System Anomaly Prediction Using Memory Networks

Anupam Godse, Cherukeshi Machan and Shantanu Chandorkar
{angodse, cmachan, schando}@ncsu.edu

***Abstract*— Distributed systems are becoming increasingly complex and important. As our reliability on such systems grow, the dimension of availability and usability that these systems offer become increasingly critical. These are influenced intensively by various systems anomalies which could be affected by a large set of parameters such as workload and node availability. Predicting such anomalies could help improve availability of such systems provided there is sufficient lead time. In our study, we aim to use a concept drift aware model that could predict such anomalies with sufficient lead time. We aim to use memory networks which could understand long term temporal dependencies in workloads to predict anomalies in a distributed infrastructure. To achieve this, we deploy LSTM based Encoder-Decoder, and GRU Auto Encoders with online learning.**

1. Introduction

Enterprise data centers are commonly complex in structure and their resources are shared, their performance and availability can be easily compromised when anomalies occur. Popular web services for instance, Netflix, Uber, Airbnb and Reddit are deployed on cloud providers such as Amazon Web Services (AWS) and depend on its availability to sustain their business and services. Various anomaly detection systems are developed to detect such anomalies. Several of them are augmented with diagnostic annotations that help automate system state recovery. The major shortcoming of these anomaly detection systems is that they only detect anomalies after they occur. They cannot be used to prevent anomalies from occurring. Anomaly prediction systems overcome this shortcoming by predicting the occurrence of anomalies a given time, called lead time, ahead.

Several recent studies, as discussed later, try to improve the prediction accuracy while increasing lead time and reducing false alarm rate. Such systems can help mitigate the effects of anomalies or even prevent them altogether. However, not all of these systems account for concept drift or are robust enough to adapt to various workloads while providing good lead time. In our study, we aim to predict anomalies by using a special type of memory network which was developed to be adaptable or robust to concept drift that could be used online for training and prediction. We aim to use memory networks which could understand long term temporal dependencies in workloads to predict anomalies in a distributed infrastructure. To achieve this, we deploy LSTM based Encoder-Decoder [13], and GRU Auto Encoders with online learning as described in [10]. We have tested our model on the NCSUs virtual computing lab (VCL) that operates in a similar way as Amazon EC2. We conducted extensive experiments using a range of real distributed systems: 1) RUBiS[16], an online auction benchmark, 2) A commercial stream processing system using Kafka[17].

The remaining of this paper is organized as following: Section 2 gives an overview of work on anomaly prediction and different models used for anomaly detection or prediction. In Section 3 we conduct anomaly prediction experiments on two system monitoring datasets and demonstrate the experimental evaluation results. Section 4 we presented the related work, and Finally in section 5, we conclude this paper.

2. Design and Algorithms

Two different models are implemented for solving the problem. Recurrent Neural Networks (RNNs) based on gated units such as Long Short Term Memory(LSTM) and Gated Recurrent Units (GRUs) have been shown to be very effective for sequential data yielding applications in speech, text, video as well as real-valued time series such as sensor data from machines [15].

GRU Auto Encoders: GRUs are Gated Recurrent Units, these are simpler versions of LSTM cells. A GRU consists of two gates, update gate and reset gate, which control the flow of data by manipulating hidden state of the unit. The reset gate is used to predict a value for hidden state at time t using the hidden state at time $t-1$ and the state of the units in the previous layer. The update gate decides what fractions of previous hidden state and proposed hidden state to use to obtain the updated hidden state. For a multi-layer RNN, the hidden state z_t^i at time t for hidden layer i is obtained from z_{t-1}^i and z_t^{i-1}

This is a variant of LSTM Auto Encoders where the LSTM cells are replaced with GRU cells. The model is implemented with online learning as described in [10].

LSTM Encoder Decoder: The Encoder-Decoder LSTM is an RNN designed to address sequence-to-sequence problems, sometimes called seq2seq. An LSTM cell has and three gates that manipulate of the flow of information inside the cell. These gates are input gate, output gate and a forget gate. The input gate controls the extent to which a new value flows into the cell, the forget gate controls the extent to which a value remains in the cell and the output gate controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit.

We pose the prediction problem as a translation problem to use this model. This architecture is comprised of two models, one for reading the input sequence and encoding it into a fixed-length vector, or a latent space, and a second for decoding the latent space representation and outputting the predicted sequence.

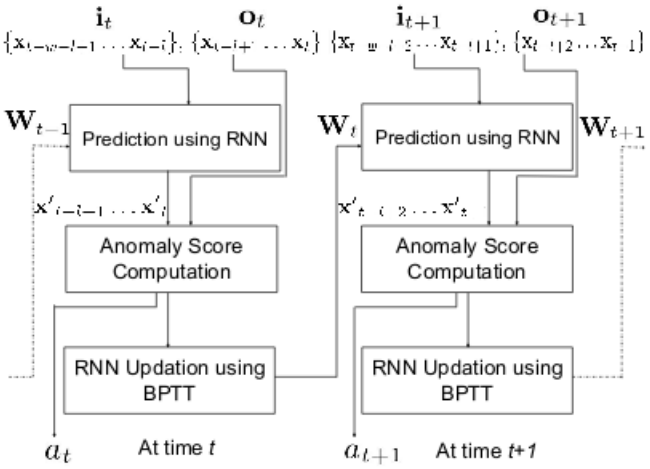

Fig. 1: Steps in Online RNN Approach

i_t and o_t are input and output at time t

W is the weight matrix

x_i is the time series

a_t is the anomaly score for time t

BPTT is back-propagation through time

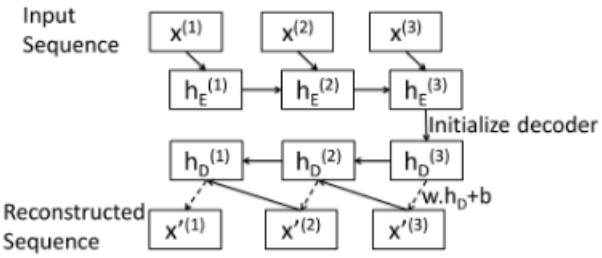

Fig. 2: LSTM Encoder Decoder Approach

x^i is the time series

h_e^t is the final state of encoder

h_D^t is the initial state of decoder obtained from h_e^i and x^i

We train an LSTM encoder-decoder to reconstruct instances of normal time-series. The LSTM encoder learns a fixed length vector representation of the input time-seriesand the LSTM decoder uses this representation to reconstruct the time-series using the current hidden state and the value predicted at the previous time-step [12].

3. Experiment Evaluation

We conducted experiments to evaluate our models by applying them on real applications. We describe experimental setup and report results in this section. A. Experimental Setup We evaluate our prediction models based on the anomaly data collected from two benchmark systems: the RUBiS[16] multi-tier online auction web application (PHP version), Real

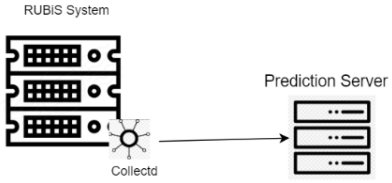

Fig. 3: RUBiS online auction benchmark

time data stream processing system.

Our experiments were conducted on the Virtual Computing Lab (VCL) infrastructure which operates in a similar way as Amazon EC2. Each VCL host has Ubuntu 18.04 LTS Base image. Collectd daemon is used to collect resource usage information (e.g., CPU usage, memory allocation). The sampling interval is 1 second.

We have chosen two benchmark systems to evaluate our prediction models in order to be sure that our system can be used in the real world. Faults were injected at different times while the systems were under dynamic workload. Fault injections last between 3-10 seconds. We now describe all the systems and fault injections in detail as follows.

RUBiS[16] online auction benchmark: The RUBiS[16] is the threetier on-line auction benchmark system (PHP version) with one web server, two application servers, and one database server.Fig 1 shows the system architecture

In order to evaluate our system under workloads with realistic time variations, we used a client workload generator that emulates the workload. The client workload generator also records the response time of the HTTP requests it made. Two types of faults, Memleak, CPUHog were injected in the RUBiS data set.Memleak and CPUHog are injected using the stress tool which imposes a configurable amount of CPU, memory stress on the system .

Real Time Streaming Application: We used Kafka[17] for the real time data stream processing system. The System consist of a streaming server that accepts the streams of data from clients and store them in kafka topics. Multi-node multi-server kafka cluster used as the streaming platform.). In order to evaluate our system under dynamic workloads with realistic time variations, We used a client workload generator that register itself as a producer to the system. Fig 2 shows the system architecture.

For Streaming System, we injected MemLeak and CPUHog faults. Memleak and CPUHog are injected using the stress tool on each host which imposes a configurable amount of CPU, memory stress on the system .

For Collection of system metrics collectd is used. Collectd sends metrics from all the servers to a prediction server. Fig 5 and Fig 6 shows the system metrics collected from the benchmark systems.

Prediction Service: The metrics, are produced by each machine and sent to a dedicated Kafka Broker. Our prediction service acts as a Kafka Consumer on an independent machine and streams these metrics to the prediction models

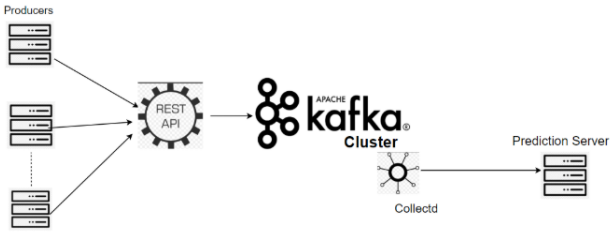

Fig. 4: Real Time Streaming Application

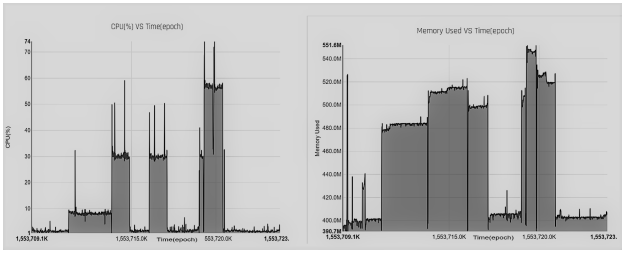

Fig. 5: RUBiS online auction benchmark

Window Size	Updation Cost
2	0.021 secs
5	0.032 secs
7	0.037 secs
10	0.048 secs
12	0.053 secs
15	0.063 secs
20	0.080 secs

TABLE I: CPU updation cost for GRU Auto Encoders

Window Size	Updation Cost
2	0.021 secs
5	0.031 secs
7	0.038 secs
10	0.049 secs
12	0.057 secs
15	0.067 secs
20	0.081 secs

TABLE II: Memory updation cost for GRU Auto Encoders

described in previous section. The messages streamed include information such as the value of metric, its instance and the host. This information is used to detect the source of anomaly and instrument the model responsible for it.

The evaluation was done by training the models on normal workload for over 2 days (**44.1 hrs**), sampled at an interval of 1 sec each. To evaluate the model performance, anomalies were introduced using fault injections as described earlier. Our models, classify each sample of each type (active cpu, used memory) for all hosts as anomaly or benign. We mark the injection time-stamp and the entire duration of injection as anomaly. This experiment was conducted using bash scripts at random intervals for a little over 2 days (**44.34 hrs**). The labels and predictions were collected and evaluated on various metrics tabulated as follows.

4. Related Work

Online anomaly prediction in real time environment aims to detect anomalies in advance using different supervised and unsupervised approaches. Previous work on these methods is described as follows.

Statistical approaches are lightweight and may seem suitable for online anomaly prediction, but these approaches detect spatial anomalies and lack to consider the temporal dependencies and assume that states are independent [2].

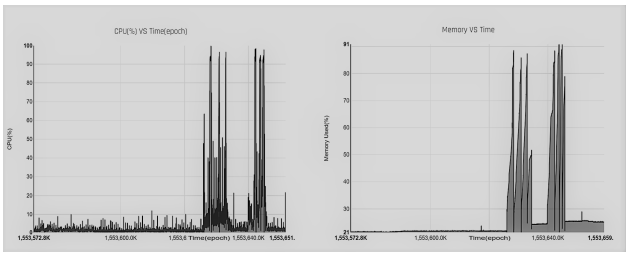

Fig. 6: Real Time Streaming Application

Advanced time-series such are ARIMA are capable of detecting temporal anomalies [8]. It is effective at detecting anomalies with periodic patterns. Recently relative entropies have been used to handle temporal anomalies [9].

Some methods are based on Markov models for instance X. Gu et al. proposed an algorithm for online anomaly prediction, which combines the Markov model and Bayesian classification process to predict the anomalies [6]. Y. Tan et al. also used the Markov model to predict the future state of the system using Tree augmented networks TANs [7]. However models based models on Bayesian networks need to be trained in parallel and picked heuristically as done by Ira Cohen and Jeffrey S. Chase [11] as they are prone to concept drift and are not good at representing high dimensional dependency.

The parameters of our model are updated in an incremental manner as new data arrives so that it adapts to changing trends in the data and becomes resistant to concept drifts [10]. Hierarchical Temporal Memory (HTM) models fall under this category and are computationally cheap but do not support multi-step predictions [2].

Recently, LSTM-based Encoder-Decoder scheme has been used for anomaly detection in multivariate time-series [12]. The encoder learns using feature vector of the input time-series and decoder uses this feature vector to reconstruct the time-series. The error in reconstruction is used to predict the anomaly. These approaches use RNNs for anomaly detection but do not adapt to concept drifts and only assume normal behaviours. Hence these models are not suitable for non-stationary time series prediction.

D. J. Dean et al. proposed an Unsupervised Behavior Learning (UBL) system for IaaS cloud computing infrastruc-

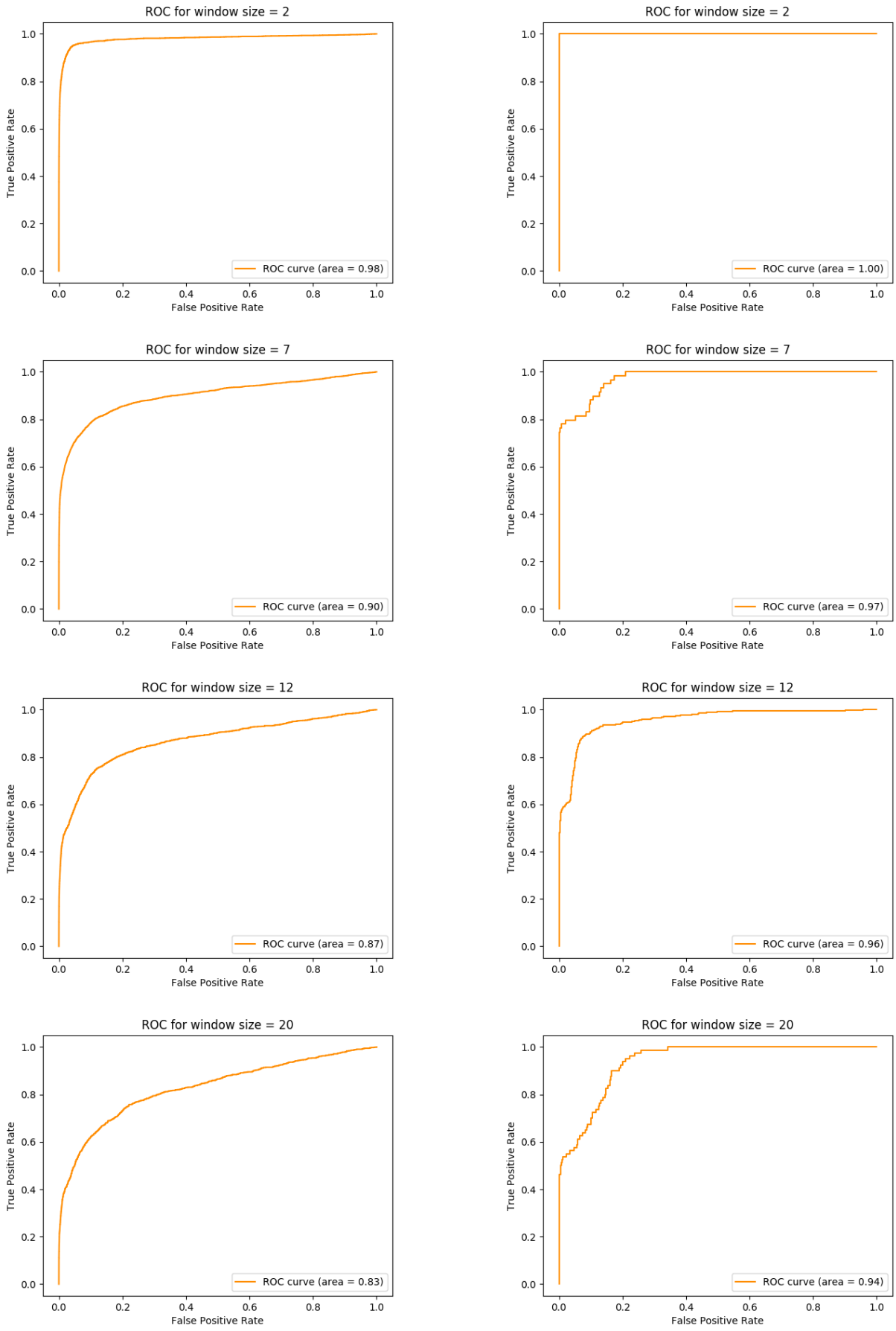

Fig. 7: ROC Curves for GRU Auto Encoder. CPU (left) and Memory (Right)

Remaining figures in repository

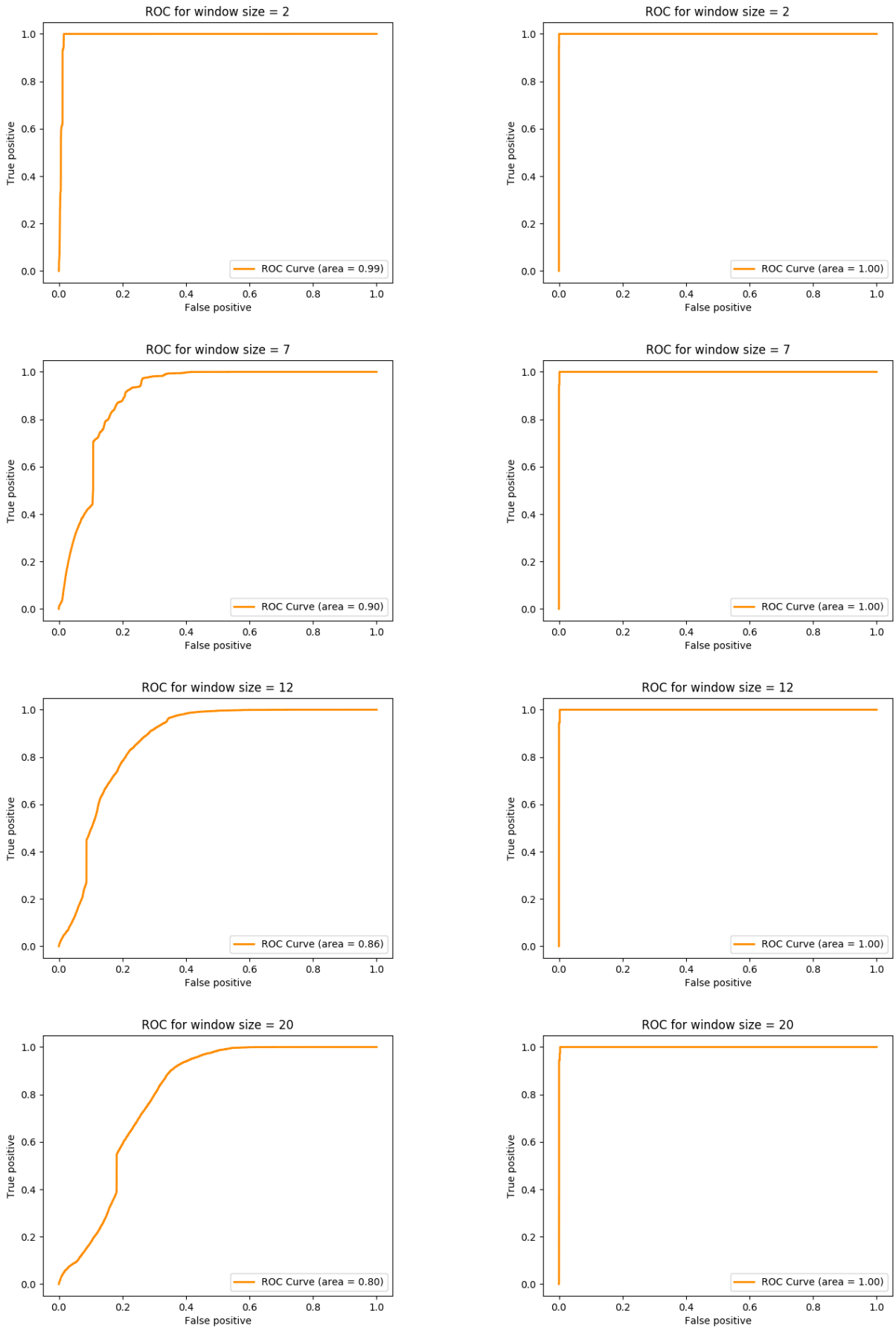

Fig. 8: ROC Curves for LSTM Encoder Decoder. CPU (left) and Memory (Right)

Remaining figures in repository

Window Size	Accuracy	Balanced Accuracy	Precision	Recall	F1	AUC
2	0.96	0.53	0.61	0.06	0.11	0.99
5	0.89	0.66	0.15	0.40	0.22	0.90
7	0.84	0.83	0.17	0.82	0.28	0.90
10	0.75	0.84	0.13	0.94	0.22	0.86
12	0.69	0.81	0.11	0.93	0.19	0.86
15	0.64	0.78	0.09	0.94	0.17	0.81
20	0.54	0.75	0.08	0.97	0.14	0.80

TABLE III: CPU Performance for LSTM Encoder Decoder

Window Size	Accuracy	Balanced Accuracy	Precision	Recall	F1	AUC
2	1.00	0.97	0.85	0.94	0.89	1.00
5	1.00	0.97	0.63	0.94	0.76	1.00
7	1.00	0.97	0.42	0.94	0.59	1.00
10	1.00	0.99	0.15	0.97	0.27	1.00
12	1.00	0.97	0.12	0.94	0.21	1.00
15	1.00	0.97	0.06	0.94	0.11	1.00
20	1.00	1.00	0.04	1.00	0.07	1.00

TABLE IV: Memory Performance for LSTM Encoder Decoder

Window Size	Accuracy	Balanced Accuracy	Precision	Recall	F1	AUC
2	0.98	0.83	0.74	0.67	0.70	0.97
5	0.98	0.88	0.64	0.78	0.70	0.90
7	0.97	0.80	0.56	0.61	0.58	0.88
10	0.96	0.73	0.49	0.49	0.49	0.85
12	0.96	0.64	0.47	0.29	0.36	0.86
15	0.94	0.88	0.38	0.81	0.52	0.79
20	0.96	0.57	0.42	0.14	0.21	0.81

TABLE V: CPU Performance for GRU Auto Encoders

Window Size	Accuracy	Balanced Accuracy	Precision	Recall	F1	AUC
2	1.00	1.00	0.24	1.00	0.38	1.00
5	1.00	1.00	0.39	1.00	0.56	0.99
7	1.00	0.96	0.44	0.93	0.60	0.97
10	1.00	1.00	0.15	1.00	0.27	0.98
12	0.99	1.00	0.09	1.00	0.16	0.96
15	1.00	1.00	0.18	1.00	0.31	0.99
20	1.00	0.87	0.26	0.75	0.39	0.94

TABLE VI: Memory Performance for GRU Auto Encoders

tures [3]. UBL leverages Self-Organizing Maps to capture emergent system behaviors and predict unknown anomalies. A. Waibel introduced the feed-forward neural network technique to recognize patterns in the relationship between perspectives [4, 5]. The model outputs the probability of a significant event occurring later. However these are unable to capture the dependency among various features or metrics. Recent work on anomaly prediction using RNNs has demonstrated to achieve high lead times and prediction accuracy with low false positive rate.

5. Conclusion

We found out that LSTM Encoder Decoder and GRU Auto encoders work well on the problem of anomaly prediction. We also experimentally evaluated and showed that online learning with RNNs [10] is feasible and can be done in production for models with similar complexity. Our experiments showed that it can be practical to use these models in real time while achieving good results and enabling online learning which is necessary to adapt to changing workloads.

REFERENCES

[1] Shaohan Huang, Carol Fung, Kui Wang, Polo Pei, Zhongzhi Luan and Depei Qian, "Using recurrent neural networks toward black-box system anomaly prediction," 2016 IEEE/ACM 24th International Symposium on Quality of Service (IWQoS), Beijing, 2016, pp. 1-10. doi: 10.1109/IWQoS.2016.7590435 .

[2] Ahmad, S., Lavin, A., Purdy, S., and Agha, Z. (2017). Unsupervised real-time anomaly detection for streaming data. Neurocomputing 262, 134147. doi:10.1016/j.neucom.2017.04.070

[3] D. J. Dean, H. Nguyen, and X. Gu, Ubl: unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems, in Proceedings of the 9th international conference on Autonomic computing, pp. 191200, ACM, 2012.

[4] A. Waibel, A. A. Alshehri, and S. Ezekiel, Multi-perspective anomaly prediction using neural networks, in Applied Imagery Pattern Recognition Workshop (AIPR): Sensing for Control and Augmentation, 2013 IEEE, pp. 16, IEEE, 2013.

[5] A. Waibel, A. A. Alshehri, and S. Ezekiel, Anomaly prediction in seismic signals using neural networks, in Applied Imagery Pattern Recognition Workshop (AIPR): Sensing for Control and Augmentation, 2013 IEEE, pp. 15, IEEE, 2013.

[6] X. Gu and H. Wang, Online anomaly prediction for robust cluster systems, in Data Engineering, 2009. ICDE09. IEEE 25th International Conference on, pp. 10001011, IEEE, 2009.

[7] Y. Tan and X. Gu, On predictability of system anomalies in real world, in Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS), 2010 IEEE International Symposium on, pp. 133140, IEEE, 2010.

[8] A.M. Bianco, M. Garca Ben, E.J. Martnez, V.J. Yohai, Outlier detection in regression models with ARIMA errors using robust estimates, J. Forecast. 20 (2001) 565579.

[9] C. Wang, K. Viswanathan, L. Choudur, V. Talwar, W. Satterfield, K. Schwan, Statistical techniques for online anomaly detection in data centers, in: Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management, 2011, pp. 385392, doi:10.1109/INM.2011.5990537.

[10] Sakti Saurav, Pankaj Malhotra, Vishnu TV, Narendhar Gugulothu, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Online anomaly detection with concept drift adaptation using recurrent neural networks. In Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, pages 7887. ACM, 2018.

[11] I. Cohen, M. Goldszmidt, T. Kelly, J. Symons, and J. S. Chase. Correlating instrumentation data to system states: a building block for automated diagnosis and control. In OSDI, 2004.

[12] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand,Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTMbased Encoder-Decoder for Multi-sensor Anomaly Detection. In Anomaly Detection Workshop at 33rd International Conference on Machine Learning. arXiv preprint arxiv:1607.00148 (2016).

[13] Narendhar Gugulothu, Pankaj Malhotra, Lovekesh Vig, and Gautam Shroff. 2018. Sparse Neural Networks for Anomaly Detection in High-Dimensional Time Series. Presented at International Workshop on AI for Internet of Things, IJCAI 2018, Stockholm, Sweden.

[14] Alex Pucher. Web. (www.alexpucher.com/blog/2015/06/29/cloud-traces-and-production-workloads-for-your-research/)

[15] Narendhar Gugulothu, Vishnu TV, Pankaj Malhotra, LovekeshVig, Puneet Agarwal, and Gautam Shroff. 2017. Predicting Re-maining Useful Life using Time Series Embeddings based onRecurrent Neural Networks.2nd ACM SIGKDD Workshop onML for PHM. arXiv preprint arXiv:1709.01073(2017).

[16] Rubis: Rice university bidding system. http://rubis.ow2.org,

[17] KAFKA-https://kafka.apache.org/documentation/"