COMP421(2020T2) - FINAL PROJECT REPORT

Au Tsz Kin

Victoria University Wellington

1. INTRODUCTION

In this report, we explore the unsupervised anomaly detection technique using Long short-term memory(LSTM) in temporal data, which LSTM is a variant of recurrent neural networks (RNNs). Akash Singh's master thesis [1] is selected to best suit the subject, there were discovery and improvement made around his LSTM RNN model.

The original code can be found in [1]. And the modified code is stored in my GitHub repository and can be downloaded and viewed publically: [`https://github.com/AUTSZKIN/COMP421-Final-Project`]

1.1. Anomaly Detection

Anomaly detection is a significant problem that has been studied within a wide variety of research areas and application domains. It refers to the problem of finding patterns or instances in data that do not match to expected behaviour. These deviated patterns or instances can be described as anomalies, outliers, exceptions, aberrations, surprises, peculiarities, discordant observations, or contaminants depending on domain and context [2]. Anomaly detection is useful and significant as the detected anomalies in data often translate to significant, actionable, or even critical information in various application domains. For example, anomaly detection techniques can be used in life-critical systems to detect faults, invasion detection for cyber-security, fraud detection for credit cards, insurance, or other finance-related areas.

1.1.1. Challenges of anomaly detection

The definition of anomalies is different across application domains. In the real-world, defining a region of data that include every variation of normal behaviour is not easy; critical anomalies that lie on the edge of the border can be considered normal, thus cause false-positive errors or vice versa.

Anomaly detection is usually considered an unsupervised problem, because anomalies are rare and often not seen before; when new samples arise or system update required, a new type of anomaly might come along, therefore it is impossible to define every variation of an anomaly in advance. Annotating anomalies is often a time-consuming process and domain experts with sufficient knowledge are required to label anomalies. We are performing anomaly detection on time series data in this report, there will be more challenges involved, QIANG YANG et al. [3] stated that time-series data remains it own problem, a large variety of time series used for predictions are contaminated by noise, that makes a prediction on short-term and long-term more difficult.

1.2. Recurrent Neural Network (RNN)

Recurrent Neural Network(RNN) is a class of artificial neural networks that allow previous outputs to be used as current inputs while having hidden states; therefore having the ability to learn long-term temporal patterns. RNNs are unlike Fully-Connected Networks or Convolution Networks, which lack "memories" when processing sequences or time series of data (e.g. climate data, stock market, medical data). In reality, when we understand the meaning of a sentence, each word is not independent, and we also have its context in our minds. Therefore the basic concept of RNN is to maintain some intermediate state information to help understand the context. RNN models are mostly used in the fields of natural language processing and speech recognition.

Fig.**??** shows a vanilla RNN, hidden units grouped with state s at time t, each neuron receive inputs from previous neurons at time step $t-1$, and passing the output to the next neuron at $t+1$, the state vector is preserved during its forward computation.

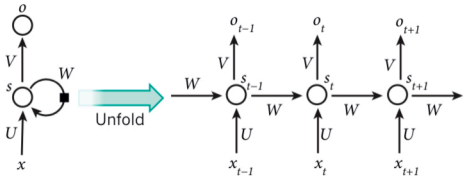


Fig. 1. A vanilla RNN, image used from [4]

1.2.1. Limitation

Training RNNs has proved to be difficult because the back-propagated gradients either grow or shrink at each time step, so when process sequences that are very long, RNN is prone to problems such as gradient exploding and gradient vanishing[4].

1.3. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a variant of RNN, it was introduced in [5], it has been proved to be a better successor of RNN in learning long-range temporal data, it mainly solved the vanishing gradients problem during long sequence training in RNN. Simply put, compared to vanilla RNNs, LSTM can perform better in learning long-term dependencies. Other than allowing previous outputs to be used as current inputs while having hidden states in RNN. LSTM adds a method that can transmit information with multiple timesteps apart. Think of a conveyor belt/carry track running together when you process the sequences. The information of each node in sequence can be put on the conveyor belt, or taken off from the conveyor belt, of course, you can also update the information on the conveyor belt. In this way, the information long ago is preserved and the loss of information is prevented.

To be exact, the main units of LSTM are introduced in [5] and [6] and the illustrative diagram of a LSTM unit is shown in Fig.2 :

- A central unit called Constant error carousel (CEC), which allows for constant error flow through special self connected units.
- Three gates/multiplicative units that control the flow in CEC;
  - Input gate: prevents CEC from receive irrelevant inputs;
  - Forget gate: the mechanism is introduced in [6] that allow LSTM to "forget" or abandon old and no longer relevant content in memory.
  - Output gate prevents other units receive disturbing information from CEE;

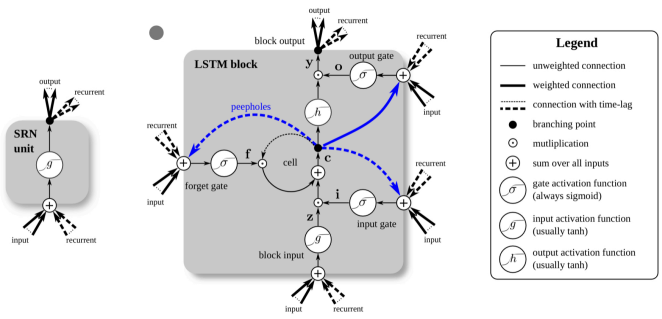


Fig. 2. LSTM with forget gate compare with a simple recurrent network (SRN), image used from [7]

1.4. Selected Paper

The suitable paper selected to match the subject is Akash Singh's master thesis [1]. He is an ML Engineer/Data Scientist; Master in Data Science from KTH Information and Communication Technology. He used an unsupervised approach to accomplish Long short-term memory (LSTM) technique for anomaly detection in time-series data since the data are unlabelled.

An RNN model with LSTM units was trained to learn the normal temporal patterns and predict values in future time steps. The anomaly scores are given by modelling prediction error result; therefore determine whether an instance is an anomaly or not. Singh also explored different ways to maintain state in LSTM and the effect of the different number of time steps used on prediction and detection performance.

Three real-world datasets were used in his experiments, the results show that maintaining the LSTM state is critical for getting a proper result, LSTM RNNs are suitable for general purpose time series modelling and anomaly detection.

1.5. Contribution

TBA

2. METHODS, NETWORK ARCHITECTURE, DATASETS AND FILE DIRECTORIES

Singh's experiments and model [1] was written and implemented using Python programming language with Keras; His algorithm contains two parts. Part one is to train a prediction model by learning normal temporal patterns from the dataset, the model can predict future time series. Part two is the anomaly detection, anomaly scores are computed from the prediction errors.

2.1. Prediction Model

The term $lookback$ number and $lookahead$ number were used in the model as input the most recent p value and output q value respectively.

The network contains hidden recurrent layer/layers followed by an output layer. The number of hidden recurrent layers and units varies for different datasets. Two recurrent layers are fully connected with each other. To avoid overfitting $dropout$ is used between two recurrent layers. The output layer is a fully connected dense NN layer. The number of neuron nodes 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, linear activation is used in the output layer and mean-square error(MSE) as the loss function [1].

The main purpose of the prediction model is to predict multiple time steps ahead, that is, predicting multiple future values; With a $lookahead$ of q at time t the model predicts the next q future values i.e. t+1, t+2..., t+q. Consider a time series with a scale of 10 minutes. Predicting $lookahead$ value of 6 can give us the behaviour of the time series for the next 30

minutes. It is useful for a system to predict possible unusual behaviour, e.g. an extreme value, early alerts can be sent out.

2.2. Anomaly Detection and Data Pre-processing

The prediction errors are used as anomaly indicators, which is the difference between prediction made at time step t-1 and the input value received at current time step t. The prediction errors from training data are modelled using a Gaussian distribution; the 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: the lower the PD values the greater likelihood of the instance being an anomaly. A validation set contains both normal instances and anomalies are used to set a threshold on log PD values; it can separate anomalies from normal instances and produce as few false-positive errors as possible. Finally, a separate test set containing both normal instances and anomalies is used to evaluate the model.

In order to learn normal time series patterns and optimise prediction performance, only normal data without anomalies is used for training LSTM RNN model. For the different dataset, each is divided into four subsets: a training set, N, with only normal values; validation set, V_N, with only normal values; a second validation set, V_A, with normal values and anomalies; and a test set, T, having both normal values and anomalies. There are three main procedures of the LSTM RNN training algorithm:

- 1. Set N with only normal values is used for training prediction model, Bayesian optimization [8] to find the best values for network hyper-parameters. And V_N is used for early stopping to prevent model overfitting.
- 2. Gaussian distribution is used for modelling prediction errors on N. The trained prediction model is applied to V_A. The log PD of errors are calculated from V_A and used as anomaly scores. A threshold is set on the log PD values which separate the possible anomalies from normal values.
- 3. The prediction errors from the test set T is used for the set threshold, therefore it is used as an anomaly indicator to identify anomalies from the test set T.

2.3. Datasets

It is difficult to judge how well an anomaly detection algorithm would generalize to different type datasets and there is no real-world validity of the actual anomalies and algorithm performance if artificial datasets are used to train the model [1].

Therefore instead of using application-specific datasets or artificial datasets, three real-world data sets were used in Singh's experiments: Numentas MachineTemperature Dataset, Power Demand Dataset and ECG Dataset. All three datasets contain real-world anomalies annotated by domain experts and all been used in previous works on anomaly detection.

2.4. Code Directories

The full code of the project is under `/Code`. There are 5 main parts of the original code that I explored or conduct experiments.

- The configuration of model hyper-parameters is placed under `/Code/configuration/`
- The real-world datasets are stored in `Code/resources/data/`
- The LSTM model implementation is placed under `Code/models/`
- The LSTM predictor that train model and predict results is `Code/lstm_predictor.py`
- The data pre-processing and anomaly detection are executed by using Python Notebook, the notebooks are under `Code/notebooks/`

3. THE ISSUES OF CURRENT SYSTEM

Singh's model is worked quite well for both Power Demand Dataset and ECG Dataset (i.e. the electrical activity of the heart). According to Singh's thesis, the model detected all 5 anomalies in the Power Demand test set with the PD threshold of -24 but there is 1 small false-positive error. For ECG test set, the model with a threshold of -23 detects all three anomalies with no error. It is because they all have consistent and repetitive patterns and the LSTM model will be relatively easy to learn these pattern. Fig.3 shows example plots of a normal and an anomalous weekly cycle, the ECG dataset is produced the plot similar to Power Demand Dataset.

3.1. Problem of the machine temperature dataset

There are four anomalies in the machine temperature dataset distributed in the sets V_A and T (See Fig.4), the results of the anomaly detection on V_A and T are shown in Fig.5 and Fig.7 respectively; Orange shaded area in the graph denotes possible anomalies made by the LSTM algorithm.

3.1.1. Results

- For the validation set V_A: The PD threshold of -11 was necessary to detect the first anomaly in set V_A, but there are quite a few false positives errors, see Fig.5.
- For the test set T: On T the threshold of -11 detected the second anomaly but did not detect the first anomaly, and also incurred quite a lot false-positive errors before the first anomaly and around the second anomaly see Fig.7.

The detection result of the machine temperature dataset is not as good as the previous two. Since the machine temperature dataset does not seem to have any repeating pattern, so the model did not maintain LSTM state between batches. And indeed the performance of the Singh's model was found to be insensitive to how the state was maintained as he mentioned on the thesis. The result with too many false positives is making the detection system unusable.

3.1.2. Original model architecture

The LSTM RNN model Singh used for the machine temperature dataset prediction and anomaly detection had a $lookback$ of 24, a $lookahead$ of 12, two hidden recurrent layers with 80 and 20 LSTM units respectively, a dense output layer with 12 neurons, and a dropout of 0.1. With Adam optimizer using a learning rate of .05, a decay of 0.99, and a batch size of 1024. The training was done for 200 epochs with early stopping. This model gave an MSE of 0.09 on N. Using set V_A a threshold of -11 was set on the log PD values. The threshold was then evaluated using set T.

3.2. Effect of lookback value

The parameter $lookback$ is an important parameter that affects learning, it is the number of time steps for which the RNN is unfolded for back-propagation. Singh found that there is no general optimal value of $lookback$ in Section 4.4.1 [1]; $Lookback$ value depends on the characteristic of data and the temporal correlations present in it. Lookback values ranging from 8 to 200 have been used successfully for different tasks. Which Singh discovery on $lookback$ is verified by my experiment.

However, the selected $lookback$ value of 24 in Singh's original machine temperature dataset model does not seem to be the best choice; In my experiment, I have discovered that the $lookback$ of 50 produced a better prediction result than Singh's result which is discussed further in Section 4.

3.3. Trade-off between Prediction Accuracy and Anomaly Detection

Singh stated in Section 4.4.2 [1] that he experienced a trade-off between prediction performance and the results of anomaly detection. An LSTM RNN model optimised for minimising the prediction MSE was not the best for anomaly detection. The prediction errors were small throughout the set V_A and T making it difficult to find a good PD threshold to split anomalies from normal values without incurring too many false positives. This is the same result as described in Section 3.1.1.

But Singh's conclusion of the trade-off between Prediction Accuracy and Anomaly Detection does not seem valid; I have conducted a series of tests to improve the prediction accuracy while gaining a better anomaly detection result at the same time, which proved the invalidity of his argument. This is discussed in the next section.

4. EXPERIMENT RESULT AND CONTRIBUTION

On the machine temperature dataset, Singh's model did not perform as desired as the other two; As the machine temperature dataset contains inconsistent patterns, the model is difficult to maintain LSTM state between batches.

Also, Singh's arguments in Section 3.2 and Section 3.3 is rather weak and vague, or there is no definite proof to support it. So the content of the following experiment is inspired by Singh's arguments. The goal of the experiment is to improve the prediction accuracy while achieving a better result of anomaly detection on the machine temperature dataset, if not worse than the existing one.

4.1. The result from Singh's LSTM RNN model

The original architecture with 24 $lookback$ described in Section 3.1.2; In my experiment, Singh's model which gave us MSE of 0.089 on the Training set (N), this is very consistent with his results. The Validation (V_A) Loss is 1.30 and the Test Loss (T) is 0.347

4.2. Change the lookback value

There is one figure of training MSE on ECG dataset for different values of $lookback$, but none for the power demand dataset nor the machine temperature dataset; There is no detail about how different values of $lookback$ are been used in the original model. From this point of view, one possible way to improve the performance on the machine temperature dataset is to test different $lookback$ values. *The training and result log can be viewed under* `Code/logs/0-original-model-24lookbacks.log`

4.2.1. Experiment steps

Test with various hyper-parameter candidates to train model to get desired/better result. All the results are stored in `Code/logs/`

- Test different value of $lookback$
  - From 10-300, with interval of 10, test each $lookback$ value on the original model, but change the lookahead value to 1 to decrease the training time.
  - The training and result log can be viewed under `Code/logs/1-lookback-10to300-run.log`
- The best $lookback$ candidates are $[50, 60 and 150]$, each were tested with epochs of $[20, 40, 80, 160 and 320]$.

  - Similarly, 50 $lookback$ trained with 320 epochs gave us the best result, N MSE: 0.017, V_A Loss:0.40 and T Loss: 0.045. It is because the $lookahead$ value is set to 1, therefore the result will have much higher prediction accuracy than the one with the$lookahead$ set to 12.
  - The training and result log can be viewed under `Code/logs/2-find-best-lookbacks.log`

- Then I used $lookback$ value of 50 instead of 24 in the original model architecture described in Section 3.1.2:
  - The result showing: N MSE: 0.0867, V_A Loss:0.39 and T Loss: 0.22; Comparing the result from Singh's model in Section 3.1.1. The improvement of using 50 $lookback$ is noticeable.
  - The training and result log can be viewed under `Code/logs/3-final-modified-model`

4.3. Result after using 50 as the lookback value

Singh argued that increasing $lookback$ values improved prediction performance up to a limit and no improvement was gained by increasing the lookback values further, which show to be true by my experiment; 200 or 300 $lookback$ did not produce a better result than 50 $lookback$ did. However, according to my experiment result, Singh's choice of selecting 24 as the optimal $lookback$ for the machine temperature dataset is proved is not the best.

The Validation and Test sets results of the final model using 50 $lookback$ is shown in Fig.6 and Fig.8; My model is similar to Singh's. The two anomalies in V_A and the second anomaly in T are found, but the first anomaly in T is not detected. Having said that, compare my results with the original results in Fig.5 and Fig.7, we can see the prediction accuracy is observable higher: the predicted value (green dot line) is better fit the true value (blue line) than the original ones; And the error (red line) is lower than the original ones as well, especially at the spot of the second anomaly in the Validation sets.

Not only the prediction accuracy of my model out-rated the original one while found the same anomalies, but it also decreases the anomaly false positive errors. The Orange shaded area in the graph denotes possible anomalies made by the LSTM algorithm. Comparing my results (Fig.6 and Fig.8) with Singh's (Fig.5 and Fig.7), it is notable that the model after the modification has much lesser false-positive anomalies. This result makes Singh's assertion of the trade-off between prediction accuracy and anomaly detection to be false.

4.4. Other experiments

I also modified the amount of recurrent layer and LSTM units in it, the hyper-parameters, architecture, and final results can be viewed in `Code/logs/4-50lookback-2layers.log`, this model also gave us a better MSE and Loss result. Due to the time constraint, the anomaly detection test is not been executed, but this gave us the opportunity of the potential improvement of the LSTM RNN anomaly detection model.

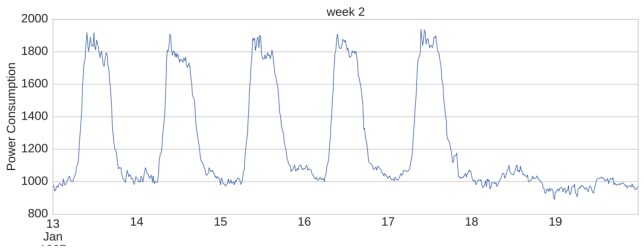
4.5. Code modified

The main code I have modified is under `Code/lstm_predictor-modified.py` and `Code/configuration/config_modified.py`

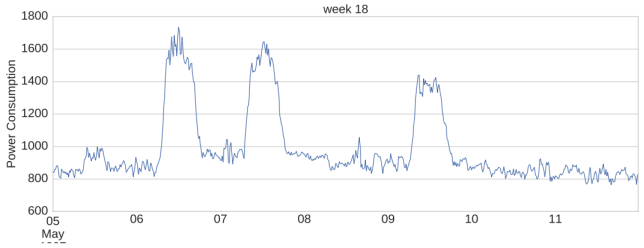
5. CONCLUSION

According to the principle of Occam's razor, I did not do too many changes to the original model of Singh, but from the basic parameters modify the test to obtain the desired result.

Although my model is similar to Singh's but the accuracy of the prediction model is significantly improved, thereby reducing the error rate of false positives. This proves that Singh's model has a lot of room for improvement. If time permits, I believe that the model can be further optimised and can be used in the real-life systems. My experiments and modification have also proved that unlike ordinary Feed-forward neural networks or Convolutional neural networks, for different types of data, there is no unified or generalised LSTM model structure for anomaly detection; depending on different data types and characteristic, the appropriate number of hidden recurrent layers, neurons and suitable hyper-parameters should be tested and used to gain the possible best result.



(a) A weekly cycle with high demand on weekends and low demand on weekdays.



(b) A week with anomalies as Monday and Thursday have low demand.

Fig. 3. Power demand normal and anomalous patterns [1]

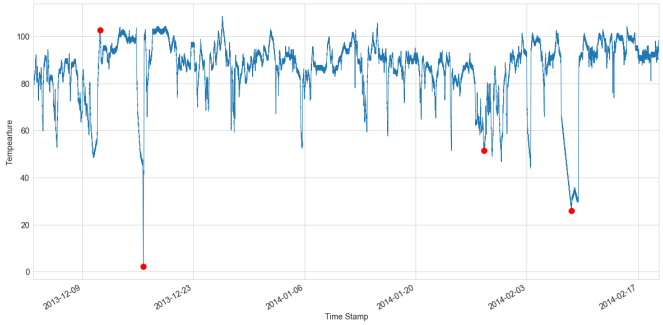


Fig. 4. Normal pattern/values and actual anomalies in the Numentas Machine Temperature Dataset; Four known anomalies indicated by red markers. The X-axis shows time steps, and the Y-axis measures temperature

Fig. 5. Original validation set results on machine temperature dataset. The X-axis shows time step. Orange shaded area in the graph denotes possible anomalies made by the LSTM algorithm

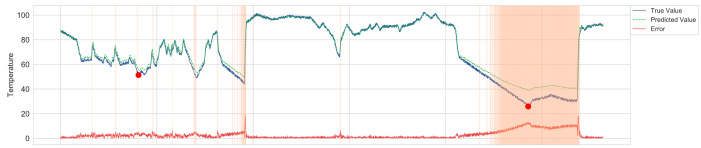


Fig. 6. Validation set results on machine temperature dataset after modification

Fig. 7. Original test set results on machine temperature dataset

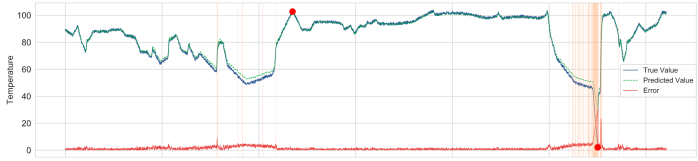


Fig. 8. Test set results on machine temperature dataset after modification

Fig. 9. Prediction result on machine temperature dataset after modification

6. REFERENCES

[1] AKASH SINGH, "Anomaly detection for temporal data using long short-term memory (lstm)," `https://github.com/akash13singh/lstm_anomaly_thesis`, 2017.

[2] Varun Chandola, Arindam Banerjee, and Vipin Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, July 2009.

[3] Qiang Yang and Xindong Wu, "10 challenging problems in data mining research," *International Journal of Information Technology & Decision Making*, vol. 5, no. 04, pp. 597–604, 2006.

[4] Yann LeCun, Y. Bengio, and Geoffrey Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, 05 2015.

[5] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[6] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins, "Learning to forget: Continual prediction with lstm," 1999.

[7] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[8] Jasper Snoek, Hugo Larochelle, and Ryan P Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, pp. 2951–2959, 2012.