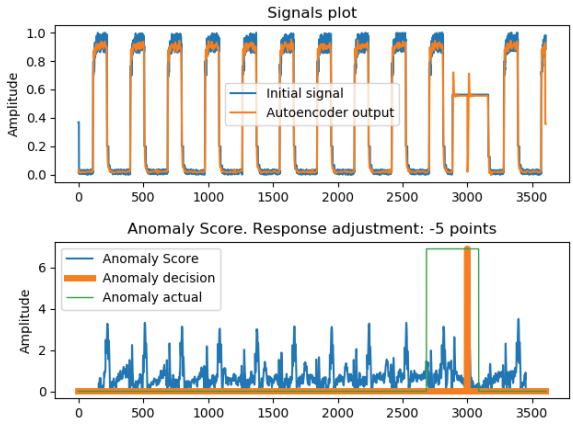
2) Detection of rare sound events dataset [4].

A. Artificial dataset set-up and results

For most of the datasets we choose window width equal to 600 and sliding step equal to 50. Each artificial signal has different structure, so it is considered as separate problem. Each file has around 4000 entries, so decreasing sliding step will increase training set size. With the above parameters training set size is equal to 69. It means we have 69 slices each of length 600.

In unsupervised anomaly detector neural network with 8 LSTM layers was used. Each layer consists of 5 LSTM units. We use average smoothing with sliding window parameter equal to 50 (see point 5 of the algorithm). Also, we use signal response adjustment. In other words, we try to fit response and signal by moving it relative to each other. (see point 3 of the algorithm)

As soon as we have only one time series for each problem, we can not use training set for adjusting threshold, above which we will consider part of the signal as anomaly.



Instead, we use 99% quantile to see the largest residuals and claim it as anomaly.

Fig. 4. Artificial dataset result example

In the fig.4 one can see the example of autoencoder graphical output. At the first subplot we can see actual and encoded signals, that are close to each other except the time of anomaly. We move response 5 points to the right to make best fit. We can see that anomaly score of autoencoder on anomaly is way above anomaly scores of other parts of the signal.

B. *DCASE dataset set-up and results*

DCASE dataset consists of 1121 environment files (30 sec. length each) and three types of rare sound events (gunshot, baby cry and glass break). However, we are using only gunshot sounds. In initial dataset rare sound events can be of three different sound levels: +6Db, 0Db and -6Db comparing to background sound. We are using only 0Db adjustment (event sound has same loudness as background). The reason of it is that with -6Db adjustment event become too quiet to be detected using autoencoder algorithms.

Sound data is down sampled to 16000 samples per second. Moving window is equal to 16000 (1 second) and moving step is same as moving window. In unsupervised anomaly detector neural network with 8 LSTM layers was used. Each layer consists of 15 LSTM units. We use average smoothing with 2000 sliding window parameter (see point 5 of the algorithm). As an optional feature it is possible to turn on signal-response adjustment, but the results with and without this feature are similar.

Total number of training set slices is 1121·30=33630. We train autoencoder on 100 epochs (use shuffled training data 100 times). During testing phase, we split each audio file (30 sec length) using moving window algorithm with moving window equal 16000 and sliding step equal to 4000. After getting the response we recombine the bunch of output slices to get one 30-sec. output time series.

During the testing phase we can set up threshold manually in json file, or evaluate it based on testing set labels. Threshold estimation method is simple: we sort all testing files from the lowest maximal anomaly score to the highest one. Then we calculate accuracies for all values of maximal anomaly score and select the highest one. Corresponding maximal anomaly score is set as threshold. So, we use testing set as validation one to select threshold hyperparameter. Therefore, accuracy for testing set is slightly higher than it might be if we select threshold based on other testing set.

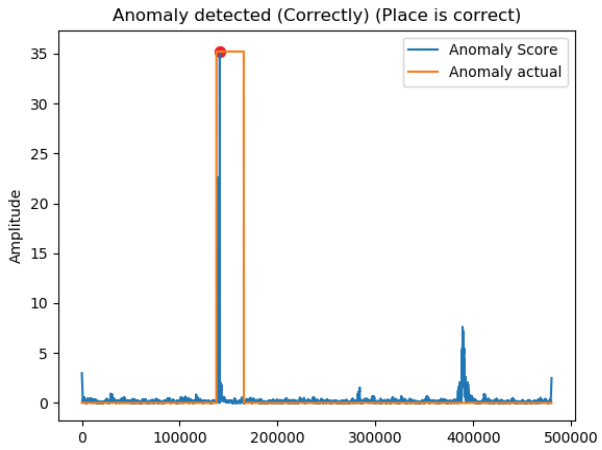


Fig. 5. DCASE dataset example

As it can be seen from the above figure, we detect not only the presence of event in 30 second slice, but also the exact position of it. Red dot is the sign that autoencoder recognizes corresponding part of the signal as anomaly. Orange bar is the position of actual anomaly.

TABLE II. CONFUSION MATRIX OF TESTING SET

Predicted	**Actual**	
	Anomaly	***No anomaly***
Anomaly	44	38
No anomaly	6	12

Confusion matrix can show not only the accuracy, but distribution of mistakes. As it is seen from the table, detector is more likely to make Type I errors rather than Type II errors. Accuracy is 87%, for correctly detected anomalies exact place is correct in 91.7% cases

VI. CONCLUSIONS

Neural network architecture and post-processing parameters need to be chosen empirically depending on the input data. The most important parameters are:

- Number of layers on neural network;
- Number of LSTM cells in each layer;
- Window size to fed into neural network;
- Smoothing window size;
- Threshold, above which we consider peak in residuals as anomaly.

Other methods may be useful for time series of certain nature, however only autoencoder approach is general and powerful enough to be used in all types of time series.

REFERENCES

[1] Robert B. Cleveland, William S. Cleveland, Jean E. McRae, Irma Terpenning. STL: A seasonal-trend decomposition procedure based on loss. Journal of Official Statistics, Vol 6., No. 1, 1990, pp. 3-73. http://www.nniiem.ru/file/news/2016/stl-statistical-model.pdf

[2] A Tutorial on Deep Learning Part 2: Autoencoders, Convolutional Neural Networks and Recurrent Neural Networks http://robotics.stanford.edu/~quocle/tutorial2.pdf

[3] Various artificial signal datasets, https://github.com/numenta/NAB/tree/master/data

[4] Detection of rare sound events dataset, http://dcase.community/challenge2017/task-rare-sound-event-detection#audio-dataset

[5] Bernhard Scholkopf, Robert Williamsonx , Alex Smolax , John Shawe-Taylory , John Plat. Support Vector Method for Novelty Detection, MIT Press (2000), pp. 582–588