CS5658 Anomaly Detection Homework2 – TimeSeries AD

Due Date: 4/16 23:30

Wafer dataset

The Wafer dataset is a dataset related to semiconductor microelectronics fabrication. The dataset consists of a collection of inline process control measurements recorded from various sensors during the processing of silicon wafers. Each sample contains the measurements recorded by one sensor during the processing of one wafer.

ECG200 dataset

The ECG200 dataset consists of ECG signals. Each series traces the electrical activity recorded during one heartbeat. There are two classes, normal heartbeat versus a myocardial infarction event (heart attack due to prolonged cardiac ischemia).

In this task, we are going to apply different time-series feature extraction techniques, and detect anomaly samples using different methods. The Wafer dataset / ECG200 dataset are available in the `HW2.zip` we provide.

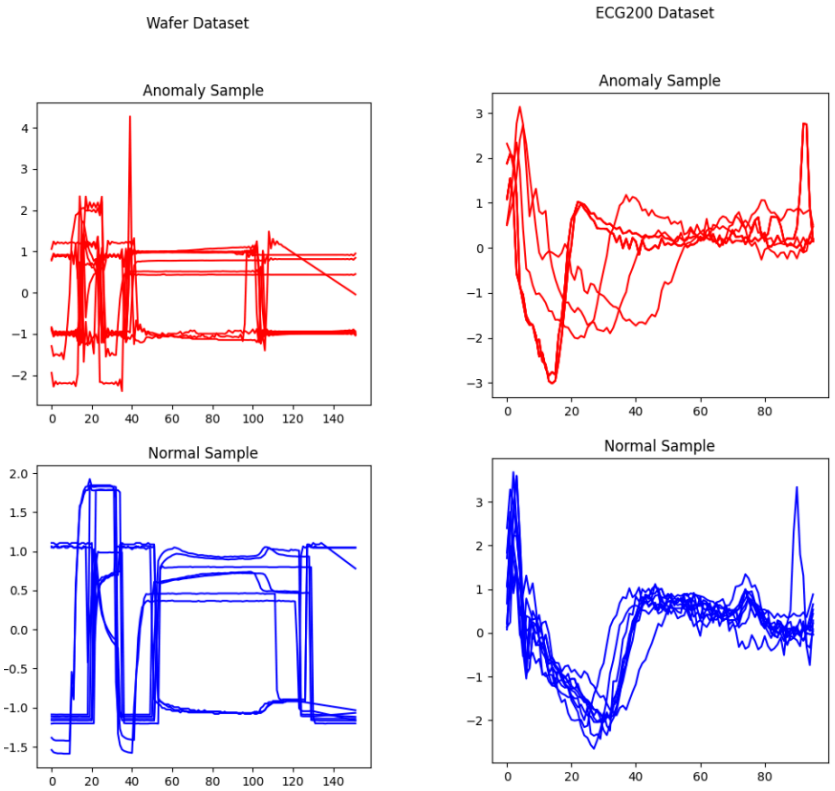
The experiment settings are as follow:

1) We individually conduct experiments for two datasets (Wafer, ECG200).
2) For the training data, we use only normal samples (class label=1).
3) For the testing data, we resample normal (class label=1) / abnormal (class label=-1) samples into 9:1.
4) For each dataset, we apply different feature extraction techniques in the “Problem” section.
5) Apply anomaly detection algorithms according to the problem statements.
6) Calculate the ROC-AUC score of the prediction.
7) Repeat Step 2~6 for two datasets and record the corresponding ROC-AUC.

Problem:

1. (10%) Visualization

Randomly choose 10 normal / 10 abnormal samples and visualize using line charts.

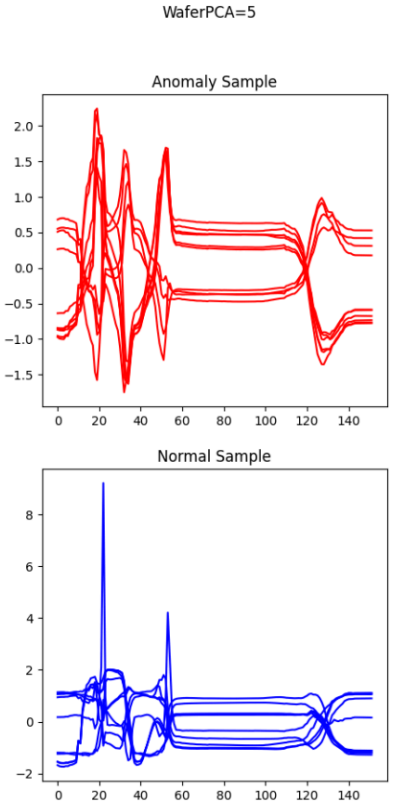


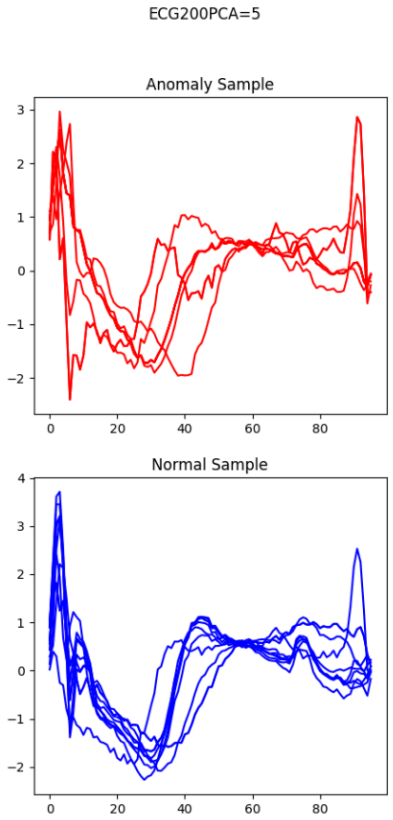
2. (10%) Raw Data

Implement KNN anomaly detection (K=5) with Euclidean distance and take the average distance among these neighbors as an anomaly score on the raw data and calculate AUC-ROC score of the experiment.

3. (20%) PCA Reconstruction

This method uses PCA to extract components of the normal samples, and uses N most important components to reconstruct the testing samples. The anomaly score is the reconstruction error (Euclidean distance). You need to **try different values of N**, and report all the performances of your chosen N (10%). Using the best N to **visualize the PCA reconstruction result** of randomly chosen 10 normal / 10 abnormal samples (10%). Discuss the performance of the hyperparameter N in your report. You are allowed to use the packages for directly calculating the PCA process.





4. (20%) Discrete Fourier Transform

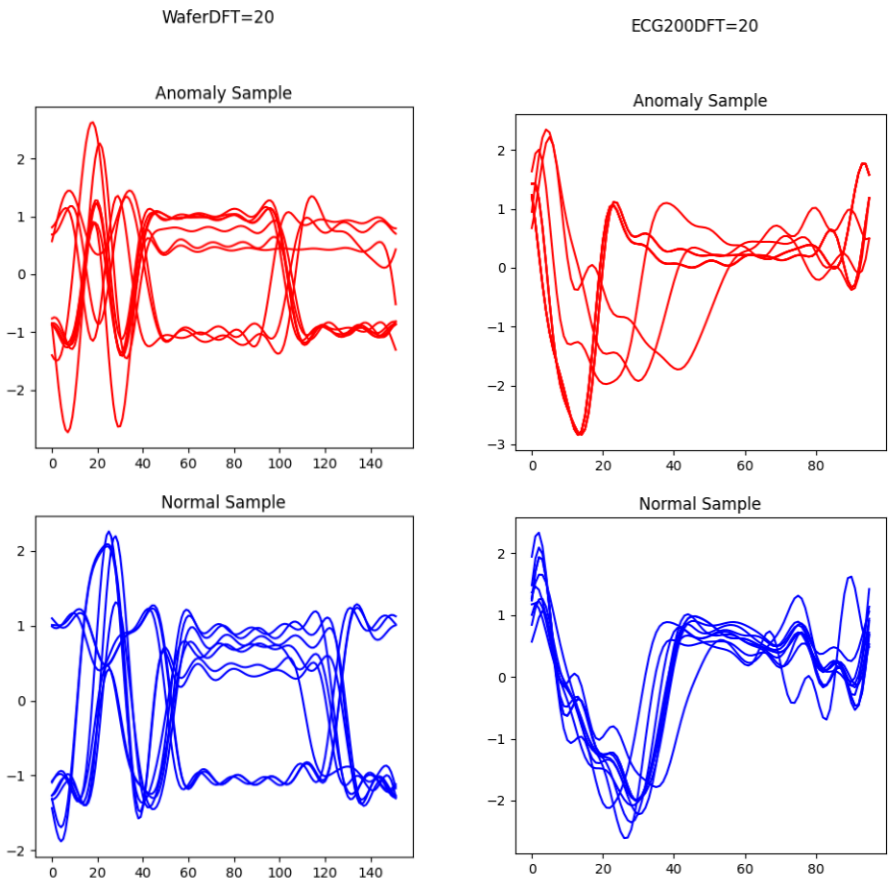
Implement KNN anomaly detection (K=5) with Euclidean distance and take the average distance among these neighbors as an anomaly score on the selected 1D DFT coefficients and calculate AUC-ROC score of the experiment. The steps are as follows:

(1) First, apply DFT to the target sample (transform to frequency domain).

(2) Second, select the lowest M DFT coefficients for the transformed time series data and concatenate their magnitudes as the DFT feature vector.

(3) Apply KNN anomaly detection with the distance between two time series samples computed as the Euclidean distance between their corresponding DFT feature vectors.

You need to **try different values of M**, and report all the performances of your chosen M (10%). Using the best M to **visualize the sequence after selecting lowest M DFT coefficients and applying an inverse DFT** for randomly chosen 10 normal / 10 abnormal samples (10%). Discuss the performance of the hyperparameter M in your report. You are allowed to use the packages in the DFT / inverse DFT process.



Normal and anomaly samples after selecting lowest M DFT coefficients and applying an inverse DFT, M is 20 in this case.

5. (20%) Discrete Wavelet Transform

Implement KNN anomaly detection (K=5) with Euclidean distance and take the average distance among these neighbors as an anomaly score on the selected Discrete wavelet transform coefficients of the time series data and calculate the AUC-ROC score of the experiment.

Please use Haar wavelet transformation and concat the detail and approximate coefficient to form the feature vector. The implementation process is as follows.

First, you need to compute different levels of the DWT with `level=range([1,ceiling(log2(deature_dim))])`. For feature dimensions not equal to power of 2, pad the rest dimensions with zeros to the next power of 2. For example, if feature dimension is 73, you need to pad feature dimension to 128 with zeros and calculate the DWT level=[1,2,3,4,5,6,7].

After you get the values of each level (like Figure below), you need to select S the most significant coefficients on the transformed data using the bottom-up order. The selection strategy is as follows: (1) Select the A_{x1} for the **last level x**. (2) Select all the coefficients of D in the level x. (3) x = x-1. Repeat (2) (3) until we select S coefficients. The number S is required to equal the power of 2. The example below shows the examples of coefficient-selection strategy:

S=1: A_{31}

S=2: A_{31}, D_{31}

S=4: A_{31}, $D_{31,}$, $D_{21,}$, D_{22}

S=8: A_{31}, $D_{31,}$, $D_{21,}$, D_{22}, $D_{11,}$, D_{12}, D_{13}, D_{14}

You need to **try different values of S**, and report all the performances of your chosen S. You are **NOT** allowed to use the packages for directly calculating the DWT process. You need to implement it by yourself.

S	80	61	75	71	63	59	76	63
Level 1	**70.5**	**73**	**61**	**69.5**	*-9.5*	*-2*	*-2*	*-6.5*
Level 2	**71.75**	**65.25**	*1.25*	*4.25*				
Level 3	**68.5**	*3.25*						

S	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
Level 1	A_{11}	A_{12}	A_{13}	A_{14}	D_{11}	D_{12}	D_{13}	D_{14}
Level 2	A_{21}	A_{22}	D_{21}	D_{22}				
Level 3	A_{31}	D_{31}						

Fig. Lecture slides P.26: Discrete Wavelet Transform

6. Report (20%, 5% for each problem)

Analyze the performance and reasons behind for each method, write down your observations. (e.g. why this method performs bad, why this hyper-parameter performs good.)

7. Bonus(10%)

Try different K for all methods and hyperparameters, show best combinations for each method.

Note (Important):

1. Do not modify the sample code segment.
2. You can use APIs to perform PCA and Discrete Fourier Transform (sklearn.decomposition.PCA, numpy.fft.fft)
3. You cannot use APIs from any package to perform Discrete Wavelet Transform. You need to implement it yourself.
4. The report should contain:
   (i.) your implementation code
   (ii.) explanation of code
   (iii.) performance (ROC-AUC)
5. You should provide a "ReadMe.txt" file about how to run your code.
6. Try adding comments as much as you can to better understand your code.
7. You should submit a HW2_{Student-ID}.zip (ex:HW2_123456789.zip) containing only the following files:
   a. hw2.py
   b. ReadMe.txt
   c. report.pdf
8. Make sure you explain everything you want to show in the report, not in your code.
9. All results should be shown in your report, not in the console or the pop-up window, or you will get 0 points.
10. Discussion of homework is encouraged, but you have to write your own.
11. Copying or submitting AI-generated documents/code is strictly prohibited.
12. Scores of late homeworks will be reduced by **20% per day**.
13. If you have any questions, please pose your questions in the eeclass.

Tips:

1. Some helpful functions & libraries:
   a. numpy
   b. matplotlib
   c. sklearn.decomposition.PCA
   d. numpy.fft.fft
   e. sklearn.metrics.pairwise_distances(ref:https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise_distances.html)
   f. sklearn.metrics.roc_auc_score (ref: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html)