DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning

Min Du, Feifei Li, Guineng Zheng, Vivek Srikumar

CCS '17 Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security

May 14, 2018

Agenda

Anomaly Detection

- Systems and applications more complex than before
- More bugs and vulnerabilities
- Attacks get increasingly sophisticated
- Many traditional anomaly detection methods based on standard mining methodologies are no longer effective.

Why are system logs good material for anomaly detection

- Record system states and significant events at various critical points
- Universally available in nearly all computer systems
- Record noteworthy events as they occur from actively running process
- Natural language, easy to process

System logs

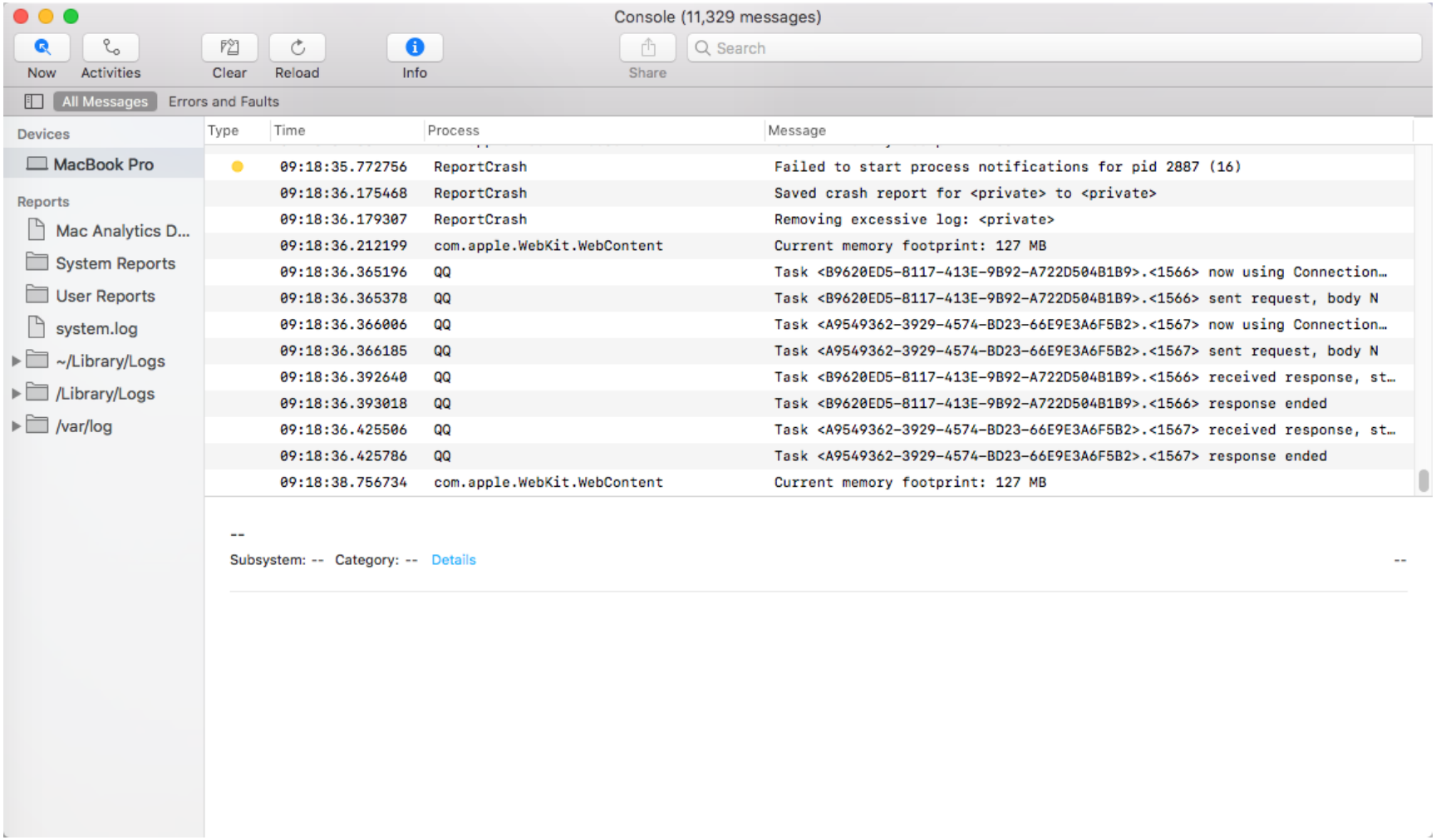


Figure: MacOS

How to use system logs?

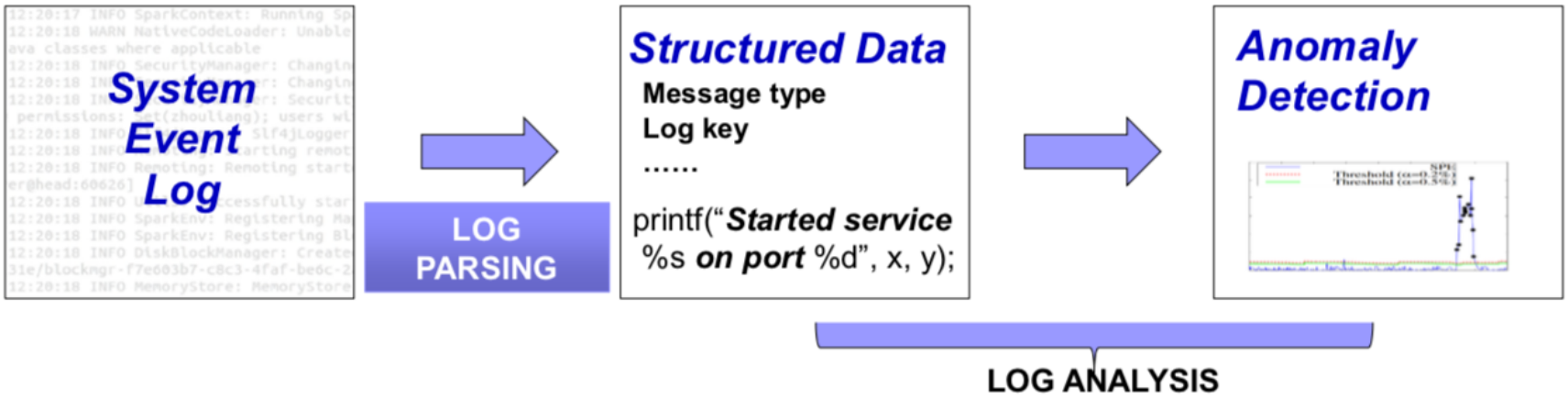


Figure: Process

Detail of Log parser

DeepLog

log message (log key underlined)	log key	parameter value vector
t_1 Deletion of file1 complete	k_1	[t_1 - t_0, file1]
t_2 Took 0.61 seconds to deallocate network …	k_2	[t_2 - t_1, 0.61]
t_3 VM Stopped (Lifecycle Event)	k_3	[t_3 - t_2]
…	…	…

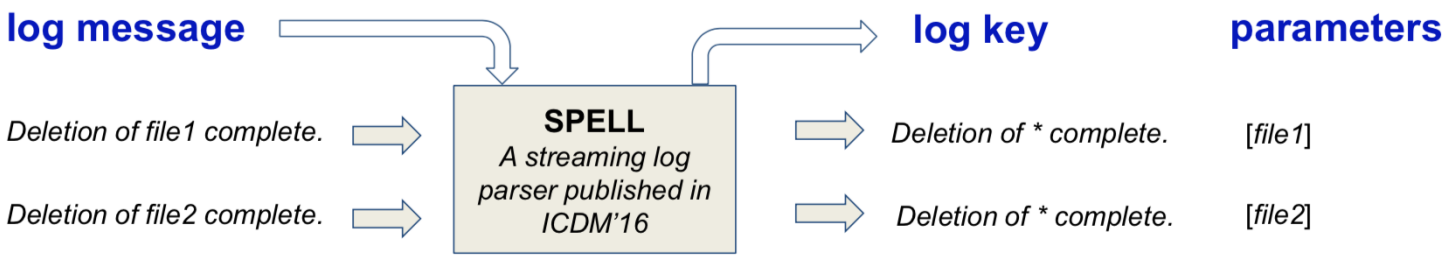


Figure: Log parser

Existing approaches

- PCA based approaches over log message counters
- Invariant mining based methods to capture co-occurrence patterns between different log keys
- TFIDF(LTSM neural network - Binary classifier)
- Workflow based methods to identify execution anomalies in program logic flows(CloudSeer-only work on OpenStack logs)

Advantage: successful in certain scenarios

Existing approaches

Disadvantage :

- None of them effective as universal anomaly detection method
- None of them is in an online fashion
- Only focus on log keys

Challenges

- Log data are unstructured and vary from systems
- Anomaly detection has to be timely in order to be useful
- Task concurrency
- Rich information in log message

Contribution

- LSTM neural network
- Log keys and ALSO metric values
- Separate log entries of concurrent task and help build workflow for diagnose

DeepLog

log message (log key underlined)	log key	parameter value vector
t_1 Deletion of file1 complete	k_1	[t_1 - t_0, file1]
t_2 Took 0.61 seconds to deallocate network …	k_2	[t_2 - t_1, 0.61]
t_3 VM Stopped (Lifecycle Event)	k_3	[t_3 - t_2]
…	…	…

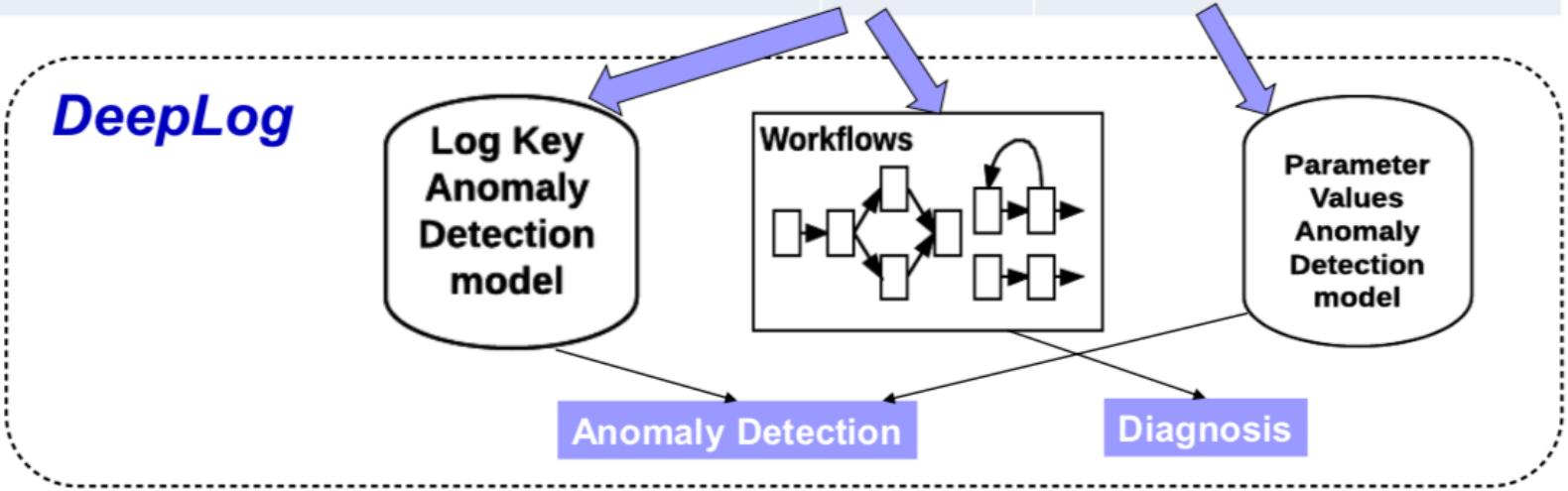
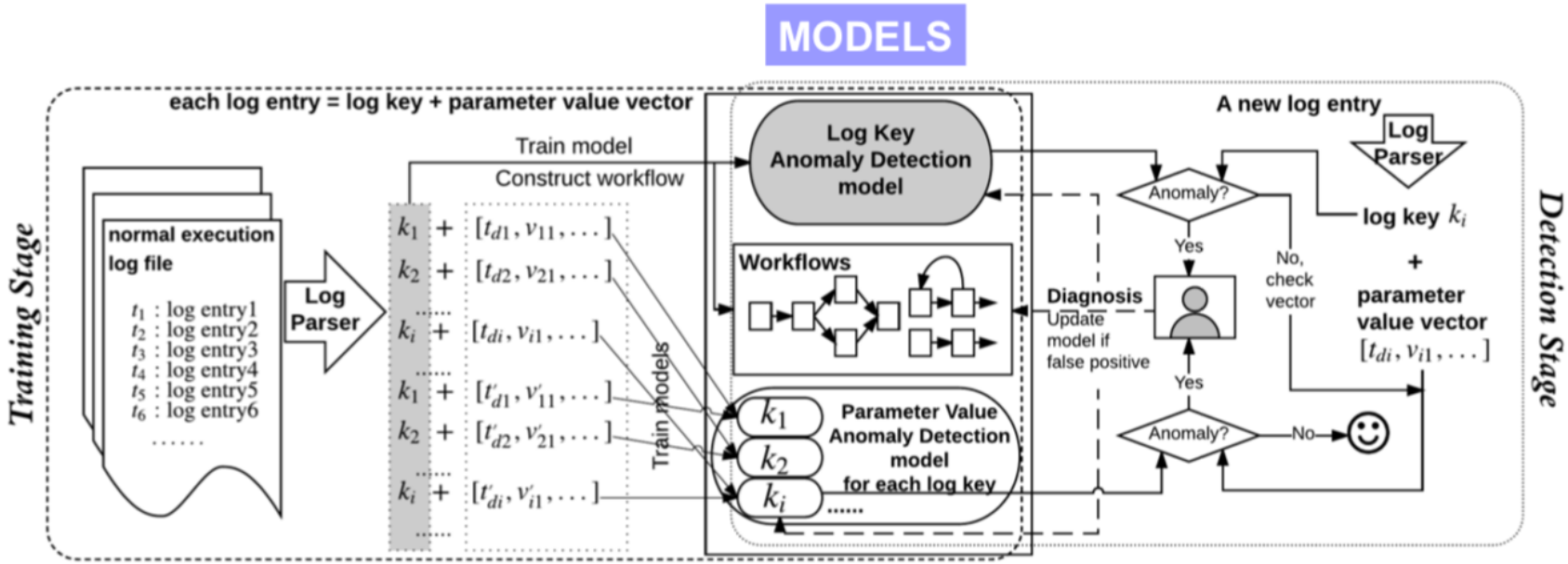


Figure: architecture

DeepLog



46

Figure: whole view

DeepLog

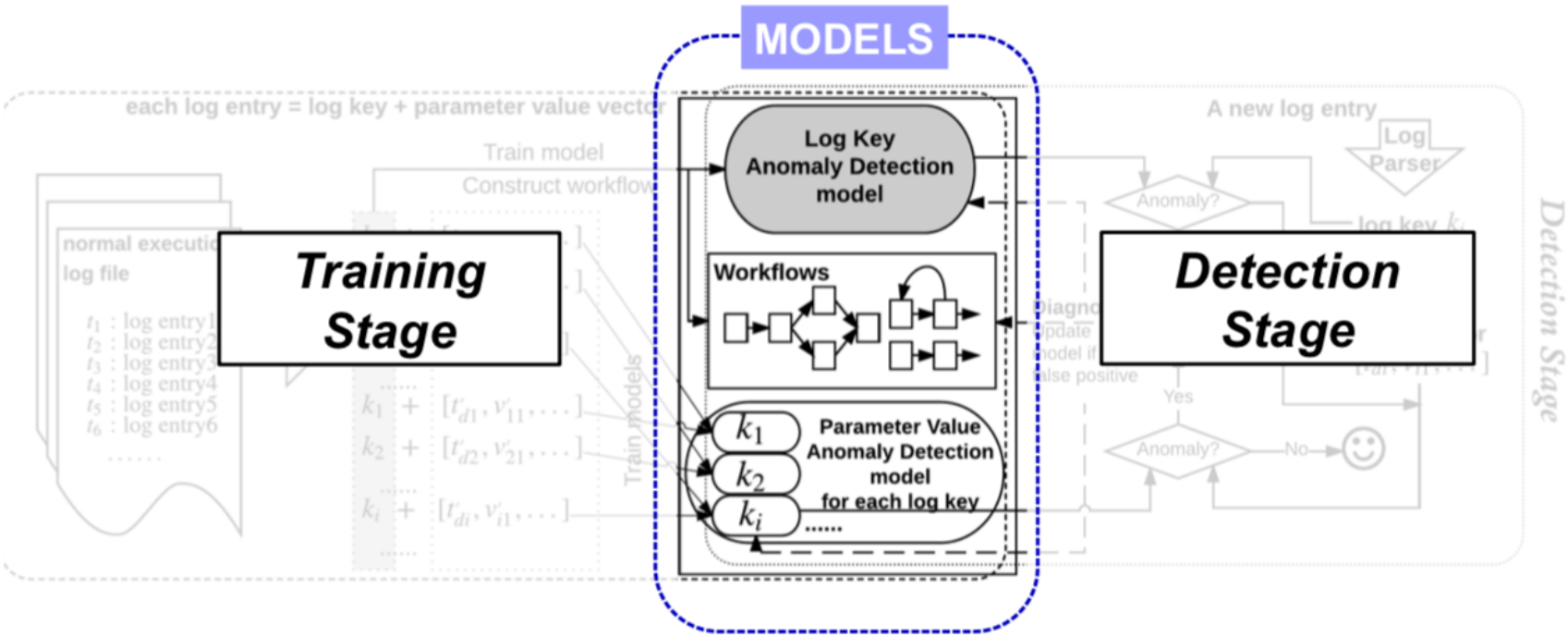


Figure: Models

DeepLog

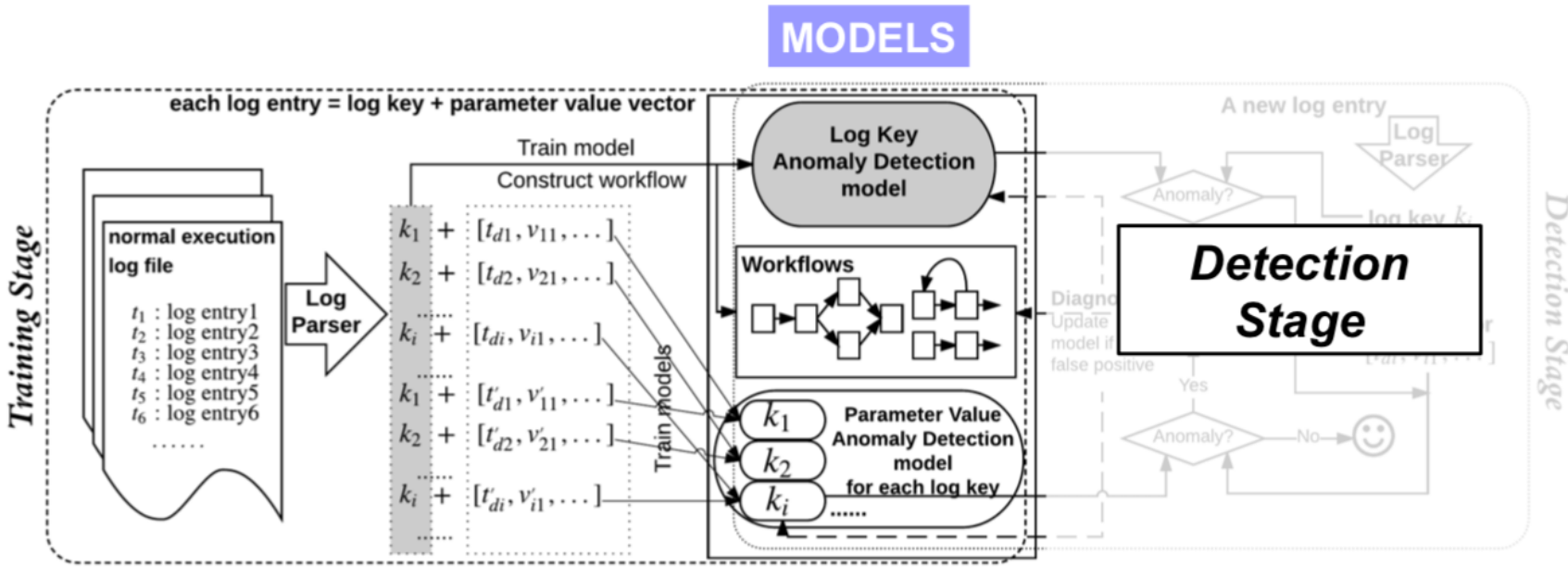
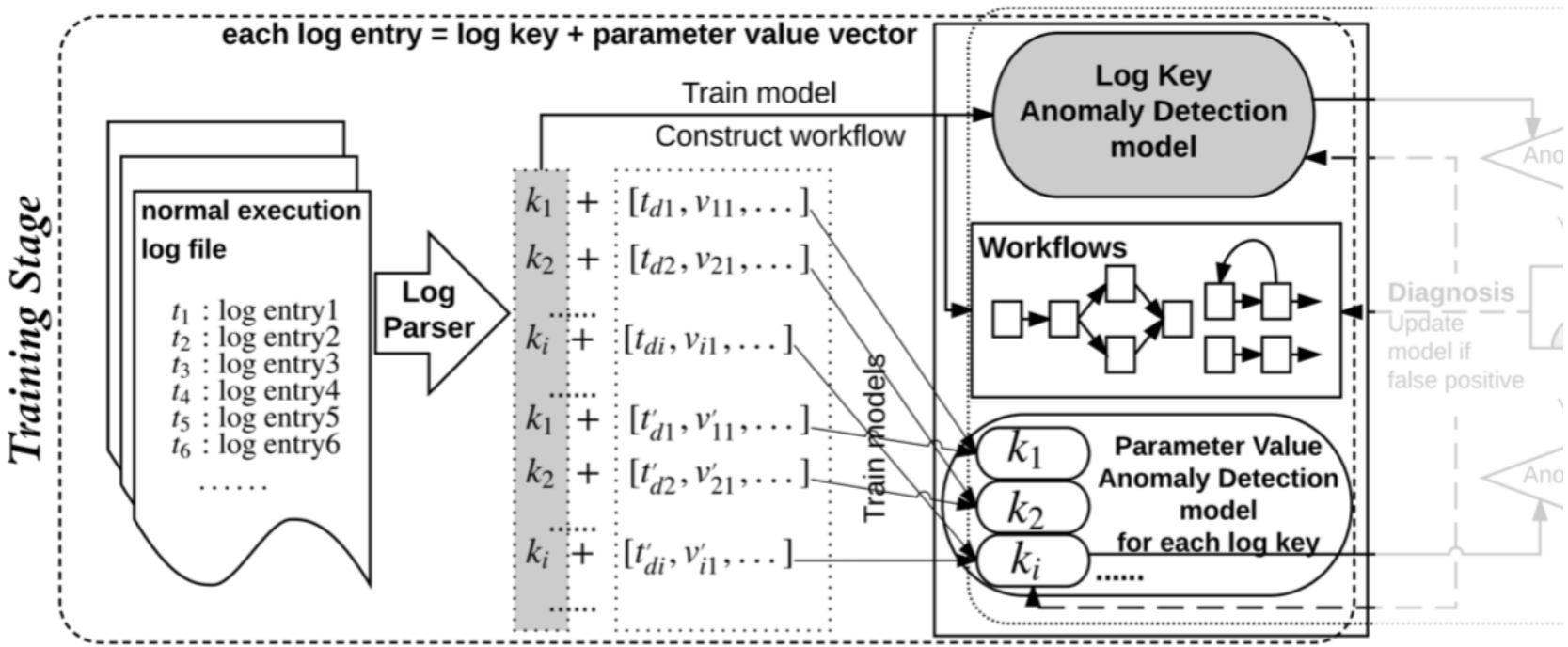


Figure: Training Stage

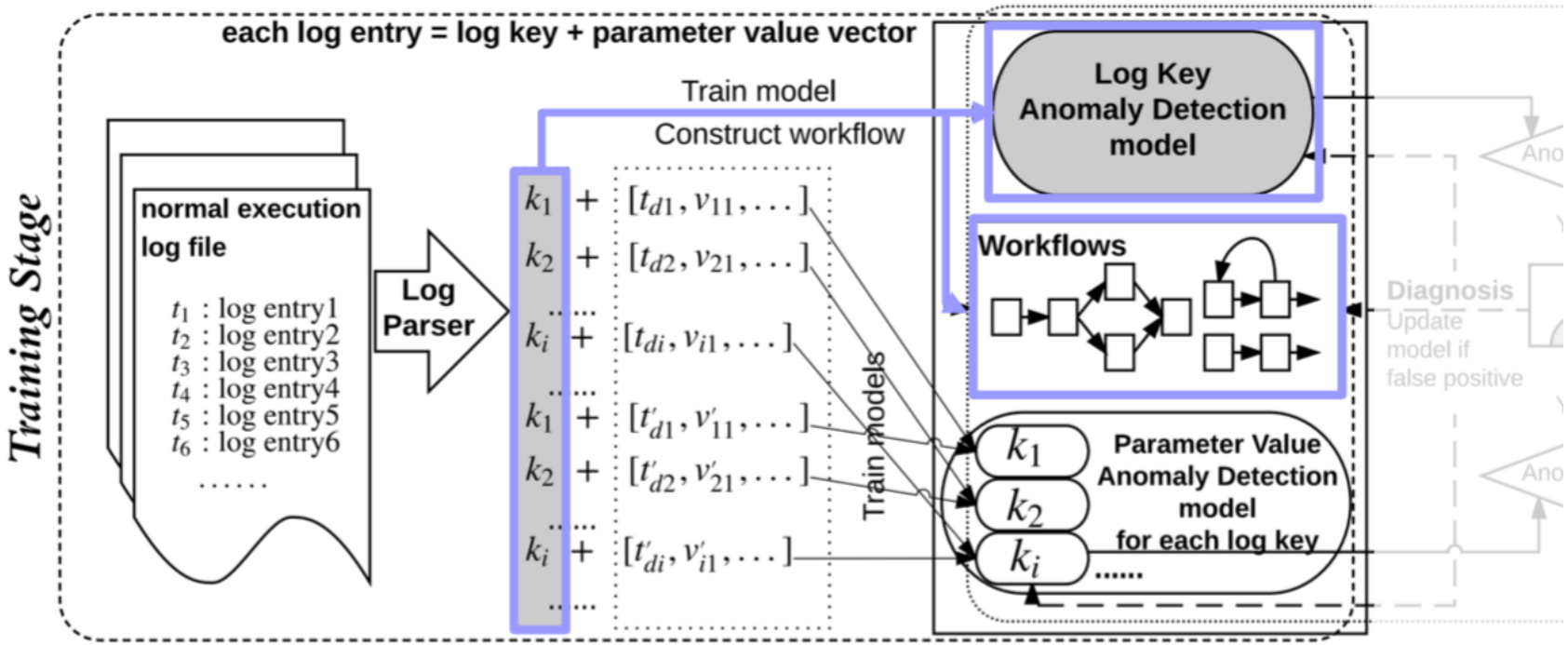
DeepLog



31

Figure: Training Stage

DeepLog



34

Figure: Training Stage

DeepLog

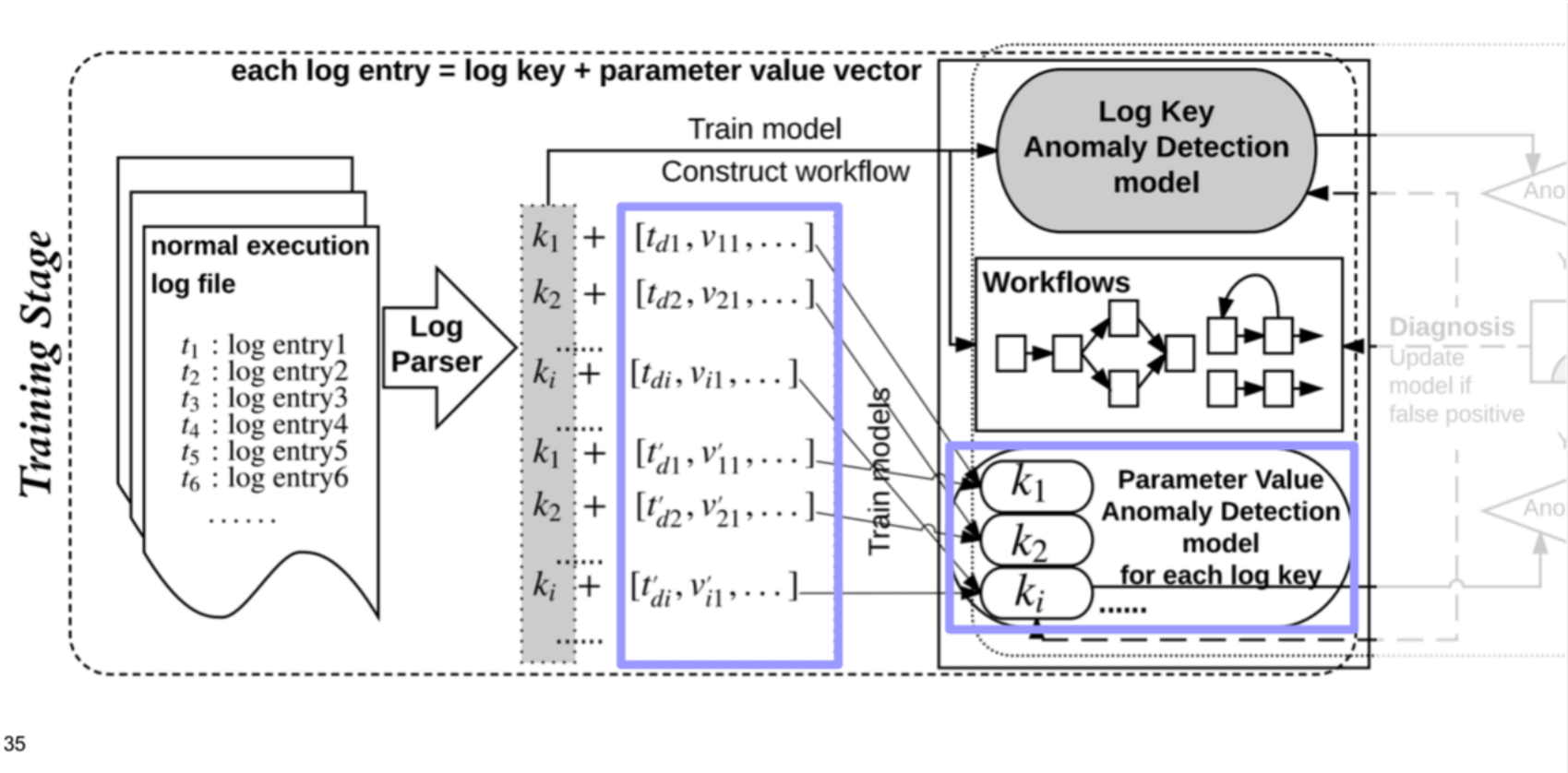


Figure: Training Stage

DeepLog

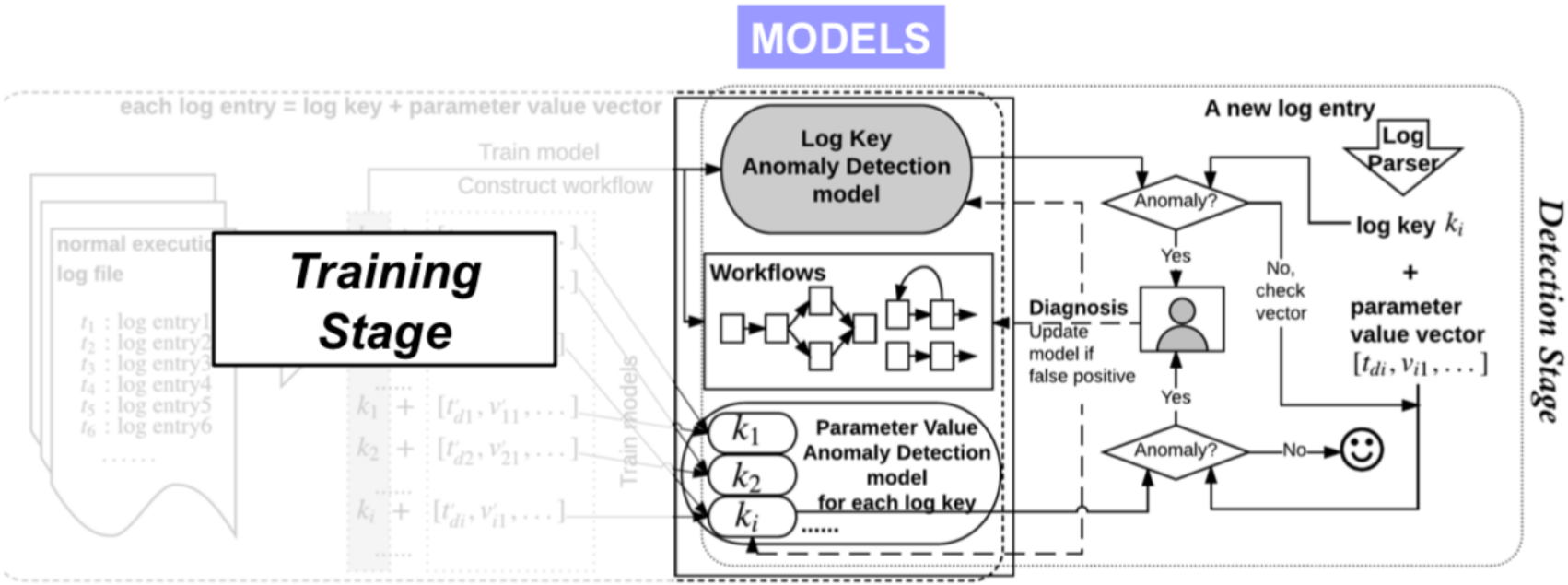
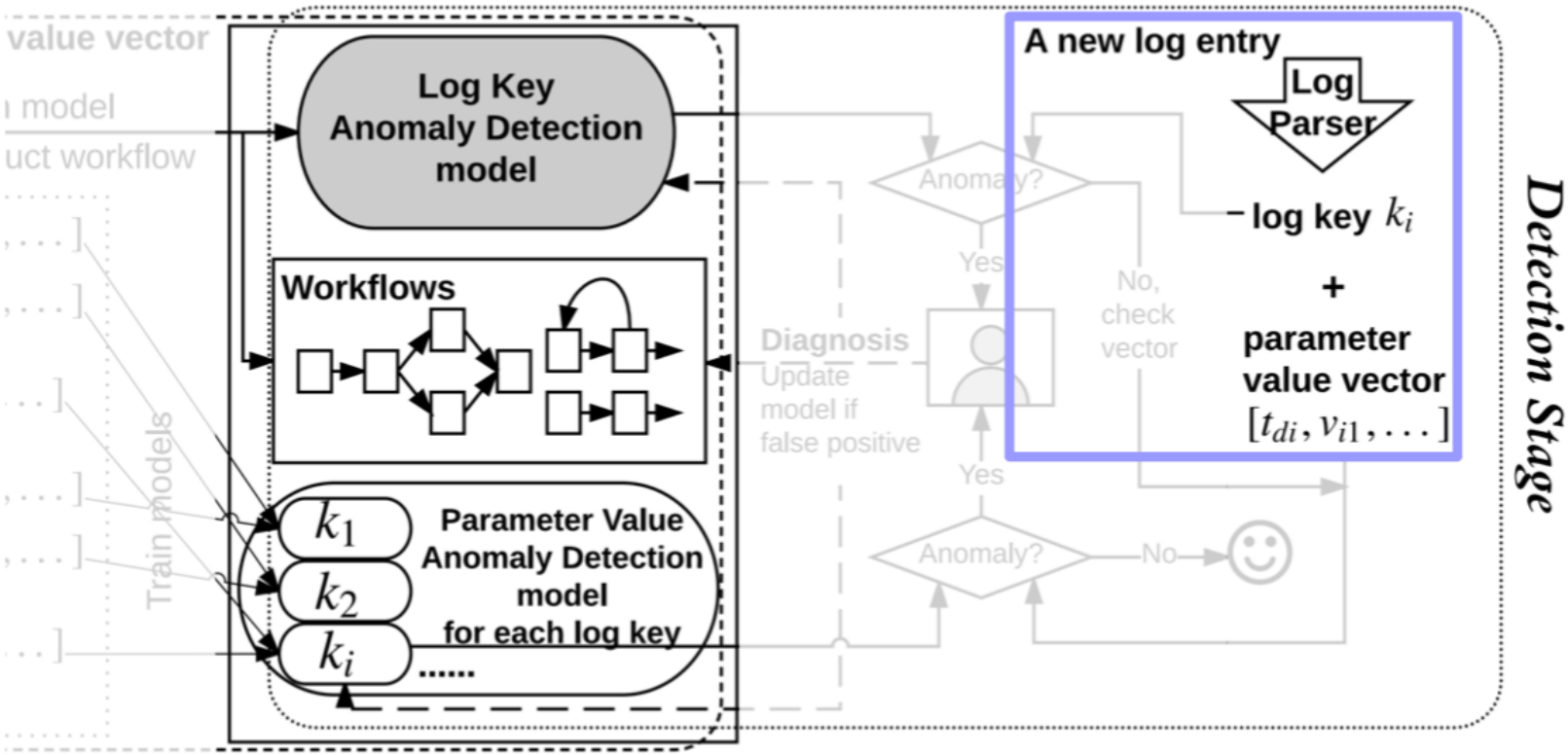


Figure: Detection Stage

DeepLog



39

Figure: Detection Stage

DeepLog

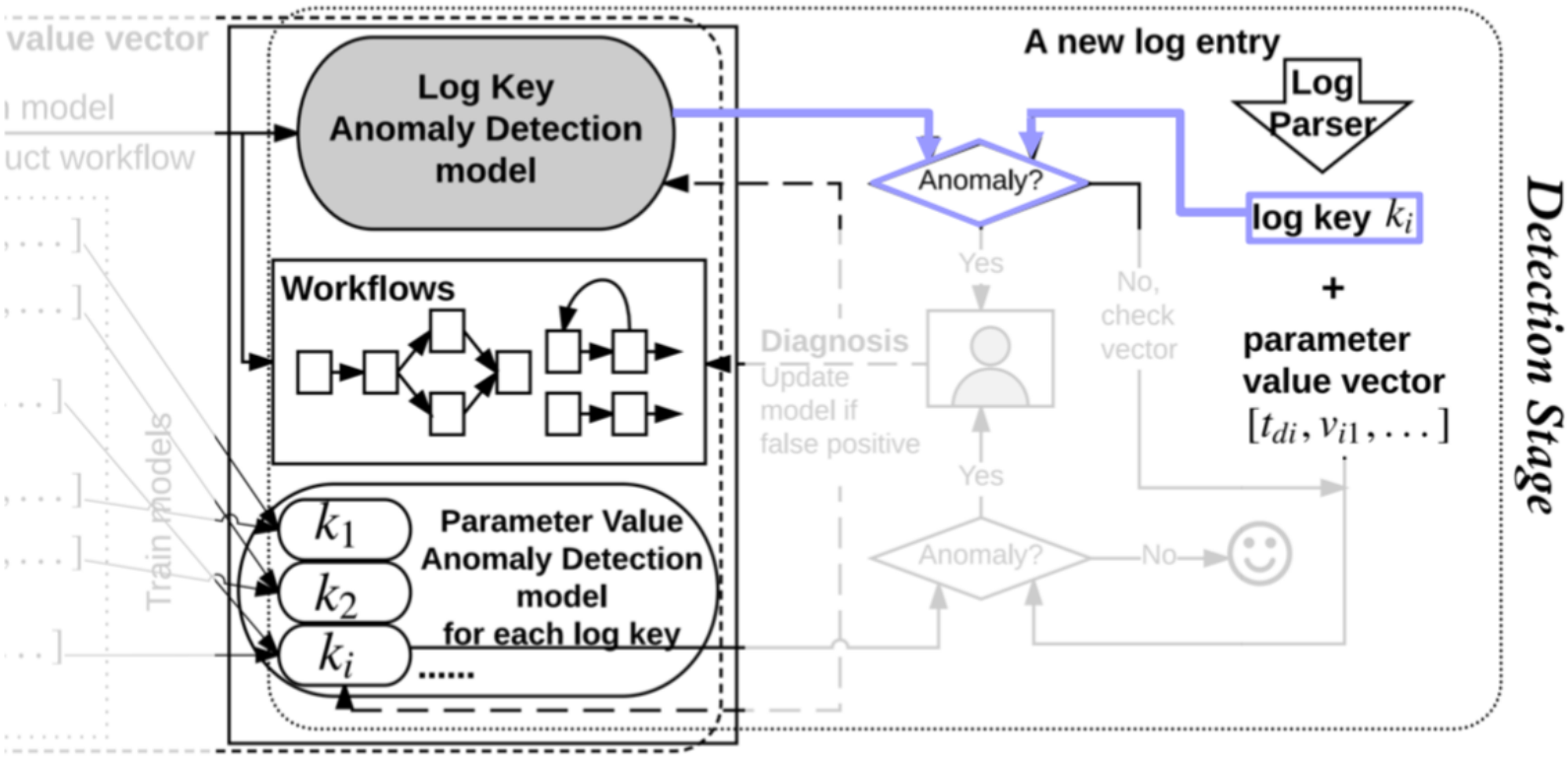
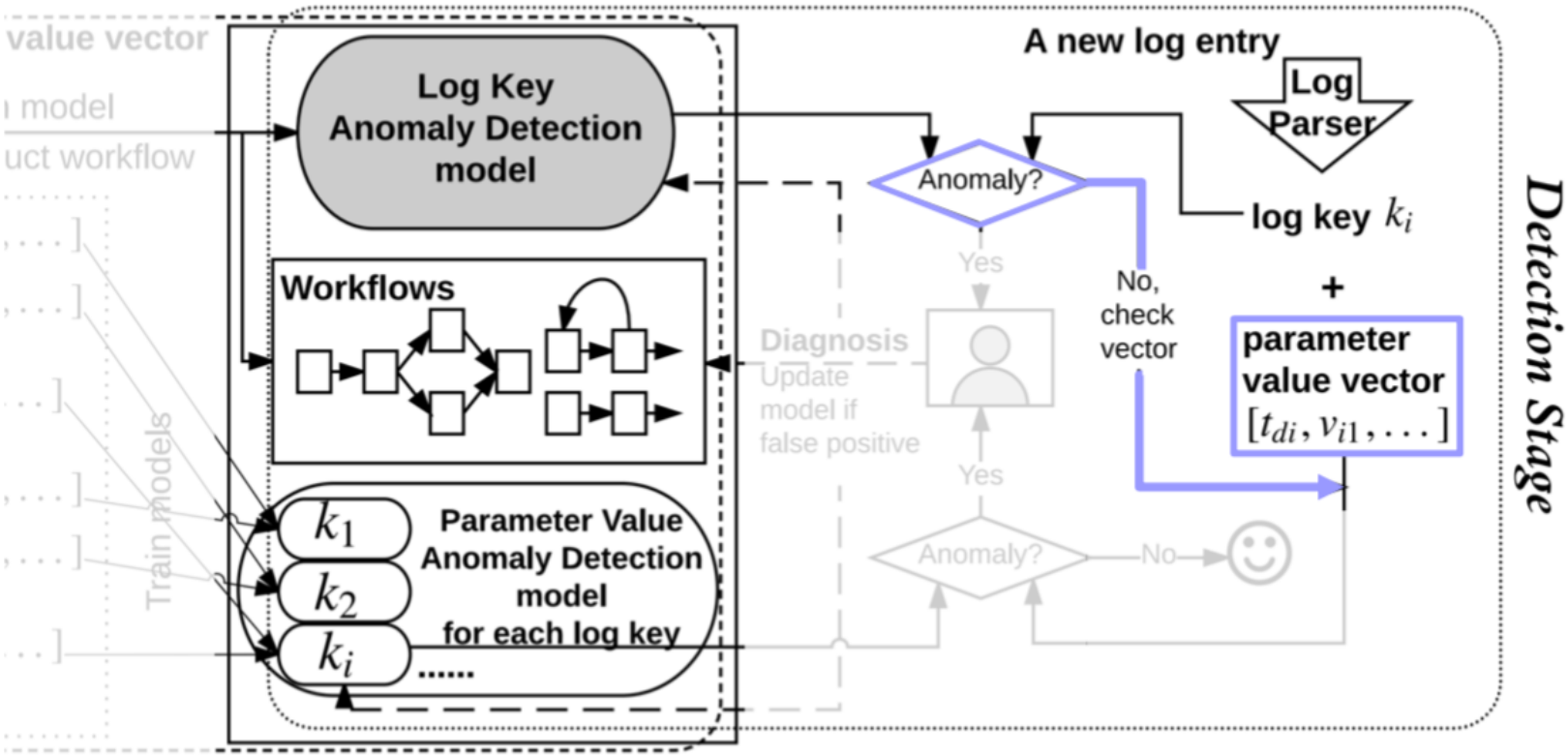


Figure: Detection Stage

DeepLog



11

Figure: Detection Stage

DeepLog

DeepLog Architecture

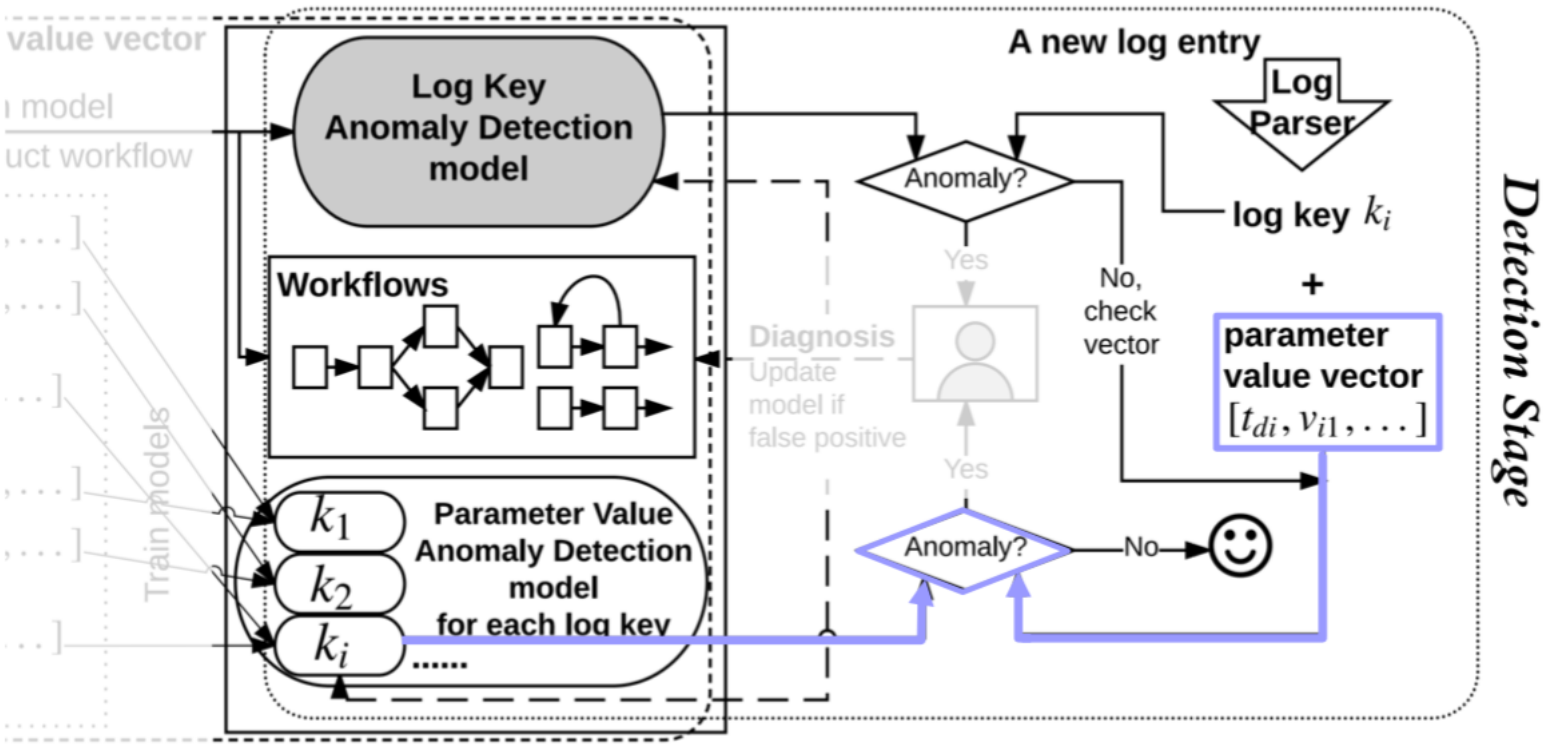


Figure: Detection Stage

DeepLog

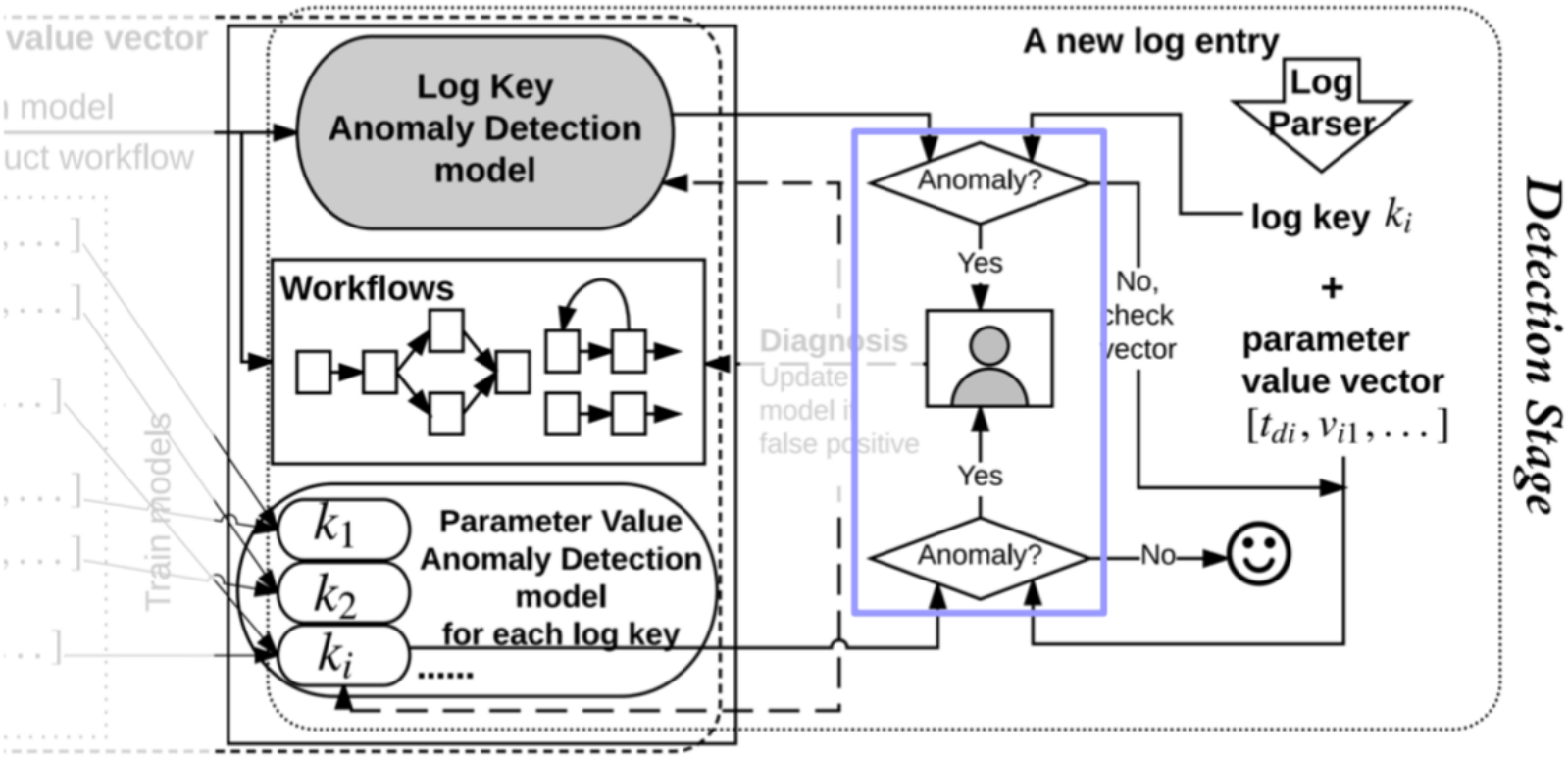


Figure: Detection Stage

DeepLog

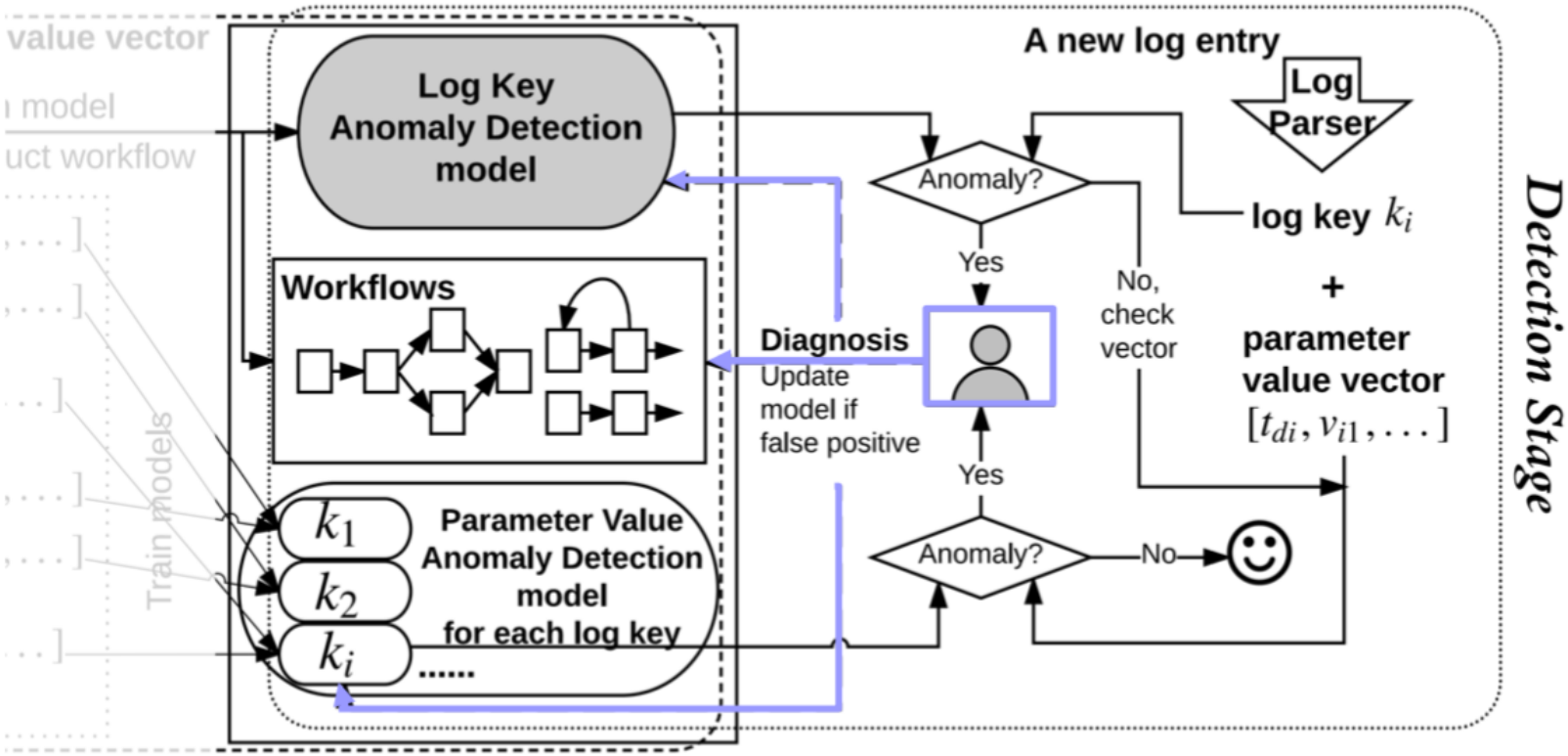
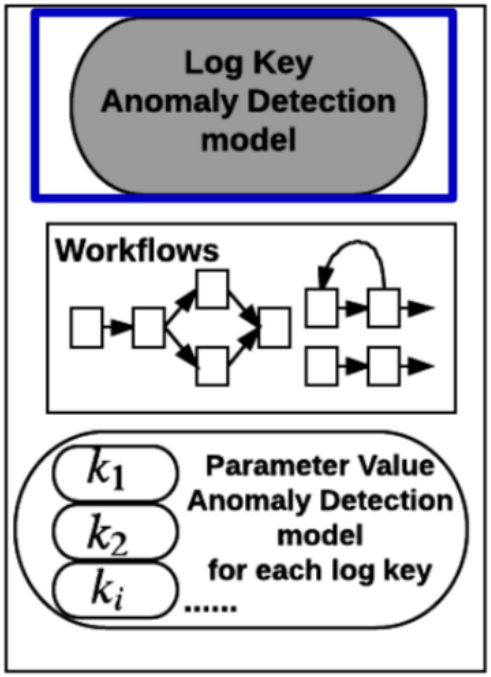
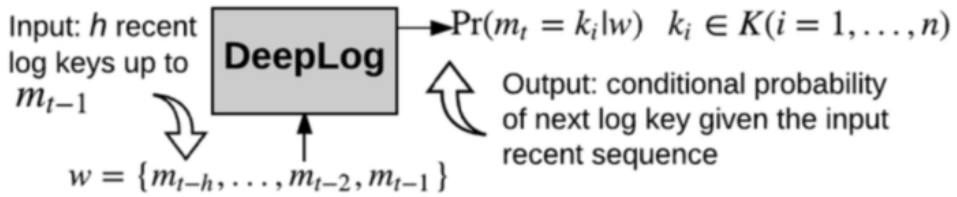


Figure: update online

DeepLog



Use long short-term memory (LSTM) architecture

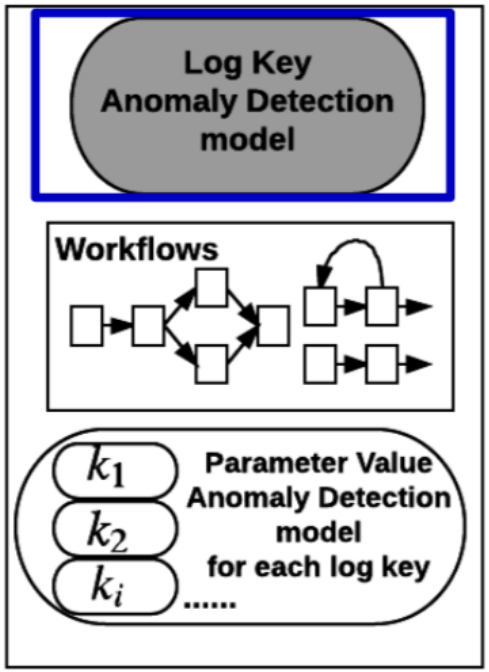


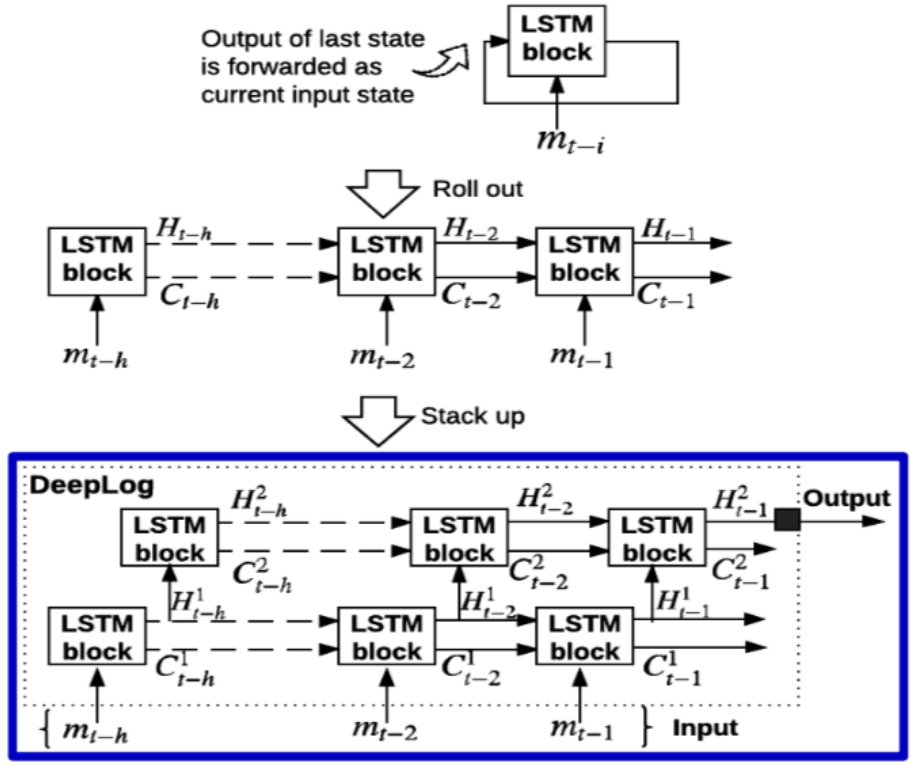
Training:

log key sequence:

h=3 25 18 54 57 18 56 … 25 18 54 57 56 18 …

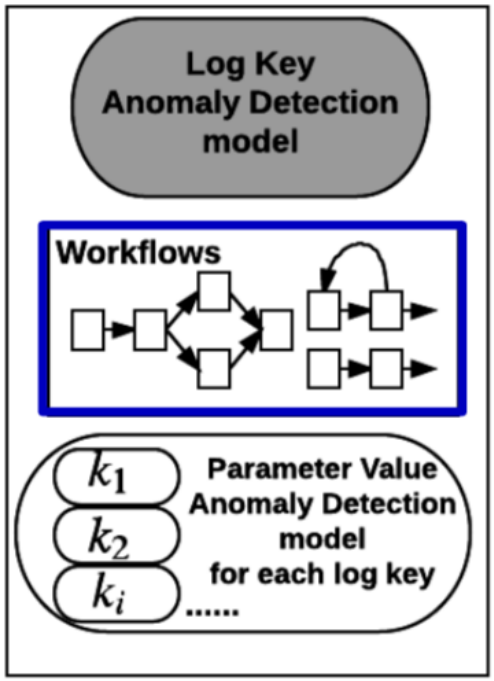
DeepLog


Log Key
Anomaly Detection
model
Workflows
k_1
k_2
k_i
.....
Parameter Value
Anomaly Detection
model
for each log key


Output of last state
is forwarded as
current input state
LSTM
block
m_{t-i}
Roll out
H_{t-h}
H_{t-2}
H_{t-1}
C_{t-h}
C_{t-2}
C_{t-1}
m_{t-h}
m_{t-2}
m_{t-1}
Stack up
DeepLog
H^2_{t-h}
H^2_{t-2}
H^2_{t-1}
Output
C^2_{t-h}
C^2_{t-2}
C^2_{t-1}
H^1_{t-h}
H^1_{t-2}
H^1_{t-1}
C^1_{t-h}
C^1_{t-2}
C^1_{t-1}
Input

59

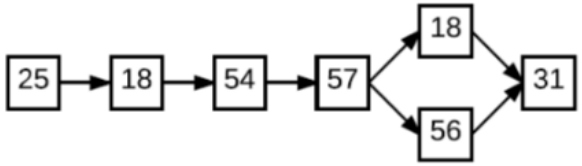
DeepLog



Input: log key sequence

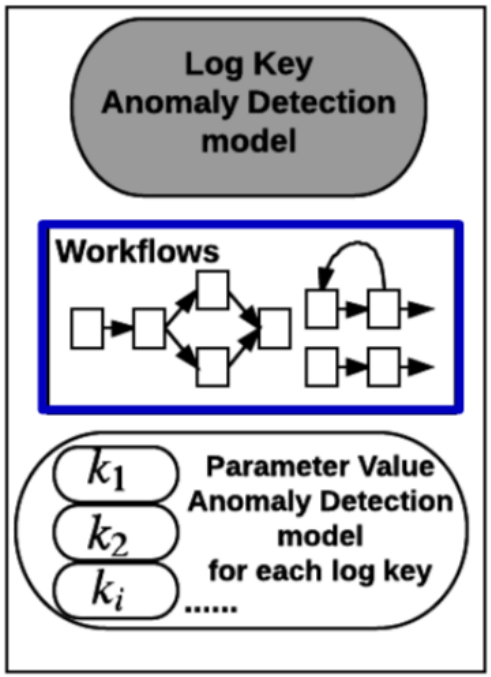
25 18 54 57 18 56 … 25 18 54 57 56 18 …

Output:



60

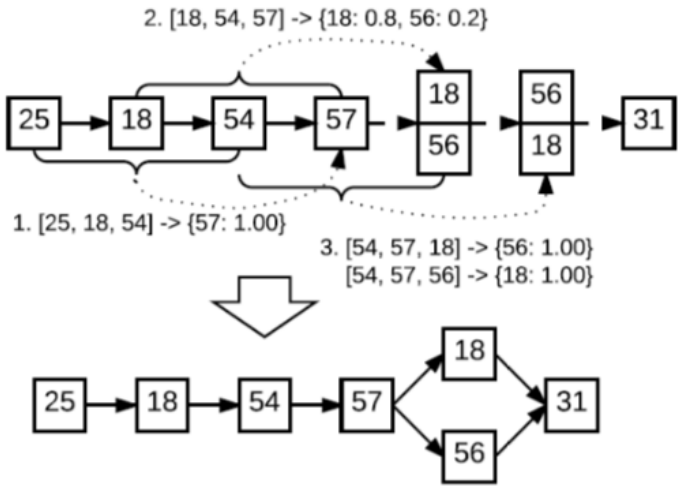
DeepLog



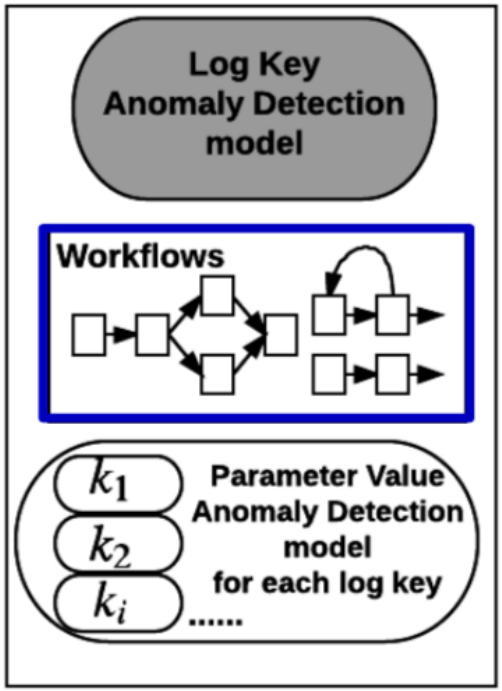
Method 1: Using Log Key Anomaly Detection model

--- LSTM prediction probabilities

An example of concurrency detection:



DeepLog



Method 2: A density-based clustering approach

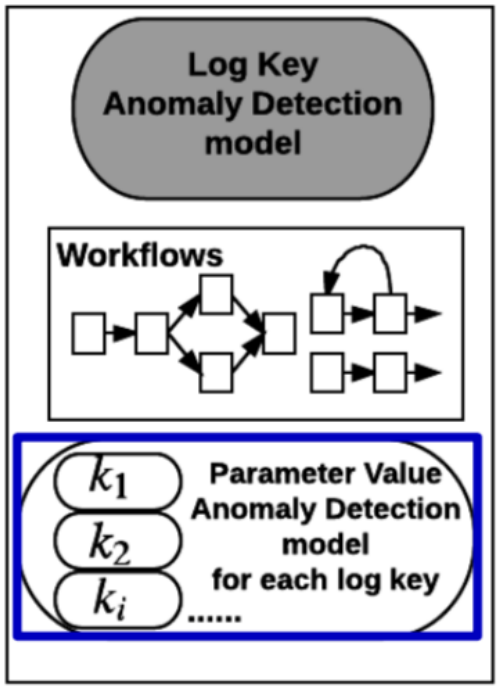
Co-occurrence matrix of log keys (k_i, k_j) within distance d

	k_1	...	k_j	...	k_n
k_1	$p_d(1,1)$		$p_d(1,j)$		
...					
k_i	$p_d(i,1)$		$p_d(i,j) = \frac{f_d(k_i,k_j)}{d \cdot f(k_i)}$		
...					
k_n	$p_d(n,1)$		$p_d(n,j)$		

$f_d(k_i, k_j)$: the frequency of (k_i, k_j) appearing together within distance d
$f(k_i)$: the frequency of k_i in the input sequence
$p_d(i,j)$: the probability of (k_i, k_j) appearing together within distance d

DeepLog

Parameter Value Anomaly Detection model



Example:

Log messages of a particular log key:

t_2: *Took* 0.61 *seconds to deallocate network* ...

t'_2: *Took* 1.1 *seconds to deallocate network* ...

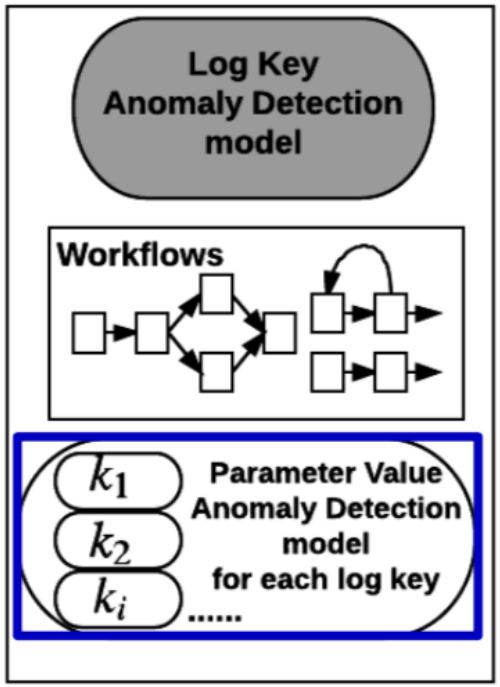
....

Parameter value vectors overtime:

$[t_2 - t_1, 0.61]$, $[t'_2 - t'_1, 1.1]$, ...

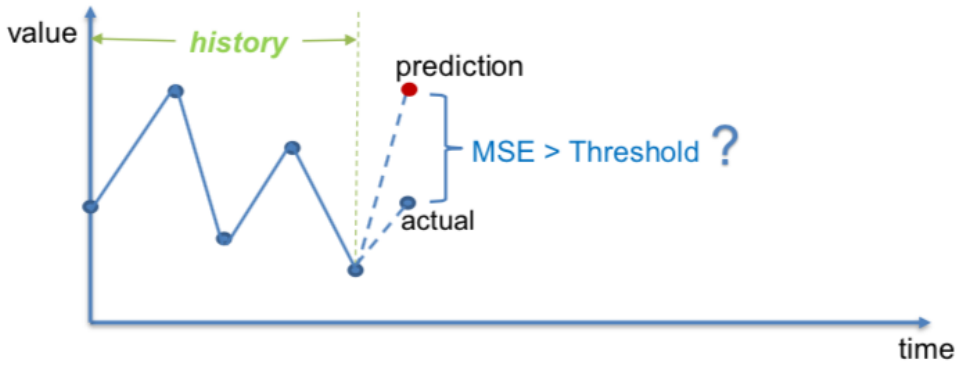
Multi-variate time series data anomaly detection problem!

DeepLog



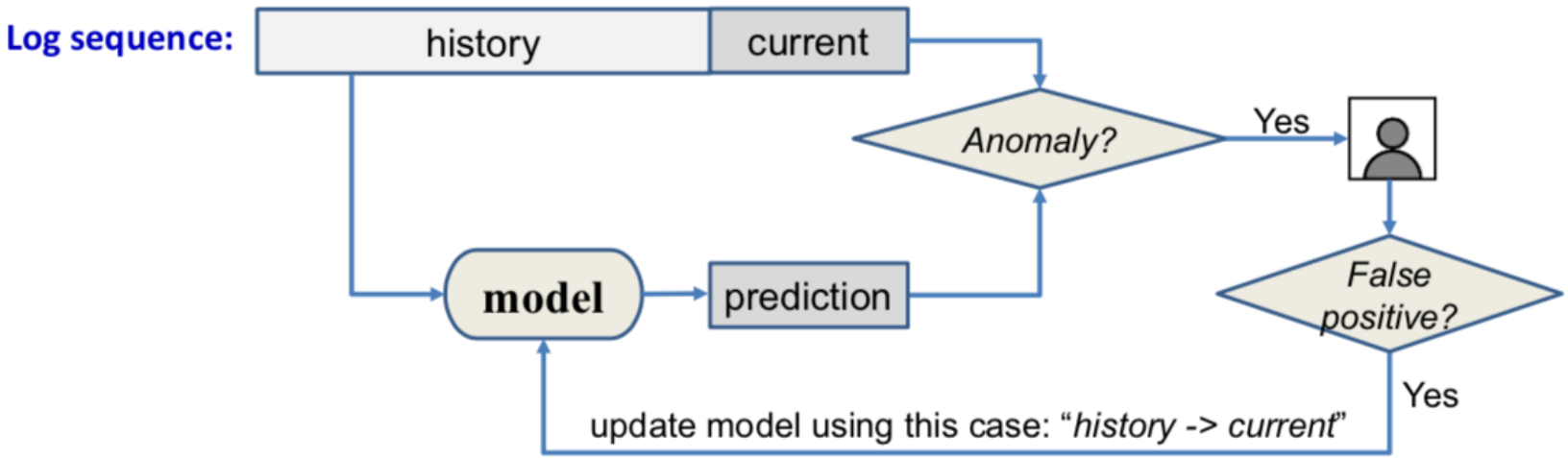
Multi-variate time series data anomaly detection problem

- ✓ Leverage LSTM-based approach;
- ✓ A parameter value vector is given as input at each time step;
- ✓ An anomaly is detected if the mean-square-error (MSE) between prediction and actual data is too big.

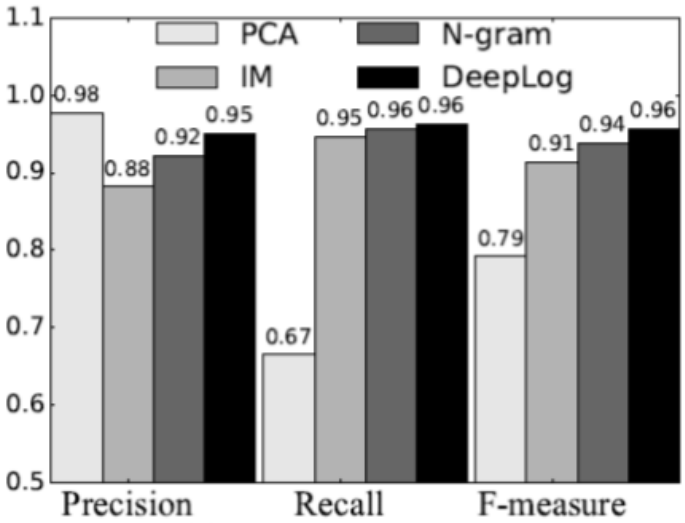


DeepLog Online Update

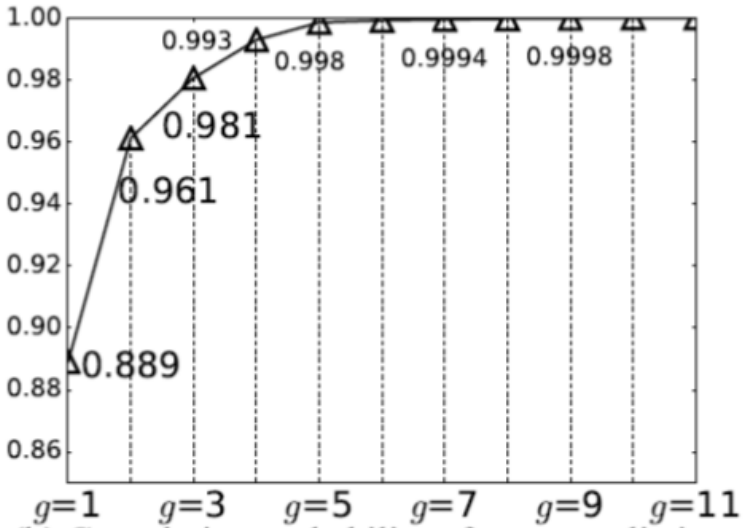
Q: How to handle false positive?



Evaluation log key anomaly detection



(a) Accuracy on HDFS.



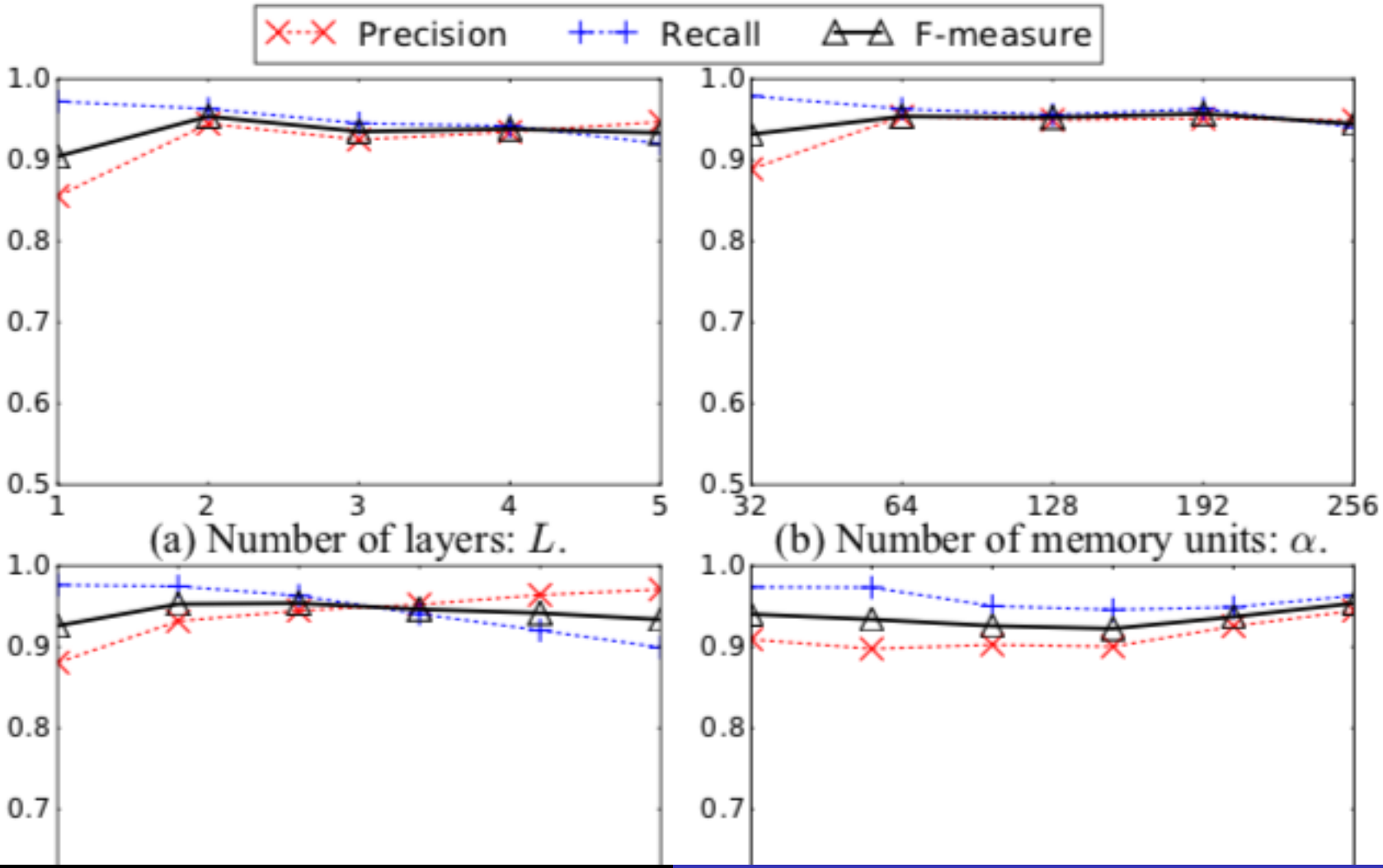
(b) Cumulative probability of top g predictions.

Evaluation results on HDFS log data [1].

(over a million log entries with labeled anomalies)

[1] PCA (SOSP'09), IM (UsenixATC'10), N-gram (baseline language model)

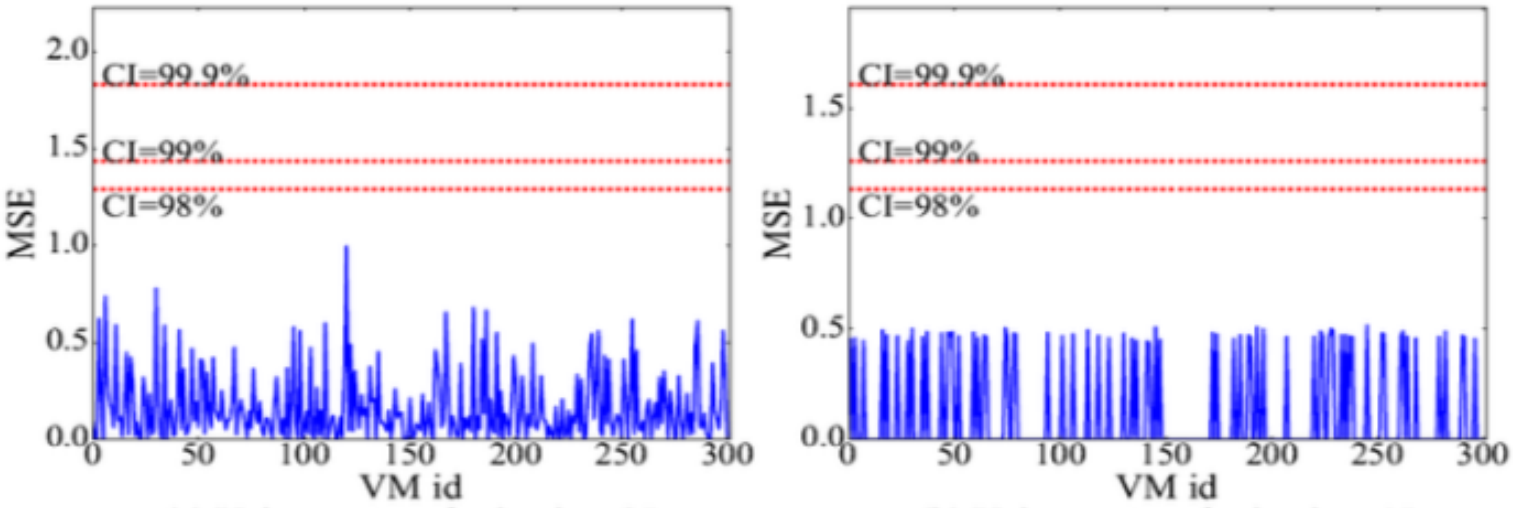
Evaluation log key anomaly detection



(a) Number of layers: L.

(b) Number of memory units: α.

Evaluation parameter value anomaly detection



(a) Value vectors for log key 25

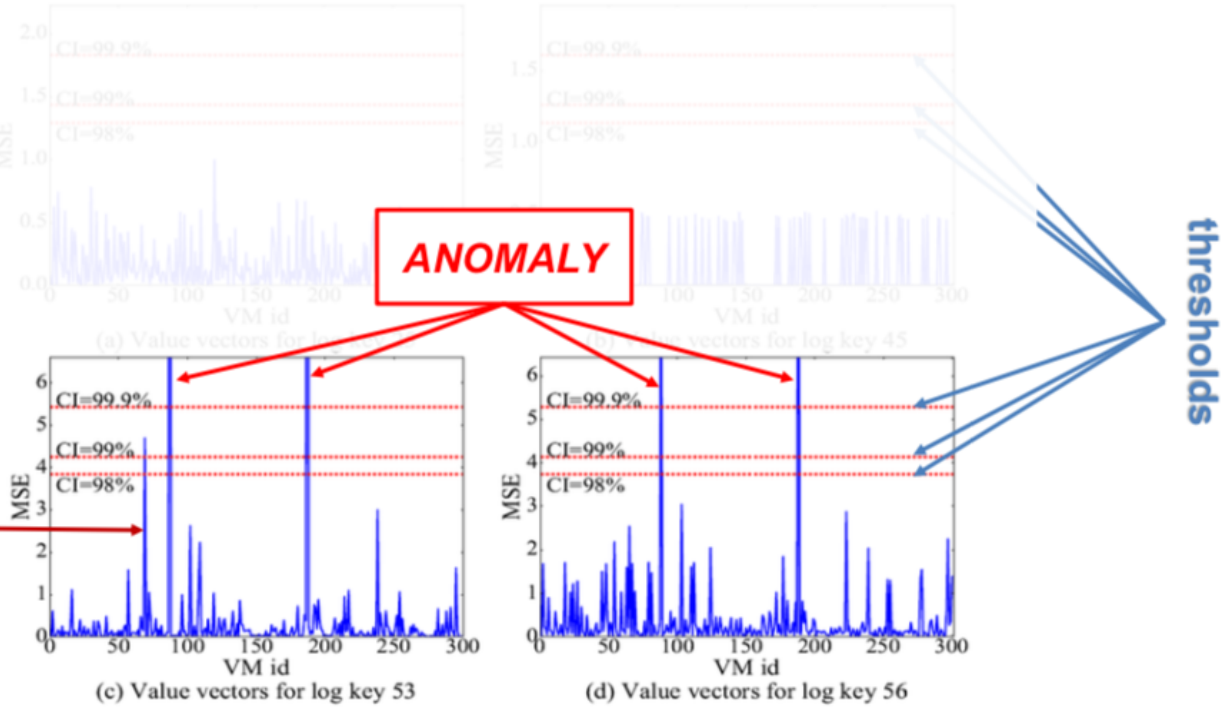
(b) Value vectors for log key 45

(c) Value vectors for log key 53

(d) Value vectors for log key 56

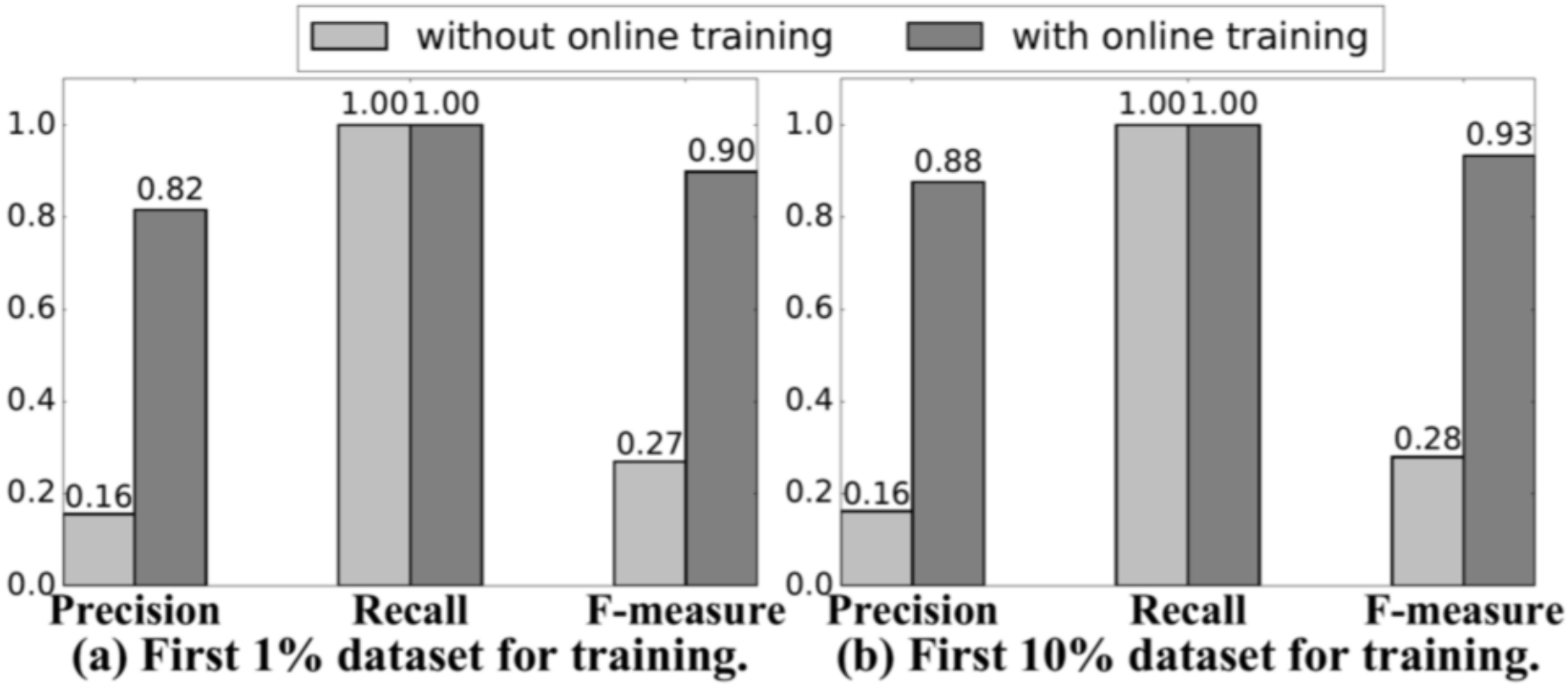
Evaluation parameter value anomaly detection

MSE:
mean square error



Evaluation results on OpenStack cloud log with different confidence intervals (CIs)

DeepLog Online Evaluation

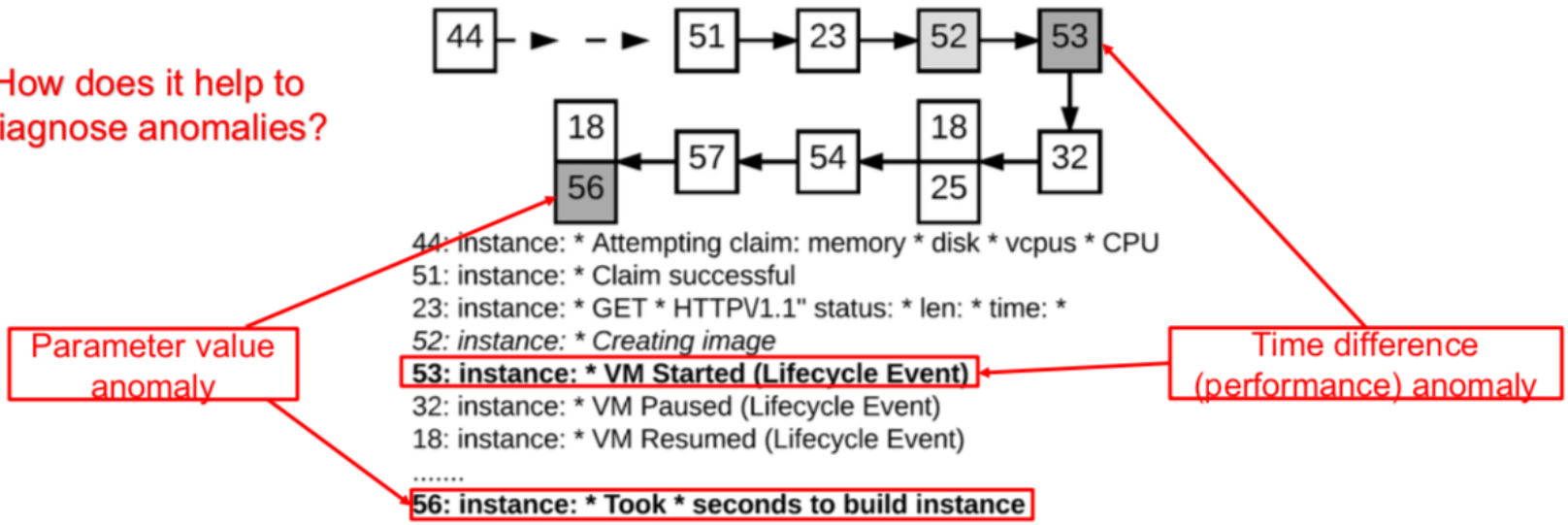


(a) First 1% dataset for training. (b) First 10% dataset for training.

Evaluation on Blue Gene/L log,
with and without online model update.

DeepLog Workflow Evaluation

How does it help to diagnose anomalies?



Constructed workflow of *VM Creation*.

(previously generated OpenStack cloud log)