International Journal of Latest Trends in Engineering and Technology
Vol.(16)Issue(4), pp.001-010
DOI: http://dx.doi.org/10.21172/1.164.01
e-ISSN:2278-621X

OUTLIER DETECTION FOR TEMPORAL DATA USING LONG SHORT-TERM MEMORY

Pratik .P. Sannakki[1], Srinand M.S[1] , Swathi N.Rao[1] , Asha .T[1]

Abstract- We explore the use of Long short-term memory (LSTM) for anomaly detection in temporal data. Due to the challenges in obtaining labeled anomaly datasets, an unsupervised approach is employed. We train recurrent neural networks (RNNs) with LSTM units to learn the normal time series patterns and predict future values. The resulting prediction errors are modeled to give anomaly scores. We investigate different ways of maintaining LSTM state, and the effect of using a fixed number of time steps on LSTM prediction and detection performance. LSTMs are also compared to feed-forward neural networks with fixed size time windows over inputs. Our experiments, with three real-world datasets, show that while LSTM RNNs are suitable for general purpose time series modeling and anomaly detection, maintaining LSTM state is crucial for getting desired results. Moreover, LSTMs may not be required at all for simple time series.

Key Words: Long Short Term Memory (LSTM), Recurrent Neural Network (RNN), Anomaly Detection, Time Series, Deep Learning

I. INTRODUCTION

Anomaly detection refers to the problem of finding instances or patterns in data that deviate from normal behavior. Depending on context and domain these deviations can be referred to as anomalies, outliers, or novelties. In this thesis, the term used is anomalies. Anomaly detection is utilized in a wide array of fields such as fraud detection for financial transactions, fault detection in industrial systems, intrusion detection, and artificial bot listeners identification in music streaming services. Anomaly detection is important because anomalies often indicate useful, critical, and actionable information that can benefit businesses and organizations.

Anomalies can be classified into four categories :

1. Point Anomalies
2. Collective Anomalies
3. Contextual Anomalies
4. Change Points.

These categories have a significant influence on the type of anomaly detection algorithm employed. Anomaly detection is considered to be a hard problem. Anomaly is defined as a deviation from normal pattern. However, it is not easy to come up with a definition of normality that accounts for every variation of normal pattern. Defining anomalies is harder still. Anomalies are rare events, and it is not possible to have a prior knowledge of every type of anomaly. Moreover, the definition of anomalies varies across applications. Though it is commonly assumed that anomalies and normal points are generated from different processes. Another major obstacle in building and evaluating anomaly detection systems is the lack of labeled datasets. Though anomaly detection has been a widely studied problem, there is still a lack of commonly agreed upon benchmark datasets. In many real-world applications anomalies represent critical failures which are too costly and difficult to obtain. In some domains, it is sufficient to have tolerance levels, and any value outside the tolerance intervals can be marked as an anomaly. Though in many cases labeling anomalies is a time-consuming process and human experts with knowledge of the underlying physical process are required to annotate anomalies. Anomaly detection for time series presents its own unique challenges. This is mainly due to the issues inherent in time series analysis which is considered to be one of the ten most

[1] *Department of Computer Science and Engineering , Bangalore Institute of Technology, Bangalore, Karnataka, India*

challenging problems in data mining research. In fact, time series forecasting is closely related to time series anomaly detection, as anomalies are points or sequences which deviate from expected values.

II. Relevant Theory

An RNN is a special type of NN suitable for processing sequential data. The main feature of an RNN is a state vector (in the hidden units) which maintains a memory of all the previous elements of the sequence. . As an RNN has a feedback connection which connects the hidden neurons across time. At time t, the RNN receives as input the current sequence element xt and the hidden state from the previous time step st−1. Next the hidden state is updated to st and finally the output of the network ht is calculated. In this way the current output depends on all the previous inputs x 0 t (for t 0 ≤ t). U is the weight matrix between the input and hidden layers similar to a conventional NN. W is the weight matrix for the recurrent transition between one hidden state to the next. V is the weight matrix for hidden to output transition. Equations below summarize all the computations carried out at each time step.

$$s_t = \sigma(Ux_t + Ws_{t-1} + b_s)$$
$$h_t = softmax(Vs_t + b_h)$$

The softmax given represents the softmax function which is often used as the activation function for the output layer in a multiclass classification problem. The softmax function ensures that all the outputs range from 0 to 1 and their sum is 1.LSTM can learn dependencies ranging over arbitrary long time intervals. LSTM overcomes the vanishing gradients problem by replacing an ordinary neuron by a complex architecture called the LSTM unit or block. An LSTM unit is made up of simpler nodes connected in a specific way. The main components of the LSTM architecture introduced in are:

1. Constant error carousel (CEC) : A central unit having a recurrent connection with a unit weight. The recurrent connection represents a feedback loop with a time step equal to 1. The CEC's activation is the internal state which acts as the memory for past information.
2. Input Gate: A multiplicative unit which protects the information stored in CEC from disturbance by irrelevant inputs.
3. Output Gate: A multiplicative unit which protects other units from interference by information stored in CEC.

The input and output gate control access to the CEC. During training, the input gate learns when to let new information inside the CEC. As long as the input gate has a value of zero, no information is allowed inside. Similarly, the output gate learns when to let information flow from the CEC. When both gates are closed (activation around zero) information or activation is trapped inside the memory cell. This allows the error signals to flow across many time steps (aided by the recurrent edge with unit weight) without encountering the problem of vanishing gradients. The problem of exploding gradients is taken care of by gradient clipping LSTM with Forget Gates The architecture of an LSTM unit with forget gates is shown in figure 2.1 and is the architecture used for the rest of this paper.

The main components of the LSTM unit are:

1. Input: The LSTM unit takes the current input vector denoted by xt and the output from the previous time step (through the recurrent edges) denoted by ht−1. The weighted inputs are summed and passed through tanh activation, resulting in zt .
2. Input gate: The input gate reads xt and ht−1, computes the weighted sum, and applies sigmoid activation. The result is multiplied with the zt , to provide the input flowing into the memory cell. 3. Forget gate: The forget gate is the mechanism through which an LSTM learns to reset the memory contents when they become old and are no longer relevant. This may happen for example when the network starts processing a new sequence. The forget gate reads xt and ht−1 and applies a sigmoid activation to weighted inputs. The result is multiplied by the cell state at previous time step i.e. st−1 which allows for forgetting the memory contents which are no longer needed.
4. Memory cell: This comprises the CEC, having a recurrent edge with unit weight. The current cell state is computed by forgetting irrelevant information (if any) from the previous time step and accepting relevant information (if any) from the current input.
5. Output gate: Output gate takes the weighted sum of xt and ht−1 and applies sigmoid activation to control what information would flow out of the LSTM unit.
6. Output: The output of the LSTM unit, ht , is computed by passing the cell state st through a tanh and multiplying it with the output gate, of equations:

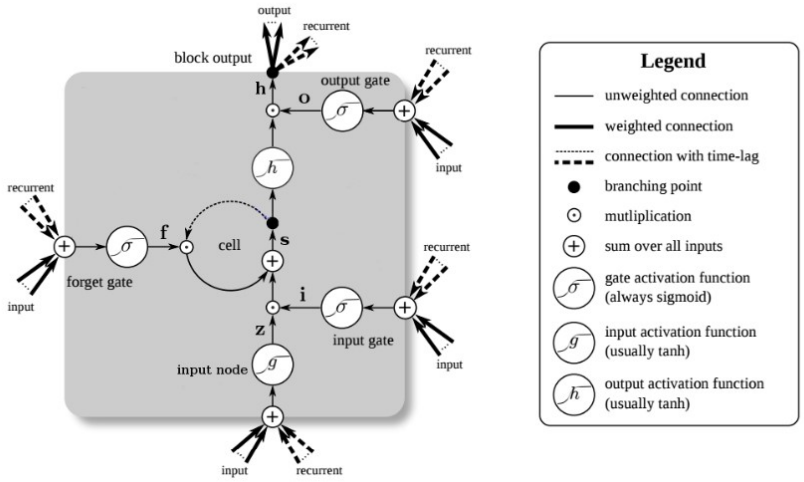

Figure 1. A LSTM unit with forget gates**.** A schematic diagram of the LSTM unit with forget gates

$$
\begin{aligned}
z_t &= tanh(W^z x_t + R^z h_{t-1} + b^z) && \text{(input)} \\
i_t &= \sigma(W^i x_t + R^i h_{t-1} + b^i) && \text{(input gate)} \\
f_t &= \sigma(W^f x_t + R^f h_{t-1} + b^f) && \text{(forget gate)} \\
o_t &= \sigma(W^o x_t + R^o h_{t-1} + b^o) && \text{(output gate)} \\
s_t &= z_t \odot i_t + s_{t-1} \odot f_t && \text{(cell state)} \\
h_t &= tanh(s_t) \odot o_t && \text{(output)}
\end{aligned}
$$

The W∗s are input weights, the R∗s are recurrent weights, and b∗s are the biases.

III. Methodology and Algorithm

A. Anomaly Detection Method –

The anomaly detection algorithm used in the project consists of two main steps. First, a summary prediction model is built to learn normal time series patterns and predict future time series. Then anomaly detection is performed by computing anomaly scores from the prediction errors.

B. Time Series Prediction –

We use LSTM RNN as the time series prediction model. The model takes as input the most recent p values and outputs q future values. We refer to parameters p, q as lookback and lookahead respectively. The network consists of hidden recurrent layer/layers followed by an output layer. The number of hidden recurrent layers and the number of units in each layer vary for each dataset. Two consecutive recurrent layers are fully connected with each other. To avoid overfitting dropout is used between two consecutive layers. The output layer is a fully connected dense NN layer. The number of neurons in the output layer is equal to the lookahead value, with one neuron for each future value predicted. Since the model is used for regression we use linear activation in the output layer and MSE as the loss function. The prediction model is trained only on normal data without any anomalies so that it learns the normal behavior of the time series.

Predicting Multiple Time Steps Ahead: We experiment with predicting multiple time steps into the future. With a lookahead of q at time t the model predicts the next q values of the time series i.e. t+1, t+2 ..., t+q. Predicting multiple time steps is done for two purposes. First, to showcase LSTM's capability as time series models, predicting multiple future values is a harder problem as compared to one-step-ahead prediction. Second, predicting multiple time steps provides an early idea of the future behavior. It could even be possible to get an early indication of an anomaly. Consider a time series with a scale of 5 minutes. Predicting 6 time steps ahead can give us an idea about

the behavior of the time series for the next 30 minutes. If there is something unusual, e.g. an extreme value, early alerts can be sent out. However, anomaly detection can happen only when the real input value becomes available. Predicting multiple time steps comes at the cost of prediction accuracy. In our experiments, we use a lookahead of greater than 1 only if the prediction accuracy is still acceptable. The actual lookahead value used is chosen arbitrarily.

C. Anomaly Detection –

Anomaly detection is done by using the prediction errors as anomaly indicators. Prediction error is the difference between prediction made at time t − 1 and the input value received at time t. The prediction errors from training data are modeled using a Gaussian distribution. The parameters of the Gaussian, mean and variance, are computed using maximum likelihood estimation (MLE). On new data, the log probability densities (PDs) of errors are calculated and used as anomaly scores: with lower values indicating a greater likelihood of the observation being an anomaly. A validation set containing both normal data and anomalies is used to set a threshold on log PD values that can separate anomalies from normal observations and incur as few false positives as possible. A separate test set is used to evaluate the model.

D. Algorithm –

The LSTM RNN is trained only on normal data to learn normal time series patterns and optimized for prediction accuracy. For this purpose, each dataset is divided into four subsets: a training set, N, with only normal values; validation set, VN , with only normal values; a second validation set, VA, with normal values and anomalies; and a test set, T, having both normal values and anomalies. The algorithm proceeds as follows:

1. Set N is used for training the prediction model. We used Bayesian optimization to find the best values for hyper-parameters: lookback, dropout, learning rate, and the network architecture (number of hidden layers and units in each layer). We use a lookahead of more than 1 only if the prediction accuracy is still reasonable. If predicting multiple time steps is not required and one needs the best prediction accuracy, lookahead can be set to 1.
2. VN is used for early stopping to prevent the model from overfitting the training data.
3. Prediction errors on N are modeled using Gaussian distribution. The mean and variance of the distribution are estimated using MLE.
4. The trained prediction model is applied on VA. The distribution parameters calculated in the previous step are used to compute the log PDs of the errors from VA. A threshold is set on the log PD values which can separate the anomalies, with as few false alarms as possible.
5. The set threshold is evaluated using the prediction errors from the test set T.

E. Dataset –

ECGs are time series recording the electrical activity of the heart.

ECG datasets available at PhysioBank's archive5 have been used for anomaly detection and among others. In this report we use the dataset used as it provides labeled anomalies annotated by a cardiologist. A snippet of a dataset showing normal patterns is shown.There are a total of 18000 readings with different kinds of anomalies. The three labeled anomalies identified as the most unusual sequences are shown Though this dataset has a repeating pattern, the length of the pattern varies.

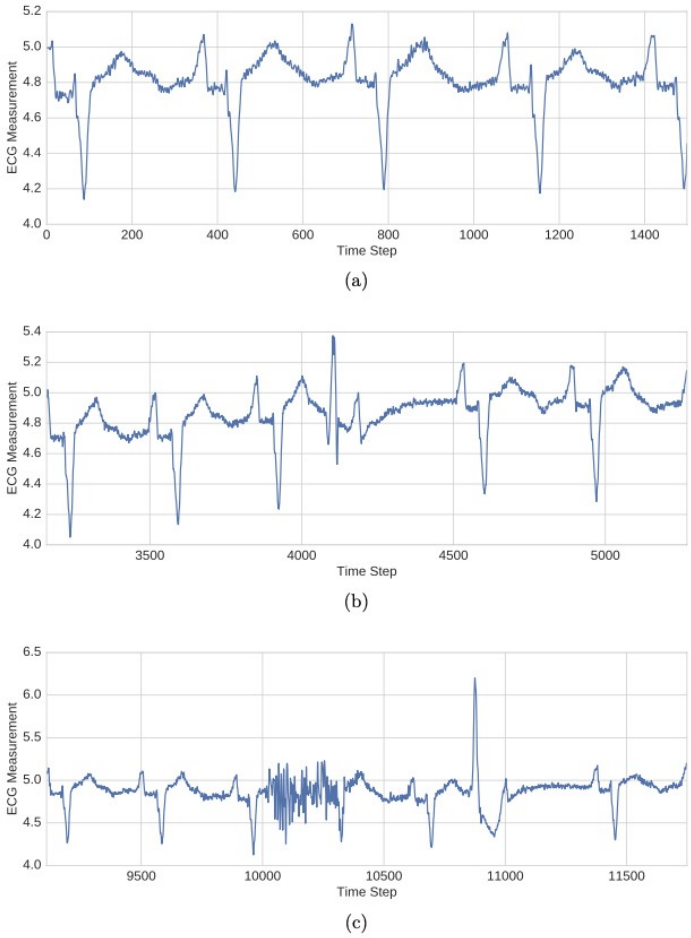

Figure 2. ECG Dataset. (a) shows a normal heartbeat pattern. (b) shows the first anomaly. (c) shows the other two anomalies. The X-axis shows time steps, while the Y-axis has the ECG measurements. These figures show snippets of the dataset so that the normal and anomalous heartbeats are easily visible

IV. Experiment and Result

A. Data Preprocessing –

For each dataset, the anomalies were divided into sets VA and T. These sets were then augmented with normal data. Datasets which had a repeating cycle were divided in such a way that the cycles remained intact. The remaining data was divided into sets N and VN . We also normalized the data to have zero mean and unit variance. The mean and standard deviation of the training set were used to normalize other sets. Finally, each set was transformed into the format required by the algorithm. So each input sample consisted of a lookback number of time steps.

B. ECG Dataset –

In the ECG dataset there are only three labeled anomalies out of 18000 samples. Thus we do not use set VA for finding threshold. Instead, the dataset was divided into a normal training set (N), a normal validation set (VN) for early stopping, and a test set (T) containing all three anomalies.

Model Details: The prediction model used was a stacked RNN consisting of two hidden recurrent layers with 60 and 30 LSTM units respectively, a dense output layer with 5 neurons, a lookback of 8, a lookahead of 5, and a dropout of 0.1. LSTM state was not maintained between batches. We trained the RNN using Adam optimizer with a learning rate of 0.1, a decay of 0.99, and a batch size of 256. The model was trained for 50 epochs with early stopping. The MSE on set N was 0.10 and a threshold of −23 was set on the log PD values of prediction errors from set T.

Evaluation: The results of anomaly detection on set T are shown in figure 4.6. With a threshold of −23 the model detects all three anomalies. These results are similar to the results, which finds the three most unusual sequences in this data. Furthermore, the rankings of the discords match the order of log PD values with a lower value indicating a more unusual sequence.

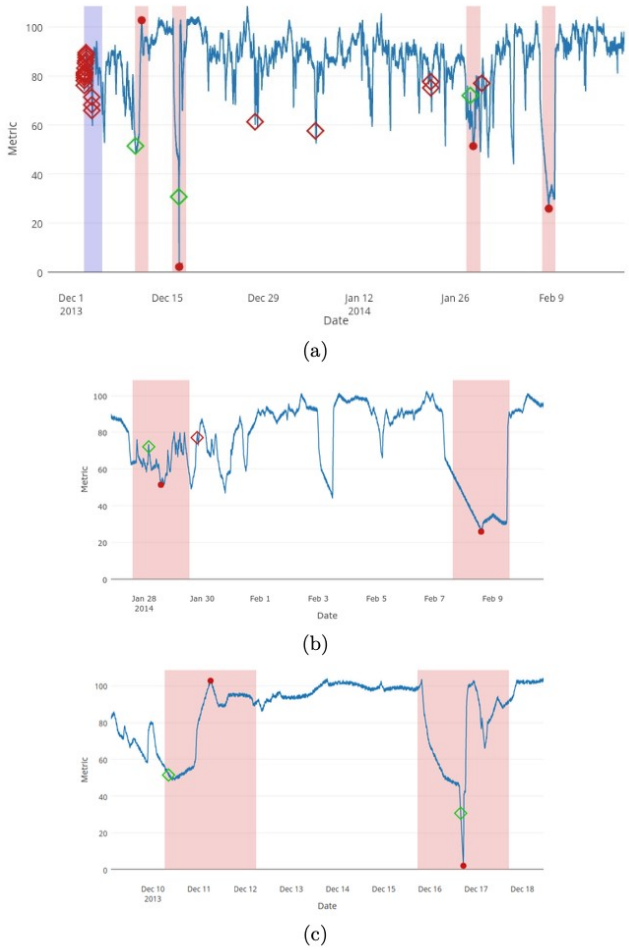

Figure 3. Results of HTM on Machine Temperature Data. (a) shows the results on the entire dataset. (b) shows the results corresponding to the training set. (c) shows the results corresponding to the test set. The X-axis shows time and the Y-axis shows the temperature. True anomalies are denoted by red markers. Green/red diamonds represent true/false positives. The pink shaded portions are anomaly windows. The purple shaded area in (a) is the testing window. Plots generated from code available at NAB GitHub repository.

Model Details: The prediction model used was a stacked RNN consisting of two hidden recurrent layers with 60 and 30 LSTM units respectively, a dense output layer with 5 neurons, a lookback of 8, a lookahead of 5, and a dropout of 0.1. LSTM state was not maintained between batches. We trained the RNN using Adam optimizer with a learning rate of 0.1, a decay of 0.99, and a batch size of 256. The model was trained for 50 epochs with early stopping. The MSE on set N was 0.10 and a threshold of −23 was set on the log PD values of prediction errors from set T .

Evaluation: The results of anomaly detection on set T are shown in figure 4.6. With a threshold of −23 the model detects all three anomalies. These results are similar to the results in, which finds the three most unusual sequences in this data. Furthermore, the rankings of the discords match the order of log PD values with a lower value indicating a more unusual sequence.

C. Maintaining LSTM states –

An important factor in making LSTMs learn useful patterns from data is the treat- ment of LSTM state. We experimented with different ways of building LSTM state and their effect on the prediction and detection performance.

The ECG dataset has a repeating pattern, of roughly 370 time steps, though the actual length of the pattern varies slightly. We expected that maintaining state between batches would give better results, but results were similar irrespective of how we handled the LSTM state. We believe even though ECG dataset had a long pattern, the relevant temporal relations were present only in recent time steps.

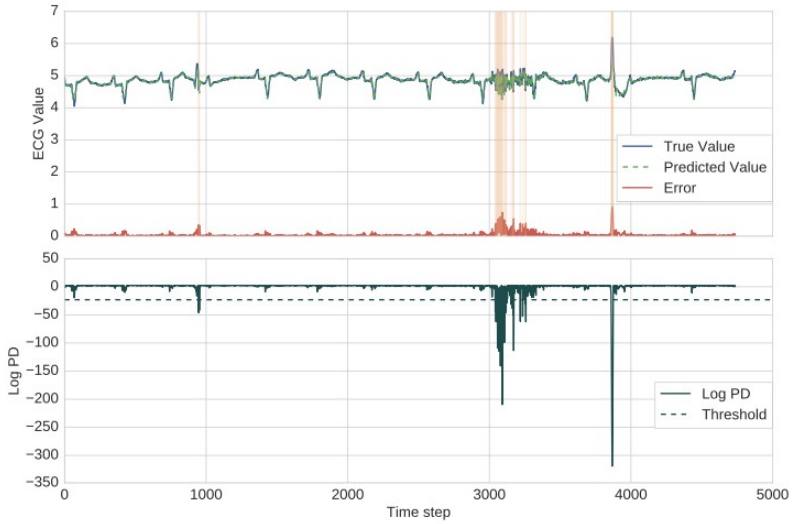

Figure 4. Results for ECG dataset. The top plot shows the predictions done on T and the corresponding prediction errors. The bottom plot shows the log PD values of the prediction errors and the threshold set at −23. The X-axis shows time steps and the Y-axis has the corresponding metric value. There are three anomalies. The anomalies detected by the algorithms are represented by the shaded regions.

D. For Feedforward NN's with fixed time Windows –

For ECG dataset the NN used had two hidden layers with 64 and 32 neutrons respectively, with sigmoid activation, and no dropout. The model was trained with Adam optimizer using a learning rate of .001, batch size of 1024, and early stopping. All other parameters were the same as earlier. The model was trained for 80 epochs and gave a MSE of .07 on set N. A threshold of −25 on log PD values of the errors on set T detected all anomalies. The results on ECG dataset are shown in figure 4.9 and are similar to the ones for LSTM model.

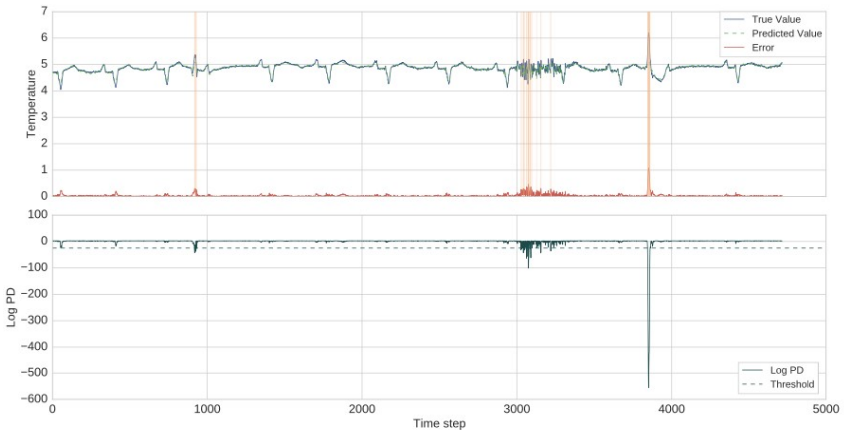

Figure 5. Results using feed-forward NN on ECG data. The results are on set T and a threshold of −25 detects all 3 anomalies. The X-axis shows the time step and the Y-axis has the corresponding metric value.

V. CONCLUSION

We developed an anomaly detection method for temporal data using LSTM networks. We started by studying LSTMs to explain their complex architecture, their ability to learn long-range dependencies, and different ways of maintaining LSTM state. Next, we developed an anomaly detection method based on a summary prediction model. We briefly touched upon the various issues prevalent in anomaly detection research. To circumvent these problems we selected three real-world datasets that have been used in previous research and contain different types of anomalies. Finally, we conducted experiments on these datasets. Based on our results we conclude that LSTMs are effective time series modelers and anomaly detectors. But we advise trying feed-forward NNs first as they are sufficient in many cases. LSTMs come with added complexity and will provide an advantage only when the data has long-range dependencies and multiple patterns.

REFERENCES

[1] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly Detection: A Survey,” *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009. doi: 10.1145/1541880.1541882. [Online]. Available: http://doi.acm.org/10.1145/ 1541880.1541882

[2] G. O. Campos, A. Zimek, J. Sander, R. J. G. B. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle, “On the Evaluation of Unsupervised Outlier Detection: Measures, Datasets, and an Empirical Study,” *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 891–927, Jul 2016. doi: 10.1007/s10618-015-0444-8. [Online]. Available: http://dx.doi.org/10.1007/s10618-015-0444-8

[3] Q. Yang and X. Wu, “10 challenging problems in data mining research,” *Inter- national Journal of Information Technology &; Decision Making*, vol. 05, no. 04, pp. 597–604, 2006. doi: 10.1142/S0219622006002258. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0219622006002258

[4] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han, “Outlier Detection for Tempo- ral Data: A Survey,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2250–2267, Sept 2014. doi: 10.1109/TKDE.2013.184

[5] Y. LeCun, Y. Bengio, and G. Hinton, “Deep Learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. doi: 10.1038/nature14539. [Online]. Available: http://dx.doi.org/10.1038/nature14539

[6] A. Håkansson, “Portal of Research Methods and Methodologies for Research Projects and Degree Projects,” *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering FECS’13*, pp. 67–73, 2013. [Online]. Available: http://urn.kb.se/resolve?urn= urn:nbn:se:kth:diva-136960

[7] “Introduction to Artificial Neural Networks - Part 1.” [Online]. Available: http://www.theprojectspot.com/tutorial-post/ introduction-to-artificial-neural-networks-part-1/7

[8] D. Williams and G. Hinton, “Learning Representations by Back-Propagating Errors,” *Nature*, vol. 323, no. 6088, pp. 533–538, 1986.48

[9] S. Ruder, “An Overview of Gradient Descent Optimization Algorithms,” *arXiv preprint arXiv:1609.04747*, 2016.

[10] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[11] P. Domingos, “A Few Useful Things to Know about Machine Learning,” *Com- munications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012.

[12] R. J. Frank, N. Davey, and S. P. Hunt, “Time Series Prediction and Neural Networks,” *Journal of Intelligent & Robotic Systems*, vol. 31, no. 1, pp. 91–103, 2001.

[13] A.-r. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic Modeling using Deep Belief Networks,” *IEEE Transactions on Audio, Speech, and Language Pro- cessing*, vol. 20, no. 1, pp. 14–22, 2012.

[14] L. Rabiner and B. Juang, “An Introduction to Hidden Markov Mod- els,” *IEEE ASSP Magazine*, vol. 3, no. 1, pp. 4–16, Jan 1986. doi: 10.1109/MASSP.1986.1165342
[15] P. J. Werbos, “Backpropagation Through Time: What It Does and How to Do It,” *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.
[16] Y. Bengio, P. Simard, and P. Frasconi, “Learning Long-Term Dependencies with Gradient Descent is Difficult,” *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.
[17] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, “Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies,” 2001. [Online]. Available: http://www.bioinf.jku.at/publications/older/ch7.pdf
[18] R. Pascanu, T. Mikolov, and Y. Bengio, “On the Difficulty of Training Recur- rent Neural Networks.” *ICML (3)*, vol. 28, pp. 1310–1318, 2013.
[19] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural com- putation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[20] F. A. Gers, J. Schmidhuber, and F. Cummins, “Learning to forget: Continual Prediction with LSTM,” *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
[21] K. Greff, R. K. Srivastava, J. KoutnÃk, B. R. Steunebrink, and J. Schmid- huber, “LSTM: A Search Space Odyssey,” *IEEE Transactions on Neu- ral Networks and Learning Systems*, vol. PP, no. 99, pp. 1–11, 2017. doi: 10.1109/TNNLS.2016.2582924
[22] M. Hermans and B. Schrauwen, “Training and Analysing Deep Recurrent Neu- ral Networks,” in *Advances in Neural Information Processing Systems 26*. Cur- ran Associates, Inc., 2013, pp. 190–198. [Online]. Available: http://papers.nips. cc/paper/5166-training-and-analysing-deep-recurrent-neural-networks.pdf
[23] A. Graves, A. r. Mohamed, and G. Hinton, “Speech Recognition with Deep Recurrent Neural Networks,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013. doi: 10.1109/ICASSP.2013.6638947. ISSN 1520-6149 pp. 6645–6649.
[24] H. Sak, A. Senior, and F. Beaufays, “Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014. [Online]. Available: http://193.6.4.39/~czap/letoltes/IS14/ IS2014/PDF/AUTHOR/IS141304.PDF
[25] S. Basu and M. Meckesheimer, “Automatic Outlier Detection for Time Series: An Application to Sensor Data,” *Knowledge and Information Systems*, vol. 11, no. 2, pp. 137–154, 2007. doi: 10.1007/s10115-006-0026-6. [Online]. Available: http://dx.doi.org/10.1007/s10115-006-0026-6
[26] D. J. Hill and B. S. Minsker, “Anomaly Detection in Streaming Environmental Sensor Data: A Data-Driven Modeling Approach,” *Environmental Modelling & Software*, vol. 25, no. 9, pp. 1014–1022, 2010.
[27] J. Ma and S. Perkins, “Online Novelty Detection on Temporal Sequences,” in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’03. New York, NY, USA: ACM, 2003. doi: 10.1145/956750.956828. ISBN 1-58113-737-0 pp. 613–618. [Online]. Available: http://doi.acm.org/10.1145/956750.956828
[28] R. S. Tsay, D. Peña, and A. E. Pankratz, “Outliers in Multivariate Time Series,” *Biometrika*, vol. 87, no. 4, pp. 789–804, 2000.
[29] Y. Yao, A. Sharma, L. Golubchik, and R. Govindan, “Online Anomaly De- tection for Sensor Systems: A Simple and Efficient Approach,” *Performance Evaluation*, vol. 67, no. 11, pp. 1059–1075, 2010.
[30] H. N. Akouemo and R. J. Povinelli, “Probabilistic Anomaly Detection in Natu- ral Gas Time Series Data,” *International Journal of Forecasting*, vol. 32, no. 3, pp. 948–956, 2016.
[31] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, “Long Short Term Memory Networks for Anomaly Detection in Time Series,” in *Proceedings*. Presses universitaires de Louvain, 2015, p. 89. [Online]. Available: https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf
[32] S. Chauhan and L. Vig, “Anomaly Detection in ECG Time Signals via Deep Long Short-Term Memory Networks,” in *2015 IEEE International Con- ference on Data Science and Advanced Analytics (DSAA)*, Oct 2015. doi: 10.1109/DSAA.2015.7344872 pp. 1–7.
[33] L. Bontemps, V. L. Cao, J. McDermott, and N.-A. Le-Khac, *Collective Anomaly Detection Based on Long Short-Term Memory Recurrent Neural Net-works*. Springer International Publishing, 2016, pp. 141–152. ISBN 978-3-319- 48057-2. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-48057-2_9
[34] M. Cheng, Q. Xu, J. Lv, W. Liu, Q. Li, and J. Wang, “MS-LSTM: A Multi-Scale LSTM Model for BGP Anomaly Detection,” in *Network Proto- cols (ICNP), 2016 IEEE 24th International Conference on*. IEEE, 2016, pp. 1–6.
[35] E. Marchi, F. Vesperini, F. Weninger, F. Eyben, S. Squartini, and B. Schuller, “Non-Linear Prediction with LSTM Recurrent Neural Networks for Acoustic Novelty Detection,” in *2015 International Joint Conference on Neural Net- works (IJCNN)*, July 2015. doi: 10.1109/IJCNN.2015.7280757. ISSN 2161-4393 pp. 1–7.
[36] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, “LSTM-Based Encoder-Decoder for Multi-sensor Anomaly Detection,” *CoRR*, vol. abs/1607.00148, 2016. [Online]. Available: http: //arxiv.org/abs/1607.00148
[37] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian Op- timization of Machine Learning Algorithms,” in *Advances in Neu- ral Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 2951–2959. [Online]. Available: http://papers.nips.cc/paper/ 4522-practical-bayesian-optimization-of-machine-learning-algorithms.pdf
[38] A. F. Emmott, S. Das, T. Dietterich, A. Fern, and W.-K. Wong, “Systematic Construction of Anomaly Detection Benchmarks from Real Data,” in *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, ser. ODD ’13. New York, NY, USA: ACM, 2013. doi: 10.1145/2500853.2500858. ISBN 978-1-4503-2335-2 pp. 16–21. [Online]. Available: http://doi.acm.org/10.1145/2500853.2500858.
[39] A. Lavin and S. Ahmad, “Evaluating Real-Time Anomaly Detection Algo- rithms - The Numenta Anomaly Benchmark,” in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, Dec 2015. doi: 10.1109/ICMLA.2015.141 pp. 38–44.
[40] E. Keogh, J. Lin, and A. Fu, “HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence,” in *Fifth IEEE International Conference on DataMining (ICDM’05), Nov 2005. doi: 10.1109/ICDM.2005.79. ISSN 1550-4786 pp. 8–.*
[41] M. Jones, D. Nikovski, M. Imamura, and T. Hirata, “Anomaly Detection in Real-Valued Multidimensional Time Series,” in *International Conference on Bigdata/Socialcom/Cybersecurity. Stanford University, ASE*, 2014. [Online]. Available: http://www.merl.com/publications/docs/TR2014-042.pdf.
[42] M. C. Chuah and F. Fu, *ECG Anomaly Detection via Time Series Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 123–135. ISBN 978-3-540-74767-3. [Online]. Available: http://dx.doi.org/10.1007/ 978-3-540-74767-3_14.
[43] J. Hawkins and D. George, “Hierarchical Temporal Memory: Concepts, Theory and Terminology,” Technical report, Numenta, Tech. Rep., 2006. [Online]. Available: http://www-edlab.cs.umass.edu/cs691jj/hawkins-and-george-2006. pdf.

[44] F. A. Gers, D. Eck, and J. Schmidhuber, *Applying LSTM to Time Series Predictable through Time-Window Approaches*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 669–676. ISBN 978-3-540-44668-2. [Online]. Available: http://dx.doi.org/10.1007/3-540-44668-0_93.

[45] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An Empirical Exploration of Recurrent Network Architectures," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 2342–2350.

[46] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," *arXiv preprint arXiv:1406.1078*, 2014. [Online]. Available: https://arxiv.org/abs/1406.1078.

[47] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," *arXiv preprint arXiv:1412.3555*, 2014. [Online]. Available: https://arxiv.org/abs/1412.3555.

[48] A. Singh, "Anomaly Detection for Temporal Data using Long Short-Term Memory (LSTM)," Dissertation, 2017.