Work Distribution Report

Sushanth kulakrni

December 28, 2023

Contents

1 Tasks

I have been allotted, the **Data Prepossessing** part of our project. Primarily I have been tasked with the following:

1. Find RAW log data, if unavailable find parsed log data.
2. Remove empty and duplicate log entries from RAW log data.
3. Parse RAW data into structured format, categorizing each entry into events.
4. Group the parsed, structured log data into set of events called a sequence.
5. Label the sequence of events as Normal or Anomalous and forward the same to Mahender.

2 Find Data

The following datasets have been considered for anomaly detection:

1. HDFS
2. BGL
3. Thunderbird
4. Spirit

RAW and parsed datasets for HDFS and BGL were available at:

- `https://github.com/logpai/loglizer/tree/master/data/HDFS`
- `https://zenodo.org/records/8115559`
- `https://github.com/logpai/loghub/tree/master`

3 Parse Data

Log Hub only provides us 2k lines of raw data, which is not adequate, hence I've used 100k or more entries of log data available at **LogPai**.
Raw data is parsed using logparser python library and we use, **Drain** parser is primarily used.
The code segment for the same :

```
1
2  import csv
3
4  input_log_file = 'E:\Log_anomaly_detection\Log_anomaly_detection\
       sushanth\preprocess\parse\demo_data\HDFS_2k.log_structured.csv'
5
6  def extract_data(input_file, output_file):
7      with open(input_file, 'r') as csvfile:
8          reader = csv.DictReader(csvfile)
9
10         data_list = [(row['EventId'], param) for row in reader for
       param in row['ParameterList'].strip("[]").replace("'", "").
       split(', ') if param.startswith('blk_')]
11
12     with open(output_file, 'w', newline='') as csvfile:
13         fieldnames = ['EventId', 'BlockString']
14         writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
15
16         writer.writeheader()
17         for data in data_list:
18             writer.writerow({'EventId': data[0], 'BlockString':
       data[1]})
19
20 extract_data(input_log_file, 'E:\Log_anomaly_detection\
       Log_anomaly_detection\sushanth\preprocess\group\ blk_eve_ids.
       csv')
```

4 Grouping parse Data

The parsed data is now in a structured format, a key component here is *"Event Id"* ! Each eventId represent a particular static part of a log entry, we group these eventIds, let the set of events be called a sequence, S_i. We generate all such S_i based on a grouping parameter.
Grouping parameters:

- HDFS : Session window protocol, **BlockId**
- BGL : Session window protocol, **Node Id**
- Thunderbird : Fixed Window, **Timestamp, window size: 900sec** *Prabas has helped me with grouping and labelling Thunderbird datset
- Spirit : Fixed Window protocol, **Timestamp, Window size: 113sec**

5 Labelling Sequence

The general idea: *"If all the events in a sequence are Normal, then the sequence is labelled 'Normal' else 'Anomalous'"*
Code for the same :

```
1
2 import pandas as pd
3
4 extracted_info_file = 'E:\Log_anomaly_detection\
    Log_anomaly_detection\sushanth\preprocess\group\proj\sequence.
    csv'  # Update with your actual file path
5 label_info_file = 'E:\Log_anomaly_detection\Log_anomaly_detection\
    sushanth\preprocess\HDFS.anomaly_label.csv'  # Update with your
     actual file path
6 output_file = 'E:\Log_anomaly_detection\Log_anomaly_detection\
    sushanth\labelled_sequnce.csv'
7
8 extracted_df = pd.read_csv(extracted_info_file)
9 label_df = pd.read_csv(label_info_file)
10
11 merged_df = pd.merge(extracted_df, label_df, how='left', left_on='
    Block', right_on='BlockId')
12
13 merged_df = merged_df[['EventId', 'Label']]
14
15 merged_df.to_csv(output_file, index=False)
16
17 print("Merging complete. Check the output file:", output_file)
```

6 Conclusion

This sequnce file is generated for all the datsets and sent to Mahender for training BERT model i=at the client side in client-server federated environment which will be done by sathwik.