MSBD 5002, Fall 21

Course: Data Mining and Knowledge Discovery

Multi-dataset Time Series Anomaly Detection

Project Report

Group Members:

Student Name	Student ID	Contact Information
QIU Yaowen	20784389	yqiuau@connect.ust.hk
BU Shihan	20787953	sbuaa@connect.ust.hk
ZHANG Yiran	20797154	yzhangml@connect.ust.hk

1. Introduction of Dataset

The dataset is from Multi-dataset Time-series Anomaly Detection Competition, SIGKDD 2021. It contains 250 independent one-dimensional time-series data subsets, and it is divided into training set and test set in advance. The training set are all composed of normal data while there is exactly one anomaly in the test set. The project intends to develop an algorithm (or multiple) to identify the anomaly for each subset.

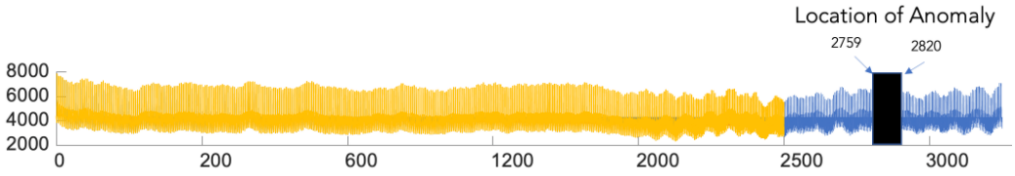

Figure 1: Example of anomaly of a single time series data

2. Data Engineering

2.1 Handle abnormal format data

Most of data subsets is stored in the .txt file with single column, which are easy to import using libraries like Pandas However, there are several data subsets are stored using multiple columns (as shown in Figure 2). The non-uniform format will lead to error in the program. Therefore, we wrote a few lines of code to catch the exception thrown by the pandas file reader and modify the data.

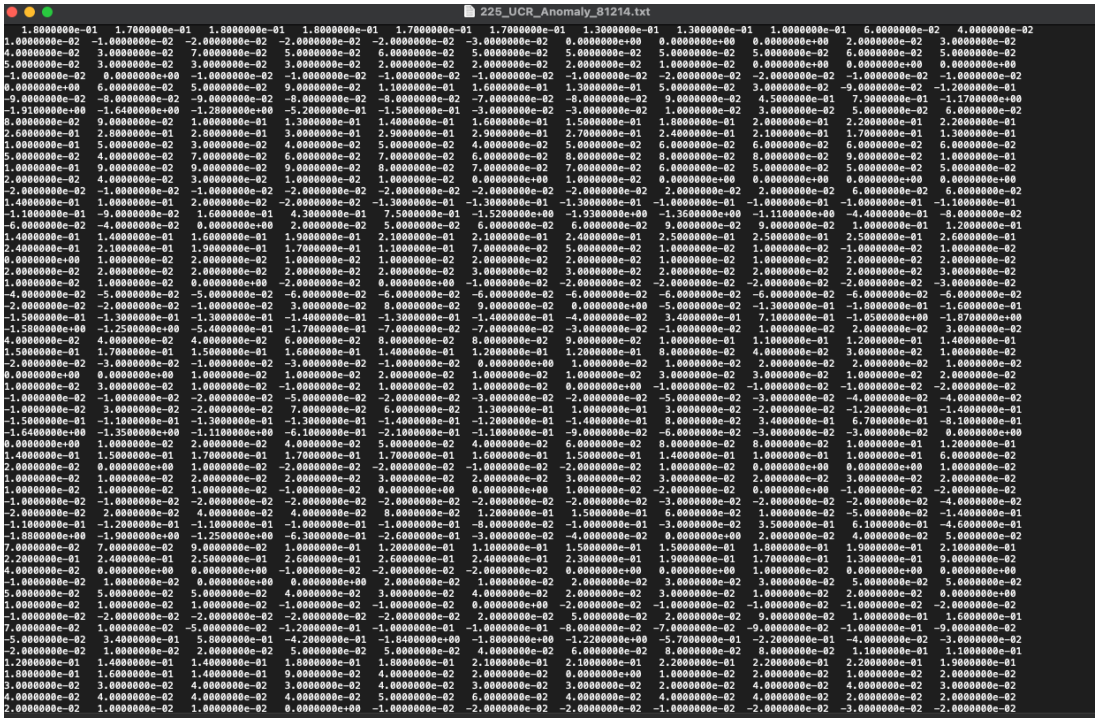

Figure 2: Screenshot of *225_UCR_Anomaly_81214.txt*

2.2 Period Analysis

Many time series data anomaly detection algorithms (includes those used in this project) require a parameter called window size, thus a sequence of data with certain length to form data structures that they requires. A proper definition of windows size has significant impact on the performance of the algorithm. In our project, we decided to use the period of the time series data as the window size of the algorithm (if requires).

Periodicity refers to a wave-like or oscillating movement around a long-term trend presented in a time series. And period is the length of the repeated sequence. We adopted the combination of Fourier Transformation and Autocorrelation Function to find the most possible period of each sub-dataset.

Fourier transform is a method of transforming time domain and spatial domain data into frequency domain data. From the prospective of Fourier transform, any waveform (time domain) can be regarded as a superposition of sine waves of different amplitudes and phases (frequency domain). Therefore, we clipped the training set sequence and used Fourier transform to decompose the time series data into a linear combination of sine wave and its Fourier coefficient. The larger the Fourier coefficient, the more likely the period of the corresponding sine wave is the period of the time series data. However, there are noise in the time series, it may not be accurate if we always choose the period with largest Fourier coefficient. In the project, we filtered Top 5 candidate periods according to the Fourier coefficient and use Autocorrelation Function to select the final period. Autocorrelation Function measures the degree of correlation of the same event at different times. The Autocorrelation Function between different lag sequences can be calculated with Pearson correlation coefficient:

$$\text{autocorr}\,(X,t) = \frac{\sum_{i=1}^{N}\,[X_i - \text{mean}(X)][X_{i-t} - \text{mean}(X)]}{\sum_{i=1}^{N}\,[X_i - \text{mean}(X)]^2} \tag{1}$$

For any given t, the larger the Autocorrelation Function, the more likely t is the period of the time series data. Hence, we chose the candidate period with largest Autocorrelation Function as the period of the time series data. In practice, we found the algorithm returned period even larger than half length of the data in few sub-datasets. For this special case, we preferred another candidate period.

2.3 Data Normalization

The scale of each sub-dataset varies. And for some algorithms such as neural network-based algorithms which are sensitive to the scale of data, a large scale may lead to slow training and slow convergence. To avoid such case, we perform z-score normalization on each sub-dataset independently, shown as formular (2).

$$x_{new} = \frac{x - \mu}{\sigma} \tag{2}$$

Where μ and σ are the mean and standard deviation of the training set, respectively.

3. Models

3.1 Autoencoder

An autoencoder [1] is a kind of neural network used to learn an efficient representation of given data in unsupervised learning. There are two parts in the autoencoder: the encoder and the decoder. The encoder part tries to represent the given data x in the latent space as $f(x)$ with much lower dimensions. And the decoder part tries to reconstruct the data x using the representation in the latent space $f(x)$, thus, $g(f(x))$. Since we hope the reconstructed data $g(f(x))$ as same as given data x as possible, the objective function of the autoencoder is the mean squared error between them.

Considering the limited computation resources we have and the volume of data is large, we built the encoder part using double convolutional layers and the decoder part using three transposed convolutional layers.

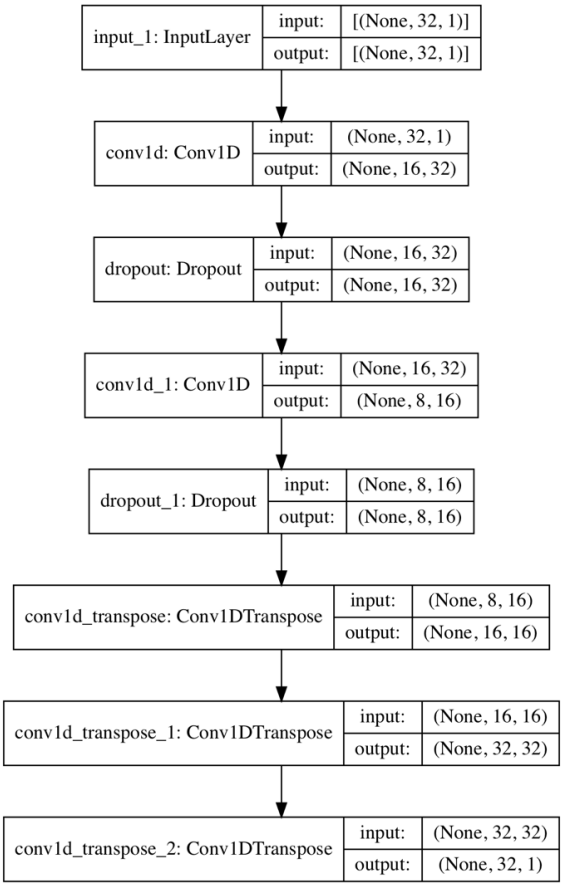

Figure 3: An Autoencoder model details in *101_UCR_Anomaly_6000.txt*

Figure 3 demonstrates an autoencoder model of *101_UCR_Anomaly_6000.txt* in details. The period of this dataset is 32. The encoder part includes four layers: 32 convolutional-1D layers with kernel size 5, a dropout layer with probability 0.2, 16 convolutional-1D layers with kernel size 5, and a dropout layer with probability 0.2. The decoder part consists of three layers: 16 transposed convolutional-1D layers with

kernel size 5, 32 transposed convolutional layers with kernel size 5, and one transposed convolutional-1D layer with kernel size 7 to guarantee the shape of reconstructed data is same as given data. There is total 6,881 parameters.

For training, the epoch is set to 100 with batch size 32. The optimizer used is Adam with 0.001 initial learning rate and the learning rate decays to 1/10 each ten epochs. Note that when the model is sequence-to-sequence, which means we can only obtain the accumulative prediction residuals of each point and the begin (i.e. the first period) and the end (i.e. the last period) of the time series data will be much lower than others. In other words, the model presumes that there is no outlier in both ends of data. The residuals will be used in the ensemble model.

Figure 4 demonstrates the example of using Autoencoder only to detect anomaly. Once the accumulative residuals are obtained, it is intuitive that the point with maximum residuals is most likely to be anormal. Therefore, this point should be reported and those 200 points around this point will be considered as a suspicious sequence.

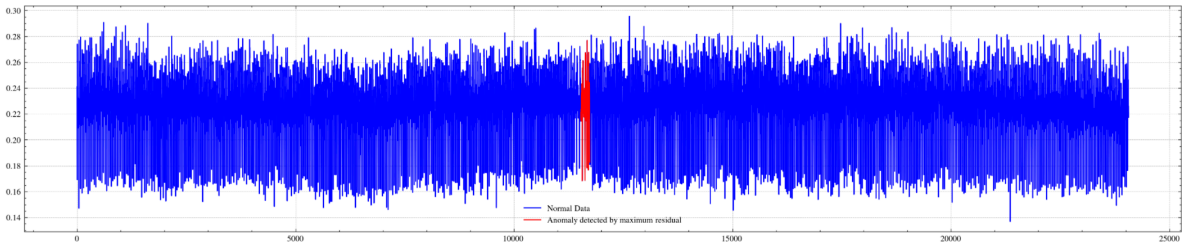

Figure 4: An example of using prediction residuals to detect anomaly using Autoencoder only

3.2 Matrix Profile

Matrix Profile [2] is a novel data structure to analyze time series data. It contains wo parts: a distance profile and a profile index. The distance profile is a vector of minimum Z-Normalized Euclidean Distances. The profile index contains the index of its first nearest neighbor. Given a window size m, computes the distance between the data in current window size and the whole time series data iteratively. The length of distance profile should be n-m+1, where n is the length of time series data. For current distance profile, an exclusion zone is set to ignore unimportant match. Then, the profile index will fill the index of minimum distance of current distance profile and the window is going to slide to the next position. After the profile distance is calculated, motif and discord can be found. A motif is a repeated pattern in the time series and a discord is an anomaly. With the Matrix Profile computed, it is simple to find the top-K number of motifs or discords. Extracting the lowest distances gives the motifs and the largest distances gives the discords. Figure 5 demonstrates how to use the matrix profile distance to find the top-5 discords in the matrix profile distance. Many algorithms have proposed to extract the discords efficiently. However, in the project, we care about the matrix profile itself since it is a type of residual that represents “how far” a point is from normal zone. We collected the matrix profile for each sub dataset and used in the ensemble model.

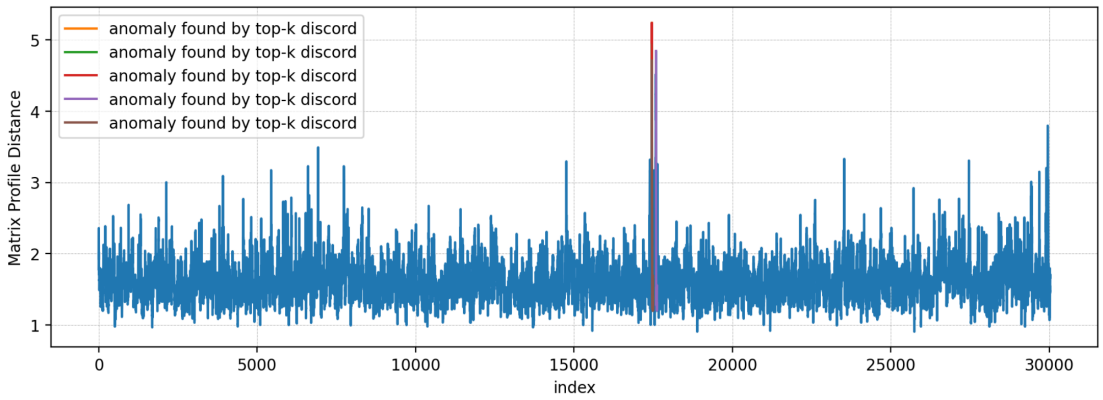

Figure 5: An example of using Matrix profile to find top-5 discords

3.3 Long Short-Term Memory Network

Long Short-Term Memory Network (LSTM) [3] is also a Neural Network to deal with sequence input which can fits our scenario of time series data. It has been successfully applied to many sequence learning problems and has been considered as one of the latest methods to deal with time series forecasting problems.

LSTM is an improved method of simple Recurrent Neural Network (RNN) that can solve the problem that RNN cannot handle long-distance dependencies of one sequence. The hidden layer at each moment in its structure contains a memory cell, and each memory contains multiple storage units. Each storage unit contains a memory unit and 3 gates (input gate, forget gate and output gate). The input gate decides whether to let new input, the forget gate deletes unimportant information, and the output gate decides what information to output.

There are 4 inputs and 1 output in the LSTM unit. The 4 inputs are the control signals of the three gates z_i, z_f, z_o, respectively control the input gate, forget gate, output gate with input data z. After inputting the data z, it will first go through an activation function and become $g(z)$. zi will also go through an activation function to become $f(z_i)$, usually the activation function of the control gate signal is $sigmoid$ function, this is to regulate the opening and closing degree of the gate between 0 and 1. After that, multiply $f(z_i)$ and $g(z)$ to feed into the memory unit. The last retained data c is stored in the memory unit, and it will be updated to c' through the forgetting operator, where the control of the forgetting gate is needed. Forgotten gate control signal z_f after a $sigmoid$ activation function becomes $f(z_f)$, and then multiply by c, thus we can derive:

$$c' = f(z_i)g(z) + cf(z_f) \tag{3}$$

After obtaining c', the value of c is updated in the memory unit, and at the same time, it passes through an activation function to become $h(c')$. Before $h(c')$ output, it will pass through the output gate for partial output. Like the previous gates, z_o has also gone through the sigmoid activation function, and the final output is:

$$a = h(c')f(z_o) \tag{4}$$

Multiple LSTM units can be connected to form a whole network comprehensively and we can have our ultimate LSTM Neural Network. For our project, we use 64 LSTM units for our time series data simulation and forecasting.

For training, we use Mean Absolute Error (MAE) for as loss function in utility of residual:

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - f(x_i)| \tag{5}$$

Intuitively, the curve of MAE is V-shaped. continuous but non-differentiable at $y - f(x) = 0$, thus difficult for the computer to solve the gradient and hard to converge. However, it is worth mentioning that MAE has an advantage over Mean Squared Error (MSE) that MAE is less sensitive to outliers and more tolerant. Because MAE calculates the absolute value of the error $y - f(x)$, no matter whether it is $y - f(x) > 1$ or $y - f(x) < 1$, there is no square term, the penalty is the same, and the weight is the same. And this is also the reason we use MAE for computing loss of our LSTM as residuals. Moreover, the residuals will be also used in our ensemble model.

The epoch is set to 100 with batch size 32. The optimizer used is Adam with 0.01 initial learning rate and the learning rate decays to 1/10 each ten epochs.

Here, we use *177_UCR_Anomaly_3200.txt* data for anomaly detection with the corresponding period that we have preprocessed in data engineering part as an example for showing simplicity. We can see from Figure 6 that it is capable that the LSTM model can converge within our parameter settings with MAE. Figure 7 shows an example result which takes MAE loss as residual for our ultimate anomaly detection.

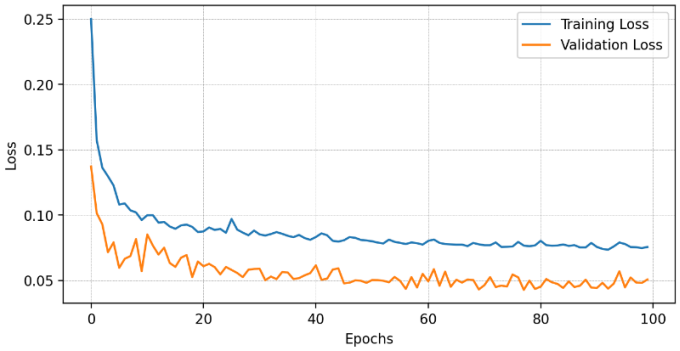

Figure 6: Training and validation loss of LSTM with MAE loss can be converged within 100 epochs

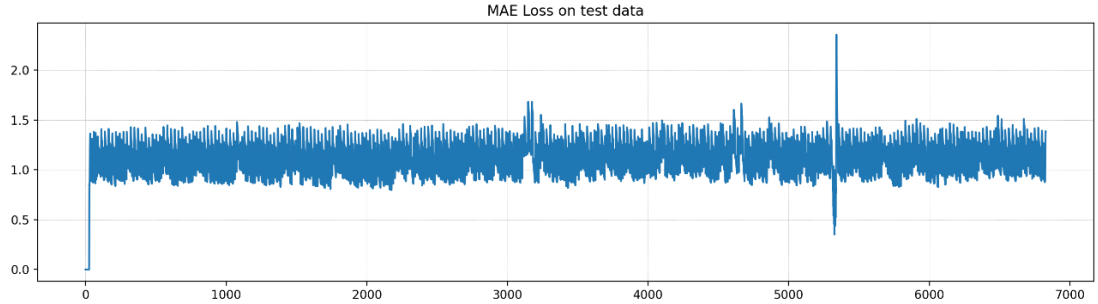

Figure 7: An example of MAE loss as final residual of LSTM prediction for anomaly detection

Figure 8 demonstrates the example of using LSTM only to detect anomaly. Once the residuals are obtained, it is intuitive that the point with maximum residuals is most likely to be anormal. Therefore, this point should be reported and those 200 points around this point will be considered as a suspicious sequence.

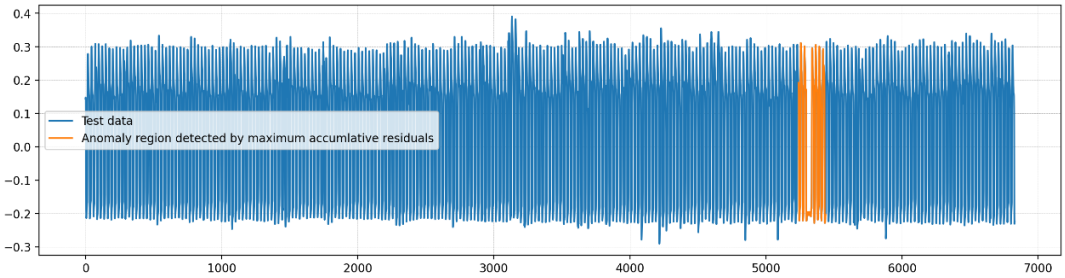

Figure 8: An example of using prediction residuals to detect anomaly using LSTM only

3.4 Seasonal Trend Composition

Seasonal Trend Composition based on Loess (STL) [4] is a common algorithm in time series decomposition. Based on LOESS, the data Y_v at a certain moment is decomposed into trend component, seasonal component and residual component:

$$Y_v = T_v + S_v + R_v, \qquad v = 1, \dots, N \tag{6}$$

STL is divided into inner loop (inner loop) and outer loop (outer loop), in which the inner loop mainly does trend fitting and period component calculation. Suppose that $T_v(k)$ and $S_v(k)$ are the trend component and period component at the end of the $(k-1)^{th}$ pass in the inner loop. With $n(i)$ represents the number of inner loops, $n(o)$denotes number of outer loops, $n(p)$ denotes the number of samples in a period, $n(s)$, $n(l)$, $n(t)$ denotes the LOESS smoothing parameter respectively. The sample points at the same position in each period cycle form a subseries. It is easy to know that there are total of n(p) such subseries, which we call cycle-subseries.

In out project, we use "statsmodels" package in Python which implements a simple version of time series decomposition for STL. With additive model approach, we can extract the trend component, and most significantly derive the residual component for anomaly detection and contribute the residual to our final ensembled model.

Here, we use *004_UCR_Anomaly_2500.txt* data for anomaly detection with the corresponding period equals to 80 that we have preprocessed in data engineering part as an example for showing simplicity. Figure 9shows the residual component we derived from STL model.

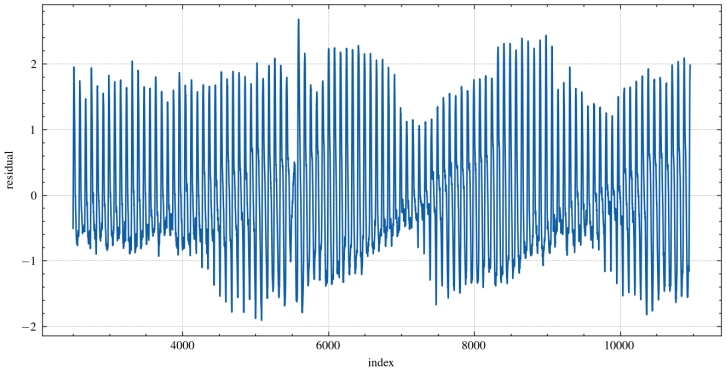

Figure 9: An example of residual component result of STL decomposition

Figure 10 demonstrates the example of using STL only to detect anomaly. Once the residuals are obtained, it is intuitive that the point with maximum residuals is most likely to be anormal. Therefore, this point should be reported and those 200 points around this point will be considered as a suspicious sequence.

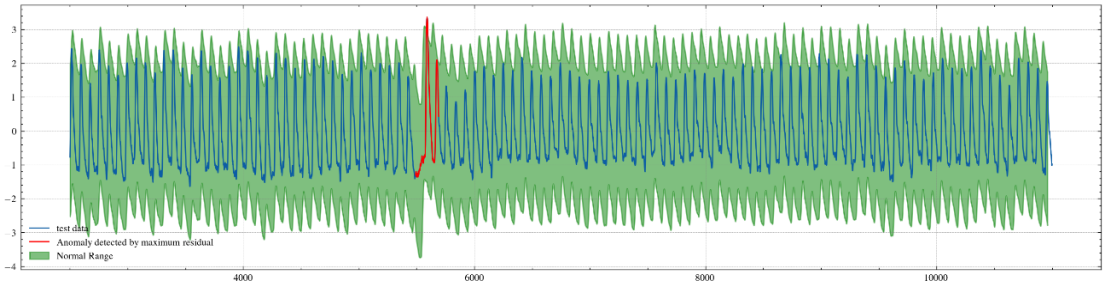

Figure 10: An example of using prediction residuals to detect anomaly using STL only

3.5 Fast Fourier Transform

Fourier Transform is a commonly used signal processing technology that is capable to complete the transformation of the signal from the time domain to the frequency domain. For time series data, Fourier Transform is used to convert time domain data into frequency domain, which makes it likely to perform some mathematical operations on frequency domain data to remove some specific frequency components, which can be anomalies in our task.

To change the time series data into frequency domain, we use x to represent time and *f* means the value at x time point. Time domain data is changed by using:

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \omega} \, dx \tag{7}$$

We use the fast Fourier transform algorithm [5] to realize the transformation, which is included in the *fftpack* package in Python.

After changing it into frequency domain, we use Butterworth filter [6] to perform noise filtering, by *signal* package. The main feature of Butterworth filter is maximally keeping the flatness of the passband and smoothly roll down to zero in the stopband, which means it is able to not only completely remove the noise frequencies, but also keep the sensitivity for wanted frequencies. The frequency response, the gain, $G(\omega)$ of n-order Butterworth low-pass filter is:

$$G^2(\omega) = |H(\omega)|^2 = \frac{1}{1 + \left(\frac{\omega}{\omega_c}\right)^{2n}} \tag{8}$$

It should be noted that to achieve a good filter performance, the order of Butterworth filter should be high, so we set its order = 8. As for filter type, we choose band pass filter, keep the frequency in range $[\omega_{c,low}, \omega_{c,high}]$, because the anomalies are possible to appear in both high and low band. If using low pass filter or high pass filter, we may miss some noise some other series data. After experiments, band pass filter is set to filter 1% high frequency noise, 1% low frequency noise, $r_{filter} = 0.01$. The cutoff frequency $\omega_{c,low}$ and $\omega_{c,high}$ are decided by the r_{filter}:

$$\omega_{c,low} = \frac{2Nr_{filter}}{N} = 2\, r_{filter} \tag{9}$$

$$\omega_{c,high} = 1 - \frac{2Nr_{filter}}{N} = 1 - 2\,r_{filter} \tag{10}$$

Take the *004_UCR_Anomaly_2500.txt* data as an instance, the following diagram shows the Fourier Transform result and frequency data after filtering.

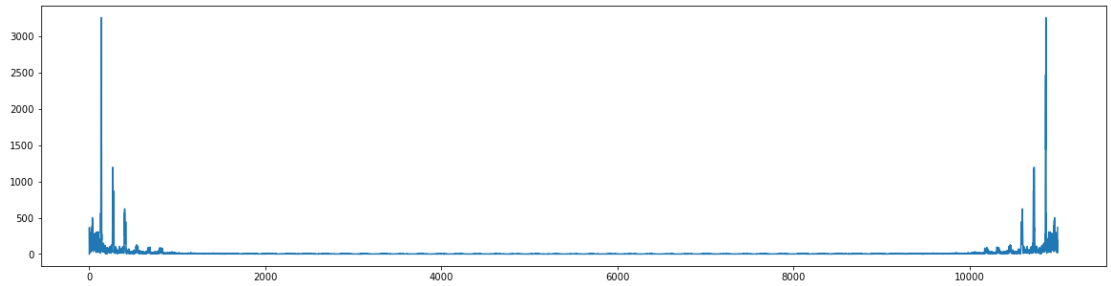

Figure 11: An example of Fourier Transform result (data in frequency domain)

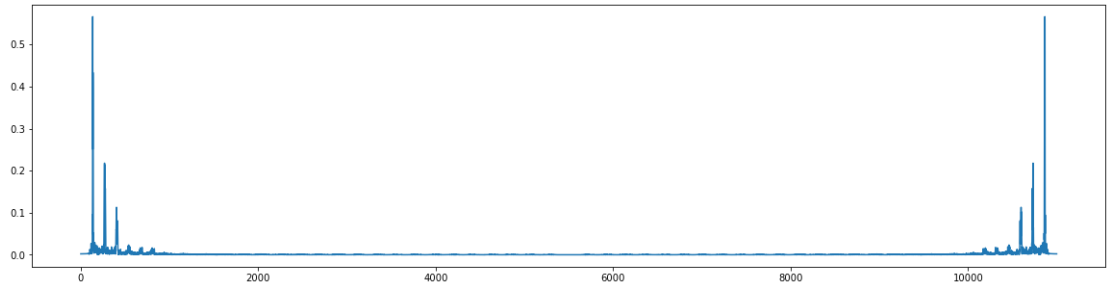

Figure 12: An example of Butterworth filter (data in frequency domain)

After filtering, the frequency data is inversely transformed to time domain. Comparing the processed data with the original data by residual, we can find the abnormal point and mark the 200 points centered on that point as the location of anomaly.

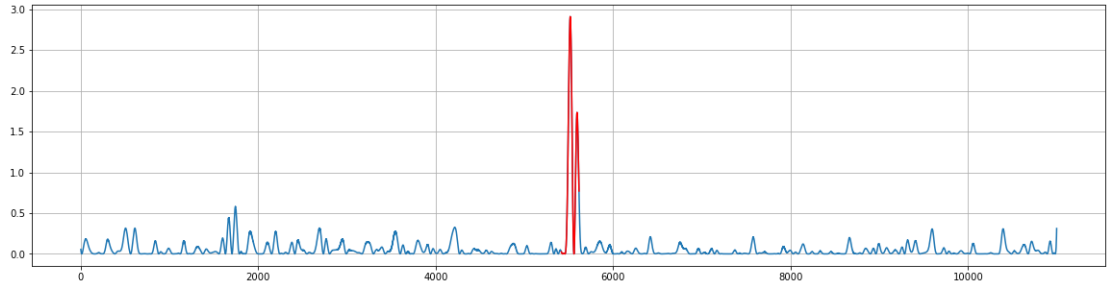

Figure 13: An example of the residuals of Fourier Transform and Butterworth filter

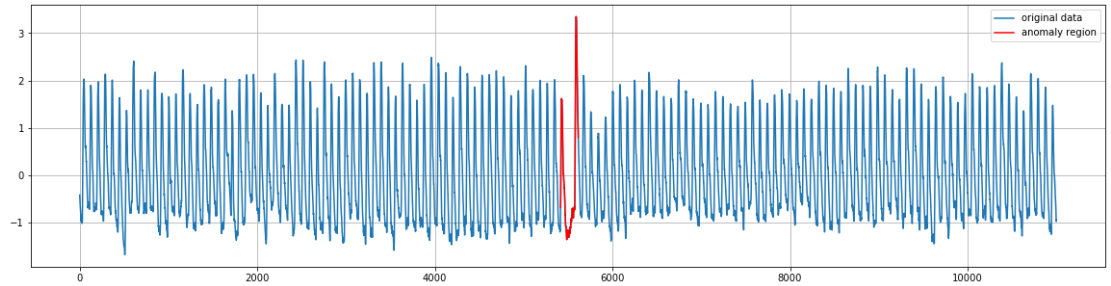

Figure 14: An example of using residuals to detect anomaly using FFT only

3.6 Ensemble Model

After the residuals of each sub-model is computed, a final ensemble model takes the weighted summation of residuals and detects the anomaly using maximum residual principle. Figure 15 demonstrates the process of the ensemble model. For each sub model, the weight of its residuals is defined as the maximum residuals subtract the mean and divided by the standard deviation. The idea is that a model should have better performance if its maximum value (i.e., detected anormal point) deviates from the group point farther, based on the premise that the model could identify the anomaly. Then, we filled the NaN value using zero. Different models output different shape of residuals of the same data. For example, the matrix profile can output the matrix profile with length equals the length of data while the autoencoder must ignore the beginning of the data, in other words, the sequence is shorter. Therefore, to make sure all the residual sequences have the same length, we expanded residual sequences which are shorter and fill the NaN value using zero. This approach defaults that anomalies will not appear in this position. Then, we rescale all the residuals using Min-Max normalization to make sure the scale is same for all models. Finally, the weighted sum of residuals can be calculated and the point with largest weighted residual is considered as anomaly point.

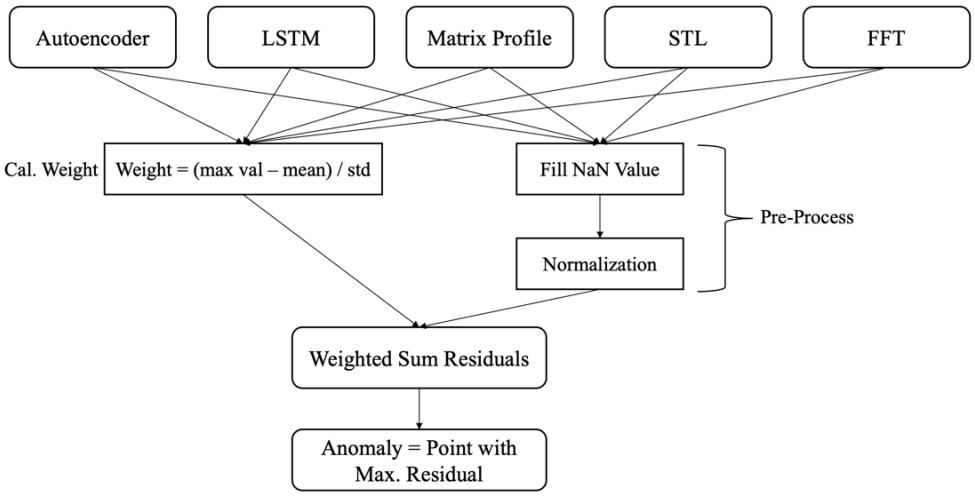

Figure 15: Illustration of the process of ensemble model

Figure 16 demonstrates an example using ensemble model to detect anomaly in *017_UCR_Anomaly_5000.txt*. Different colors represent anomalies detected by each independent model. To visualize better, 200 points nearest to the anomaly is plotted as an abnormal sequence. Each sub model does not detect the anomaly correctly while the ensemble one can identify it precisely.

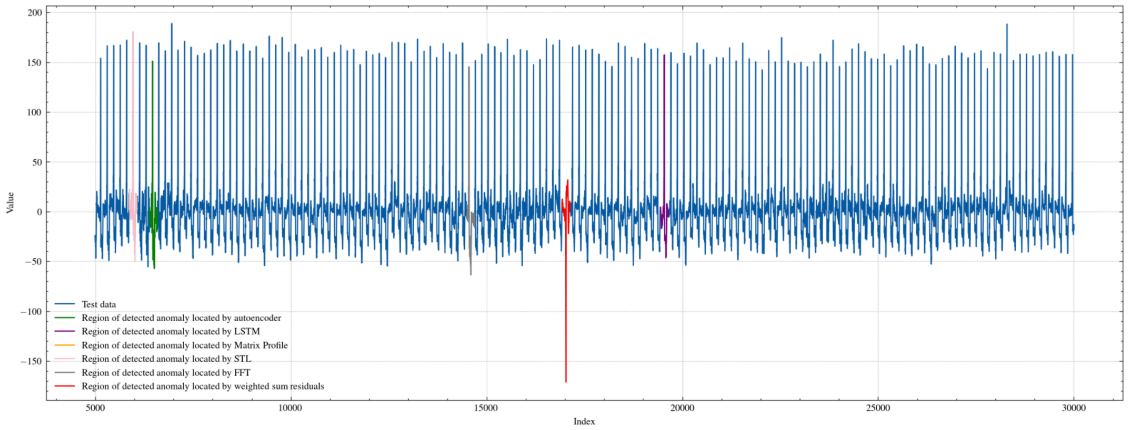

Figure 16: An example using ensemble model to detect anomaly in *017_UCR_Anomaly_5000.txt*

4. KPI Dataset

4.1 Description

The KPI dataset is a supervised time series anomaly dataset provided by AIOps Challenge. The dataset is consisting of four features: a UNIX timestamp is the time when the value was recorded, a value, a label indicates whether it is an anomaly and a KPI ID represents which time series group the value belong to. Our task is to identify the anomaly of given test dataset.

4.2 Data Engineering

The distribution of label is highly skewed in both training set and test set. Figure 17 demonstrates the fraction of normal data (blue) and abnormal data (green) in each dataset. In general, the fraction of anomalies is lower than 1%, which will lead to a biased model.

Perform data engineering could bring positive impact on the performance of model. First, we decomposed the feature timestamp. Since the timestamp is exactly the UNIX timestamp, it is easy to know the exact time (i.e., year, month, day, hour) when the value was record. Furthermore, according to the statistics of data, the time expansion of data does not exceed three months, which means year and month would be helpless. And the unique value of day is 31, which will create a high dimensional dataspace after one-hot encoding. Therefore, we classified day into early, mid, and late of a month in particular. With similar reason, we divided hour to four time zones in a day and ignore the minutes and the seconds.

In addition, for each independent time series data (i.e., data with same KPI ID), we performed z-score normalization on the data and use the method in section 2.2 to identify the period of data.

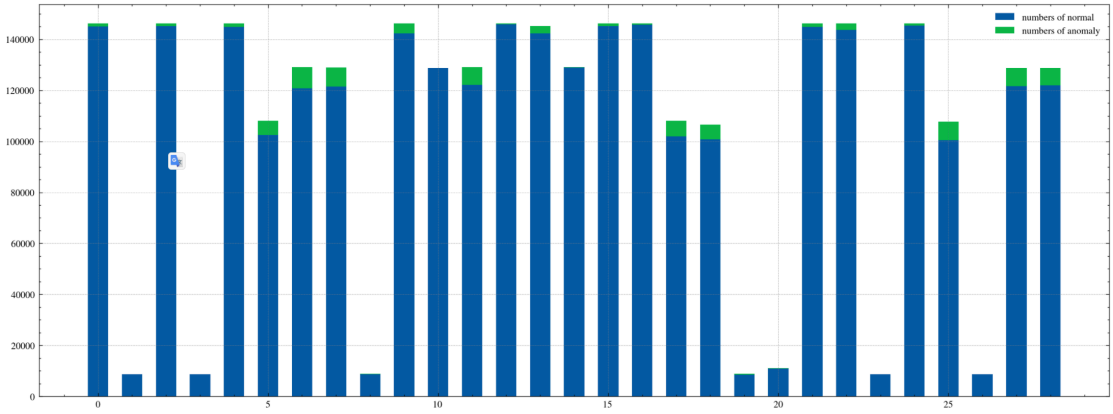

Figure 17: The fraction of normal data (blue) and abnormal data (green) in each dataset

4.3 Model

For the KPI dataset, we used a simple convolutional model to identify the anomaly for each independent time series data. Figure 18 demonstrates the structure of data with KPI ID = 05f10d3a-239c-3bef-9bdc-a2feeb0037aa. The input of the model is a matrix with height = period and width = numbers of features. The idea is the anomaly in the time series data should have certain correlation with previous data and data in the future. It may be worse if we just use the features of current timestamp. Next to the input layer are two convolution 1D layers. Each with 32 filters and 7 * 7 kernel. Then a global max pooling 1D layer with down sampling the features are followed by two dense layers to make the prediction. Since the label distribution is highly skewed, the weight of class of the loss function is set to be 5:95. The former is the weight of normal data and the latter is weight of anomaly.

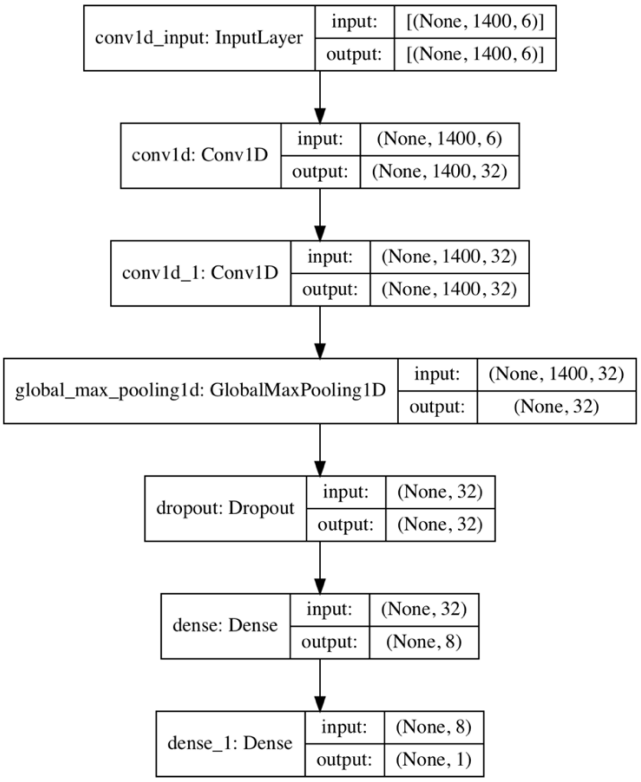

Figure 18: A convolutional model used for KPI ID = 05f10d3a-239c-3bef-9bdc-a2feeb0037aa

4.4 Result

Table 1 demonstrates the classification report of KPI Dataset. Although the accuracy is high, there is a huge gap between the f1-score of normal data and anomaly data. The model tends to predict given data as normal data due to the imbalance of label distribution.

	Precision	Recall	F1-score	Support
0	0.99	0.98	0.98	2864287
1	0.22	0.29	0.28	54560
Accuracy			0.98	2918847
Macro avg	0.59	0.63	0.61	2918847
Weighted avg	0.97	0.97	0.97	2918847

Table 1: Classification Report of KPI Dataset

References

[1] Bank, D., Koenigstein, N., & Giryes, R. (2020). Autoencoders. arXiv preprint arXiv:2003.05991.

[2] Yeh, C. C. M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H. A., ... & Keogh, E. (2016, December). Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th international conference on data mining (ICDM)* (pp. 1317-1322). Ieee

[3] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9, 1735-1780.

[4] Cleveland, R., Cleveland, W.S., McRae, J.E., & Terpenning, I.J. (1990). STL: A seasonal-trend decomposition procedure based on loess (with discussion).

[5] Nussbaumer, H. J. (1981). The fast Fourier transform. In Fast Fourier Transform and Convolution Algorithms (pp. 80-111). Springer, Berlin, Heidelberg.

[6] Butterworth, S. (1930). On the theory of filter amplifiers. Wireless Engineer, 7(6), 536-541.

[7] The solution for KDD Cup 2021 Multi-dataset Time Series Anomaly Detection challenge from DeepBlueAI (DBAI) team. URL: https://www.youtube.com/watch?v=1v64_tzJPTE&list=PL7rX8Mnmifgc-SfuGgMz2SZNYakKy-Zu_&index=4&ab_channel=ZhixingHe

[8] Presentation for the KDD Cup 2021 mutli-dataset Time Series Anomaly Detection Competition, team Old Captain (2nd place). URL: https://www.youtube.com/watch?v=4PdlUcmwWu0&list=PL7rX8Mnmifgc-SfuGgMz2SZNYakKy-Zu_&index=2&ab_channel=Shaido987