DWS203: Assignment

April 2023

1 Introduction

Exathlon [2] [1] is a cutting-edge benchmark for explainable anomaly detection in high-dimensional time series data. It was developed using real data traces from repeated executions of large-scale stream processing jobs on an Apache Spark cluster. In order to create a diverse range of anomalies, six different types of anomalous events were intentionally introduced during some of these executions. These events include misbehaving inputs, resource contention, and process failures.

For each instance of anomaly, ground truth labels have been provided for both the root cause interval and the extended effect interval. This feature enables the development and evaluation of a wide range of anomaly detection tasks, making Exathlon a powerful tool for both researchers and developers. Figure 1 shows an overview of the framework.

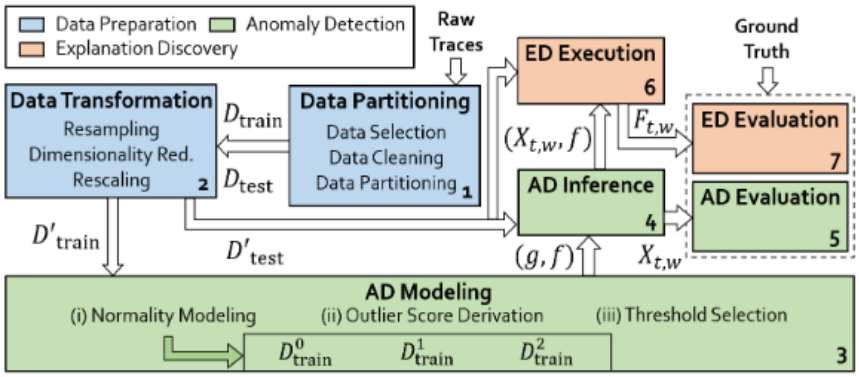

Figure 1: Exathlon Framework [2].

The benchmark also introduces four levels of range-based evaluation metrics to evaluate the efficiency of an anomaly detection method. The range-based recall and precision are used to calculate an F-beta score (F1 in our case). The four levels for anomaly detection evaluation are AD1 (Anomaly Existence), AD2 (Range Detection), AD3 (Early Detection) and AD4 (Exactly-Once Detection), where each one is stricter that the previous one, as shown in Figure 2.

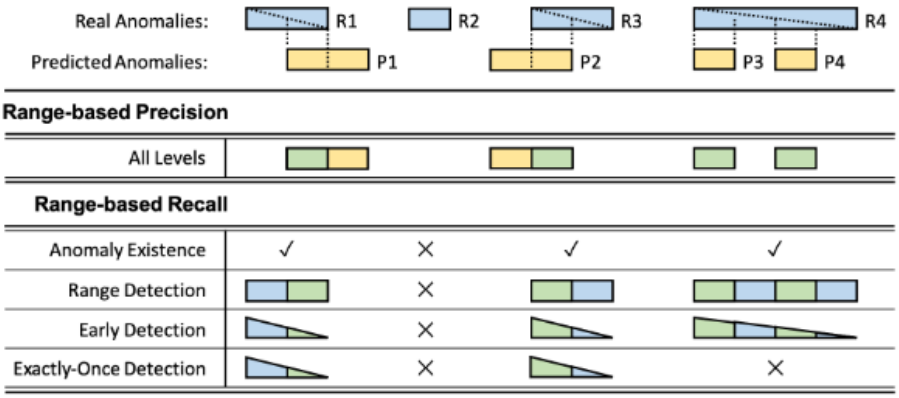

Figure 2: AD levels [2].

In addition to the benchmark dataset, the Exathlon implementation[1] also includes three state-of-the-art anomaly detection models that are specifically designed for time series data. These models are trained using normal data to either reconstruct or forecast the input signal. When the models encounter abnormal data, they produce a high error, which is used as the anomaly score for new data.

2 Data

As described earlier, the data concerns records from traces recorded from repeated execution of Spark applications. You will use only the first application's data, which has 9 traces, where each trace has multiple records. The data are already pre-processed using the Data Transformation component of Exathlon framework (Figure 1).

A train and a test set are provided to you, where the train set contains 6 traces with no anomalies and the test set contains 3 traces of records with known anomalies of different types. The data are provided in two python dictionaries.

The test set contains three datasets:

- 1. *test*: a numpy array of shape (#traces, #records, #features).
- 2. *y_test*: numpy array with the labels for each record, of shape (#traces, #records).
- 3. *test_info*: a list of size #traces where the type of anomalies is stored for each trace.

The data for the train set are organized in a similar manner, with the difference that the *keys* of the dictionary are *train*, *y_train*, and *train_info*.

[1] https://github.com/exathlonbenchmark/exathlon

3 Your task

Your task is to implement two anomaly detection techniques that can either classify trace records as anomalous or not or provide an anomaly score to them as accurately as possible. Essentially, your goal is to develop the **AD Modeling** component in the Exathlon framework, as illustrated in Figure 1. Once implemented, you will demonstrate the capabilities of your technique on specific datasets using the **AD Evaluation** component. **The first anomaly detection techniques will be a semi-supervised learning one and the second an unsupervised one. Both techniques must either use a sliding window or look only at past data when making predictions. You are encouraged to adopt a proximity-based rationale for both techniques, so that your techniques can be easily transferred to an online streaming case.**

To evaluate the performance of your technique, you will be required to compare it against the already implemented techniques. You will use the AD1, AD2, AD3, and AD4 F1 scores as performance metrics, focusing mostly on AD1 and AD2.

You can use the train set to model the normality of the data when implementing the semi-supervised anomaly detection technique; you can ignore the train set or when implementing an unsupervised technique.

An example of an implementation is presented in files *main_my_detector.py* and *mydetector.py*. You can use the *mydetector.py* to implement your method or create your own file.

The *helper.py* file has been implemented to help you load train and test data, and call the evaluation process.

To test the efficiency of your anomaly detection method you have to add in the test set dictionary two key-value pairs, scores for the records in the test set with key equal to *"test_scores"* and the predictions for the records in the test set with key name *"test_preds"*. **The values for both of them must be in the form of (#traces, #records). Regarding *"test_preds"*, the values must be binary (0 and 1, not True or False)**.

It is up to you how to implement the functionality to calculate the scores of predictions. In any case, you have to provide predictions for all records in the test set.

The results regarding AD1, AD2, AD3, and AD4 levels can be found in
*ad*1_1.0_*detection_comparison.csv*,
*ad*2_1.0_*detection_comparison.csv*,
*ad*3_1.0_*detection_comparison.csv*, and
*ad*4_1.0_*detection_comparison.csv*
under the folder *myADoutput*. The results regarding the anomaly scores are stored in files
*ad*1_1.0_*scoring_comparison.csv*,
*ad*2_1.0_*scoring_comparison.csv*,
*ad*3_1.0_*scoring_comparison.csv*, and
*ad*4_1.0_*scoring_comparison.csv*.

Finally, under the *scoringOutput* folder, you can see a visualization regarding anomaly scores evaluation results, and in the folder *AutoEncoderResults* you can find the results in all levels of AD evaluation (both for scores and detection) of the already implemented Auto Encoder solution in the exathlon benchmark, which can be used to compare your solution.

4 Installation

Listing 1: set up

```
# First unzip the assignment file.
# open the folder
cd assignment

# if your are on windows open the anaconda prompt
# and navigate to the project folder /assignment

conda create -n exathlon python=3.7
conda activate exathlon
pip install -r requirements.txt


# navigate to myAD package,
# which will be used to test your implementation.

cd src/myAD/

# before running the a testing script, change the first
# line in the file /src/myAD/path.txt
# to the full path of the folder /assignment.

# run a dummy method which is already implemented in
# mydetector.py.
python main_my_detector.py
```

References

[1] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. A demonstration of the exathlon benchmarking platform for explainable anomaly detection. *Proc. VLDB Endow.*, 14(12):2827–2830, jul 2021.

[2] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. Exathlon: A benchmark for explainable anomaly detection over time series. *Proc. VLDB Endow.*, 14(11):2613–2626, jul 2021.