Network Anomaly Detection Based on Logs

Jiansheng Xiong

xiongjs@chinaunicom.cn

2021.07.06

Contents

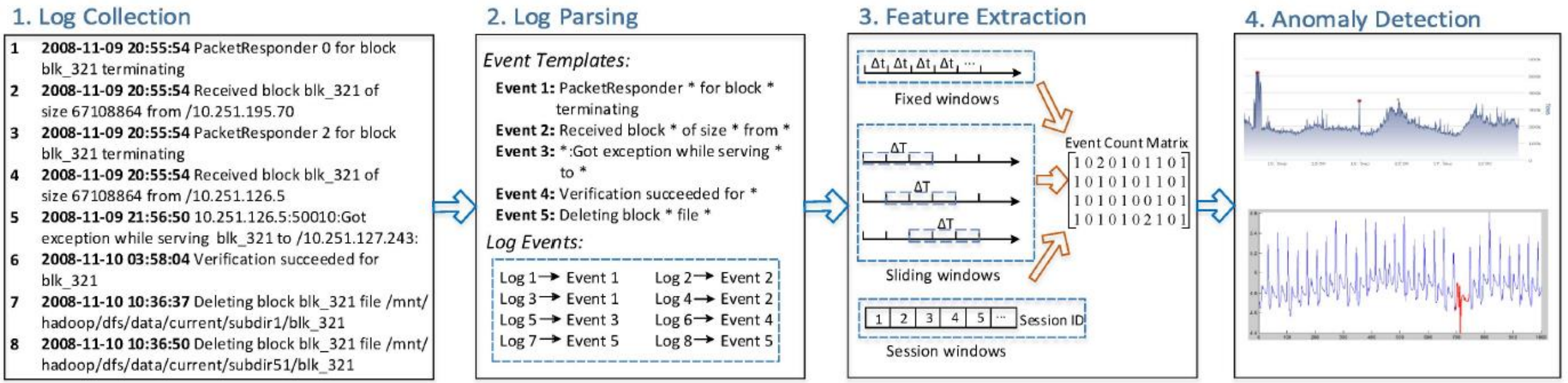
Introduction

- Logs: record system runtime information
- Traditional Log File Anomaly Detection:
  - based on domain knowledge
  - use keywords search or regular expression match
- Challenges
  - 5G network system getting more complex to comprehend
  - 5G network system generate tons of logs
- Intelligent anomaly detection in log file is highly demanded

```
17/06/09 20:10:40 INFO spark.SecurityManager: Changing view acls to: yarn,curi
17/06/09 20:10:40 INFO spark.SecurityManager: Changing modify acls to: yarn,curi
17/06/09 20:10:40 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set
17/06/09 20:10:41 INFO spark.SecurityManager: Changing view acls to: yarn,curi
17/06/09 20:10:41 INFO spark.SecurityManager: Changing modify acls to: yarn,curi
17/06/09 20:10:41 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set
17/06/09 20:10:41 INFO slf4j.Slf4jLogger: Slf4jLogger started
17/06/09 20:10:41 INFO Remoting: Starting remoting
17/06/09 20:10:41 INFO Remoting: Remoting started; listening on addresses :[akka.tcp://sparkExecutorActorSystem@mesos-slave-07:55904]
17/06/09 20:10:41 INFO util.Utils: Successfully started service 'sparkExecutorActorSystem' on port 55904.
17/06/09 20:10:41 INFO storage.DiskBlockManager: Created local directory at /opt/hdfs/nodemanager/usercache/curi/appcache/application_1485
17/06/09 20:10:41 INFO storage.MemoryStore: MemoryStore started with capacity 17.7 GB
17/06/09 20:10:42 INFO executor.CoarseGrainedExecutorBackend: Connecting to driver: spark://CoarseGrainedScheduler@10.10.34.11:48069
17/06/09 20:10:42 INFO executor.CoarseGrainedExecutorBackend: Successfully registered with driver
17/06/09 20:10:42 INFO executor.Executor: Starting executor ID 5 on host mesos-slave-07
17/06/09 20:10:42 INFO util.Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 40984.
17/06/09 20:10:42 INFO netty.NettyBlockTransferService: Server created on 40984
17/06/09 20:10:42 INFO storage.BlockManagerMaster: Trying to register BlockManager
17/06/09 20:10:42 INFO storage.BlockManagerMaster: Registered BlockManager
17/06/09 20:10:45 INFO executor.CoarseGrainedExecutorBackend: Got assigned task 0
17/06/09 20:10:45 INFO executor.CoarseGrainedExecutorBackend: Got assigned task 1
```

Contents

Framework



- **Log collection**: collect logs from large-scale systems
- **Log parsing**: extract a group of event templates to make the log structured
- **Feature extraction**: encode the parsed logs into numerical feature matrix
- **Anomaly detection**: the feature matrix can be fed into machine learning models for training to generate a model for anomaly detection

Shilin He, et al. Experience Report: System Log Analysis for Anomaly Detection, 2016

Contents

Methodology - Log Parsing

Log Parsing

- Constant parts
- variable parts
- Example:
  - Connection from 10.10.34.12 closed
  - Connection from 10.10.34.13 closed
  - Connection from * closed

Methodology - Log Parsing

- **Drain :** an online log parser in a streaming manner

  Step1: Preprocess by domain knowledge

  Step2: Search by log message length

  Step3: Search by preceding tokens

  Step4: Search by token similarity

  Step5: Update the parse tree

$$simSeq = \frac{\sum_{i=1}^{n} equ(seq_1(i), seq_2(i))}{n},$$

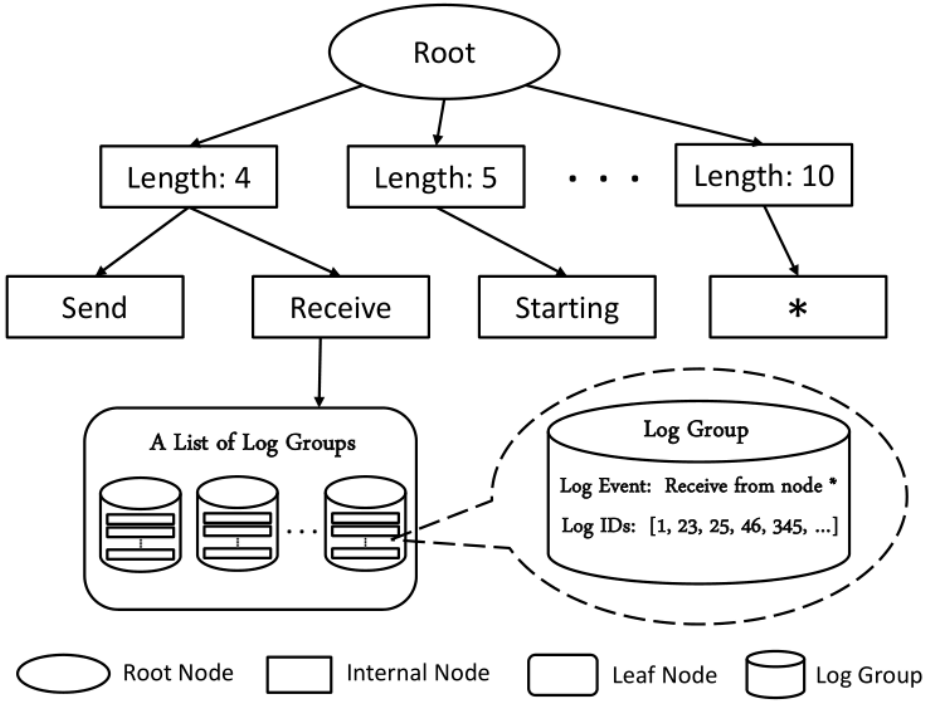


Fig. 2: Structure of Parse Tree in Drain ($depth$ = 3)

Pinjia He, et al. Drain: An Online Log Parsing Approach with Fixed Depth Tree, 2017.

Methodology - Log Parsing

TABLE IV: Different messages belonging to message type "SIF" in Table II, and the words ordered according to L

Message No.	Detailed Message	Words ordered according to L
M_1	Interface ae3, changed state to down	"changed", "state", "to", "Interface", "down", "ae3"
M_2	Vlan-interface vlan22, changed state to down	"changed", "state", "to", "Vlan-interface", "down", "vlan22"
M_3	Interface ae3, changed state to up	"changed", "state", "to", "Interface", "up", "ae3"
M_4	Vlan-interface vlan22, changed state to up	"changed", "state", "to", "Vlan-interface", "up", "vlan22"
M_5	Interface ae1, changed state to down	"changed", "state", "to", "Interface", "down", "ae1"
M_6	Vlan-interface vlan20, changed state to down	"changed", "state", "to", "Vlan-interface", "down", "vlan20"
M_7	Interface ae1, changed state to up	"changed", "state", "to", "Interface", "up", "ae1"
M_8	Vlan-interface vlan20, changed state to up	"changed", "state", "to", "Vlan-interface", "up", "vlan20"

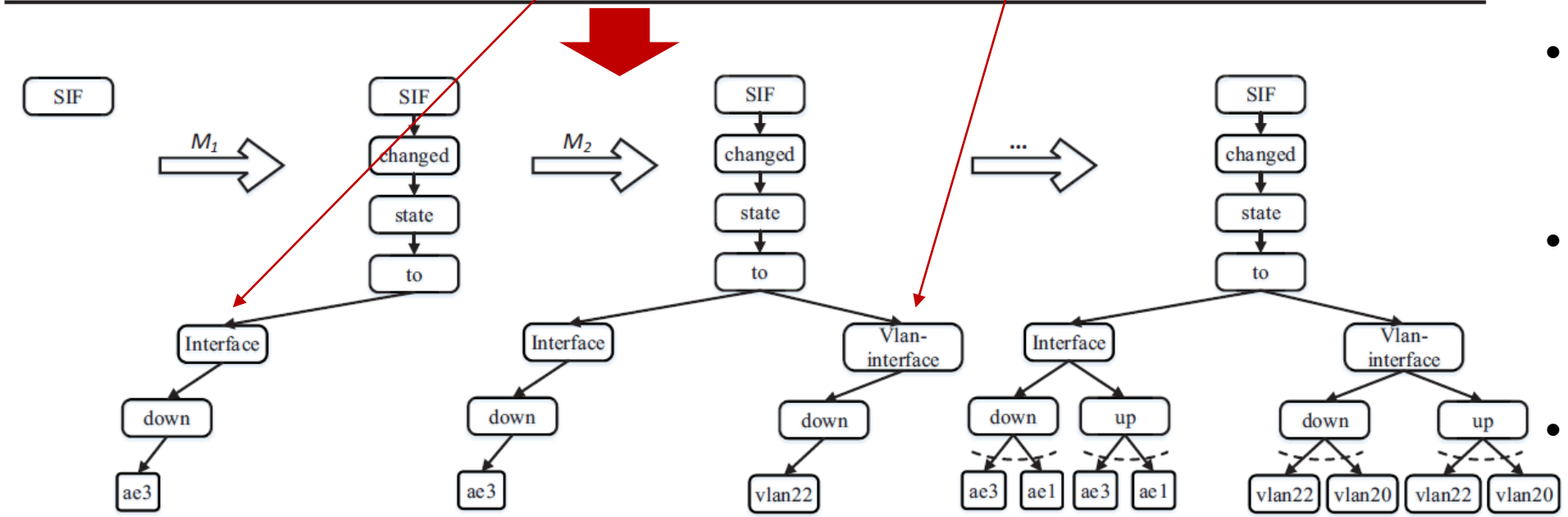


Fig. 3: Example of constructing an FT-tree

FT-tree

- Arrange each line of log words in descending order according to word frequency
- Insert descending word sequence into a tree structure
- prune the tree, remove the node which has too many children
- support incremental Template Learning

Shenglin Zhang, et al. Syslog Processing for Switch Failure Diagnosis and Prediction in Datacenter Networks, 2017.

Methodology - Feature Extraction

Fixed Windows

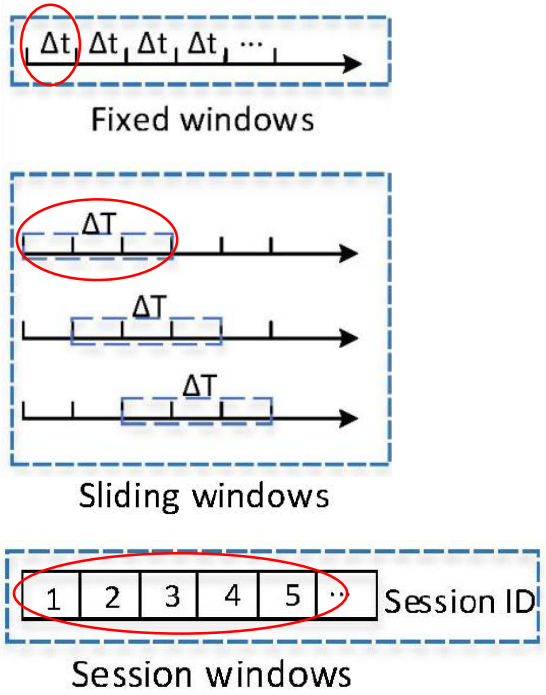
- based on timestamp
- window size: time duration (delta t)
- window number depends on the predefined window size

Sliding Windows

- based on timestamp
- window size & step size (e.g. hourly window sliding every five minutes)

Session Windows

- based on identifier
- e.g. HDFS logs with block_id record the allocation, writing, replication, deletion of certain block



Event Count Vector
[1, 0, 2, 0, 1, 0, 1, 1, 0, 1]

Event Count Matrix

$$\begin{bmatrix} 1 & 0 & 2 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 2 & 1 & 0 & 1 \end{bmatrix}$$

Methodology – Anomaly Detection

Step 1: convert event counts to chi squared stats

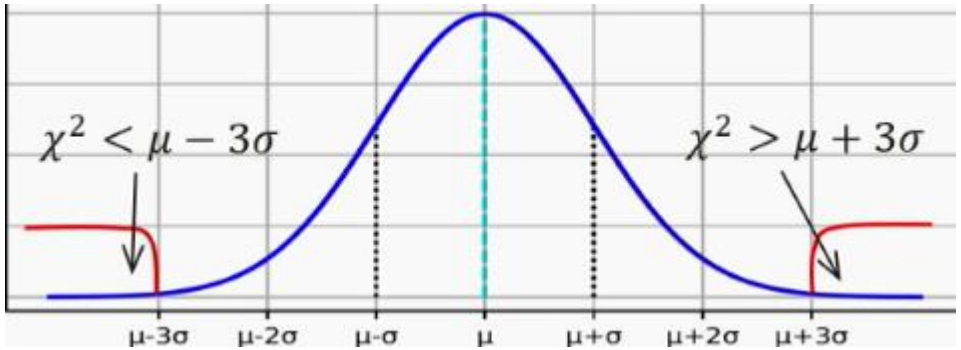
(Bayesian Likelihood, TFIDF Chi2, Auto-Encoder)

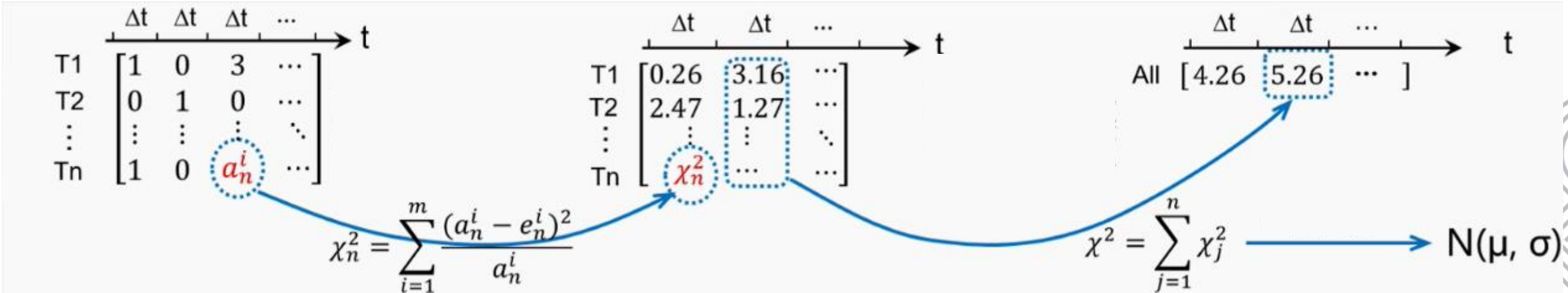
Step 2: sum the chi squared stats

Step 3: detect sequence anomalies

- Static Threshold, Time Series Models

3-sigma





Contents

Challenge Statement

Datasets

- trainsets: log files without anomalies
- testsets: log files with anomalies

Evaluation

- F1 score
- code efficiency

Submission

- concat_answer.csv
- source code

	A	B	C
1	LogName	TimeSlice	Label
2	messages	5399328	0
3	messages	5399329	1
4	messages	5399330	0
5	messages	5399331	0
6	messages	5399332	0
7	messages	5399333	0
8	messages	5399334	0
9	messages	5399335	0
10	messages	5399336	0
11	messages	5399337	0

Trick: the logs contain the keywords like “failed”, ” error”, “warn” do not necessary represent system anomalies.

China unicom中国联通
创新·改变世界

BEIJING 2022

北京2022年冬奥会官方合作伙伴
Official Partner of the Olympic Winter Games Beijing 2022

Thank You