Log Anomaly Detection: Interpretability and Transferability

Xiao Han, PhD Qualifying Exam
Utah State University
December 6, 2021

Education

2018/09 - 2020/05	M.S. in Data Analytics George Washington University, Washington, D.C.
2014/09 – 2016/12	M.Eng. in Computer Science Oregon State University, Corvallis, OR
2008/09 – 2012/06	B.Eng. In Computer Science and Technology Shandong University, Jinan, Shandong, China

Courses

Fall 2020	CS 5665 Introduction to Data Science	A
	CS 6890 ST: Robot Intelligence	A
Spring 2021	CS 6665 Data Mining	A
	CS 6675 Advanced Data Science & Mining	A-

Publication

[1] Xiao Han, He Cheng, Depeng Xu and Shuhan Yuan. 2021. "InterpretableSAD: Interpretable Anomaly Detection in Sequential Log Data". In Proceedings of the 2021 IEEE International Conference on Big Data (BigData 2021).

[2] Xiao Han and Shuhan Yuan. 2021. "Unsupervised Cross-system Log Anomaly Detection via Domain Adaptation". In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21). Association for Computing Machinery, New York, NY, USA, 3068–3072. (short paper)

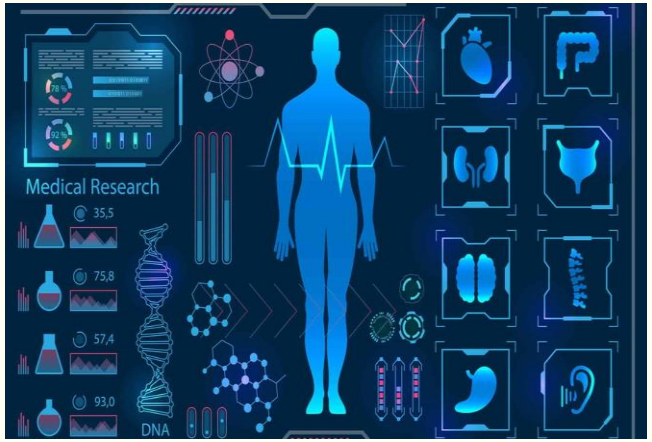
What are Anomalies

- An anomaly is an event or a pattern in the data that does not conform to the expected behavior
- Also referred to as outliers, exceptions, etc.
- Real world anomalies:

Credit Card Fraud

Cyber Intrusions

Medical Diagnostics



Images from: https://traderdefenseadvisory.com/financial-protection/credit-card-fraud/
https://www.vodigynetworks.com/how-intrusion-prevention-systems-sniff-out-sneaky-cyber-attacks/
https://www.trademed.com/news/294779083/AI-Paving-the-Way-for-New-Generation-of-Medical-Diagnostic-Devices.html

What is Log Anomaly Detection

- System logs are widely used on online services to record the status of the system.
- Anomalous logs can be useful in maintaining and increasing reliability and stability.
- Log anomaly detection is aimed to detect a point of the anomalous event or an abnormal pattern of multi-status.

What are System Logs

- Logs are generated by logging statements (print, logging.info)
- A log message (log entry) records timestamp, PID, level, and content
- Logs are unstructured data

```
1 081109 203615 148 INFO dfs.DataNode$PacketResponder: PacketResponder 1 for block blk_38865049064139660 terminating
2 081109 203807 222 INFO dfs.DataNode$PacketResponder: PacketResponder 0 for block blk_-6952295868487656571 terminating
3 081109 204005 35 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.73.220:50010 is added to blk_7128370237687728475 size 67108864
4 081109 204015 308 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_8229193803249955061 terminating
5 081109 204106 329 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_-6670958622368987959 terminating
6 081109 204132 26 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.43.115:50010 is added to blk_3050920587428079149 size 67108864
7 081109 204324 34 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.203.80:50010 is added to blk_7888946331804732825 size 67108864
8 081109 204453 34 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.250.11.85:50010 is added to blk_2377150260128098806 size 67108864
...
```

HDFS logs

Log Parsing

Logevent Template:
event1,dc2c74b7,PacketResponder <*> for block <*> terminating
event2,5d5de21c,BLOCK* NameSystem.addStoredBlock: blockMap updated: <*> is added to <*> size <*>

1 081109 203615 148 INFO dfs.DataNode$PacketResponder: PacketResponder 1 for block blk_38865049064139660 terminating
2 081109 203807 222 INFO dfs.DataNode$PacketResponder: PacketResponder 0 for block blk_-6952295868487656571 terminating
3 081109 204005 35 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.73.220:50010 is added to blk_7128370237687728475 size 67108864
4 081109 204015 308 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_8229193803249955061 terminating
5 081109 204106 329 INFO dfs.DataNode$PacketResponder: PacketResponder 2 for block blk_-6670958622368987959 terminating
6 081109 204132 26 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.43.115:50010 is added to blk_3050920587428079149 size 67108864
7 081109 204324 34 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.203.80:50010 is added to blk_7888946331804732825 size 67108864
8 081109 204453 34 INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.250.11.85:50010 is added to blk_2377150260128098806 size 67108864

Raw Data

New log:
dfs.DataNode$PacketResponder: PacketResponder 1 for block blk_38865049064139660 terminating

Logkey template

New log:
dfs.DataNode$PacketResponder: PacketResponder 0 for block blk_-6952295868487656571 terminating

Logkey template

evnnt1,dfs.DataNode$PacketResponder: PacketResponder 1 for block blk_38865049064139660 terminating

New log:
INFO dfs.FSNamesystem: BLOCK* NameSystem.addStoredBlock: blockMap updated: 10.251.73.220:50010 is added to blk_7128370237687728475 size 67108864

Logkey template

event1, PacketResponder <*> for block <*> terminating

Log Parsing

1,event1
2,event1
3,event2
4,event1
5,event1
6,event2
7,event2
8,event2

InterpretableSAD: Interpretable Anomaly Detection in Sequential Log Data

Xiao Han[1], He Cheng[1], Depeng Xu[2], and Shuhan Yuan[1]
[1]Utah State University
[2]University of Arkansas
In Proceedings of the 2021 IEEE International Conference on Big Data (BigData 2021).

Goals

- To train a binary classifier for log anomaly detection, we propose a novel negative sampling strategy to generate potential anomalous samples.
- To achieve anomalous event detection, we novelty apply a model interpretation approach, Integrated Gradients (IG).
- As IG relies on an appropriate baseline input for feature attributions, we further propose a novel baseline generation algorithm for log anomaly detection.

Preliminary – Anomaly Detection in Log Data

- Traditional supervised learning -> Require an enormous number of labeled data
  - Logistic regression, decision tree, and SVM
- Traditional unsupervised learning -> Hard to capture the order information of sequencial data
  - PCA, Isolation forest, and OC-SVM
- Deep learning -> No detailed information on the sub-sequence level
  - DeepLog and LogAnomaly

Preliminary – Data Augmentation

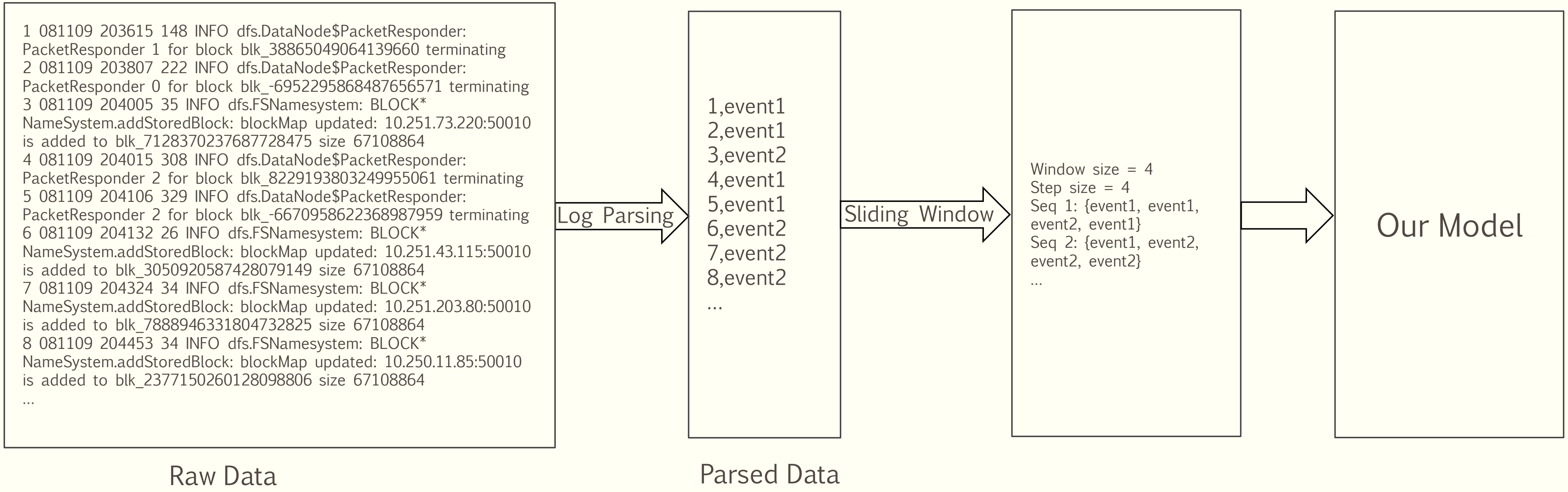
- Data augmentation technique is to tackle the scarcity of labeled data issue by artificially expanding the labeled dataset.
- Extensively used in image classification and natural language processing.
  - Rotation and flip for image data, synonym replacement for text data
- Negative sampling is a special data augmentation technique.

Preliminary – Interpretable Machine Learning

- Interpretable machine learning aims at providing a human understandable explanation about the decisions.
- The interpretable anomaly detection models are very limited.
- The attention mechanism provides an attention score that is more about the correlation among events instead of the correlation between events and the label.

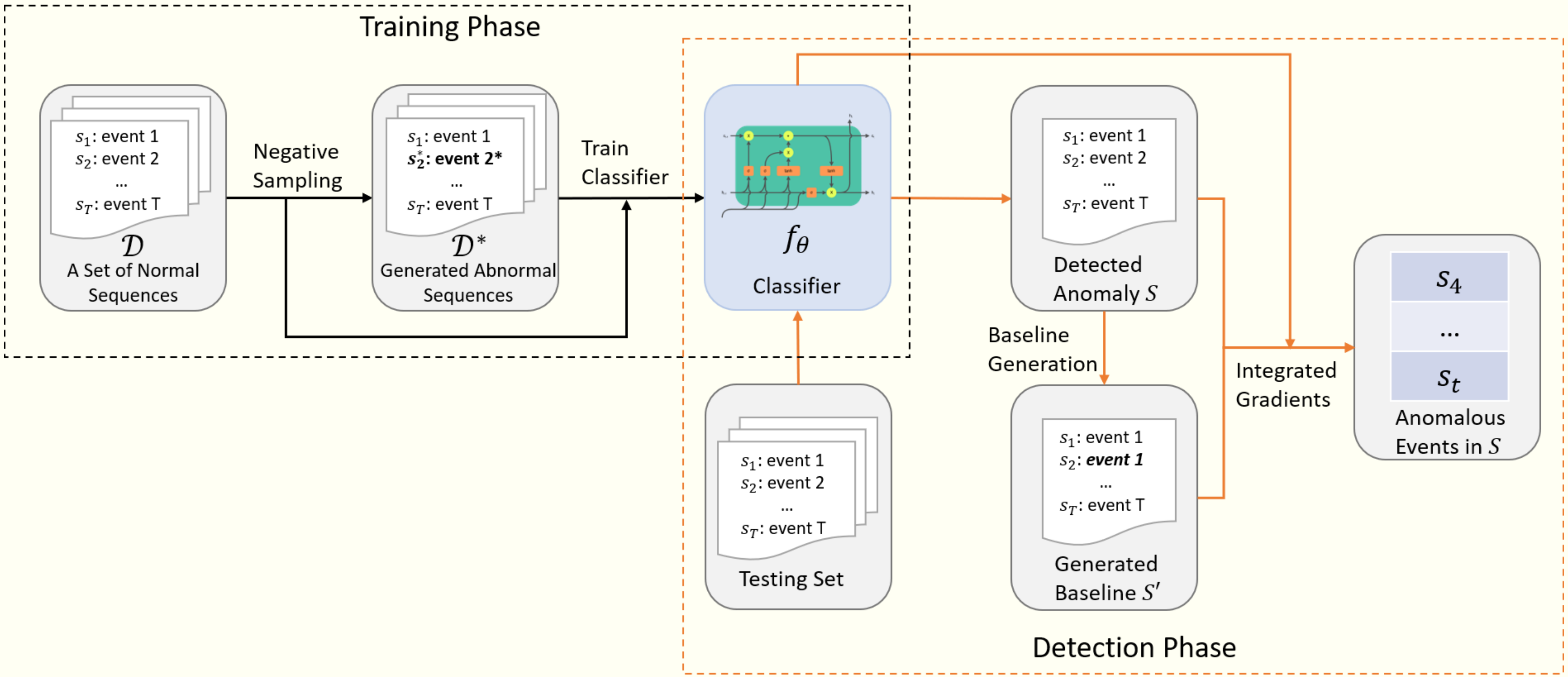
Problem Statement

- Consider a log sequence of discrete events $S = \{s_1, \dots, s_t, \dots, s_T\}$, where $s_t \in \mathcal{E}$ indicates the event at the t-th position, and $\mathcal{E}$ is a set of unique events.



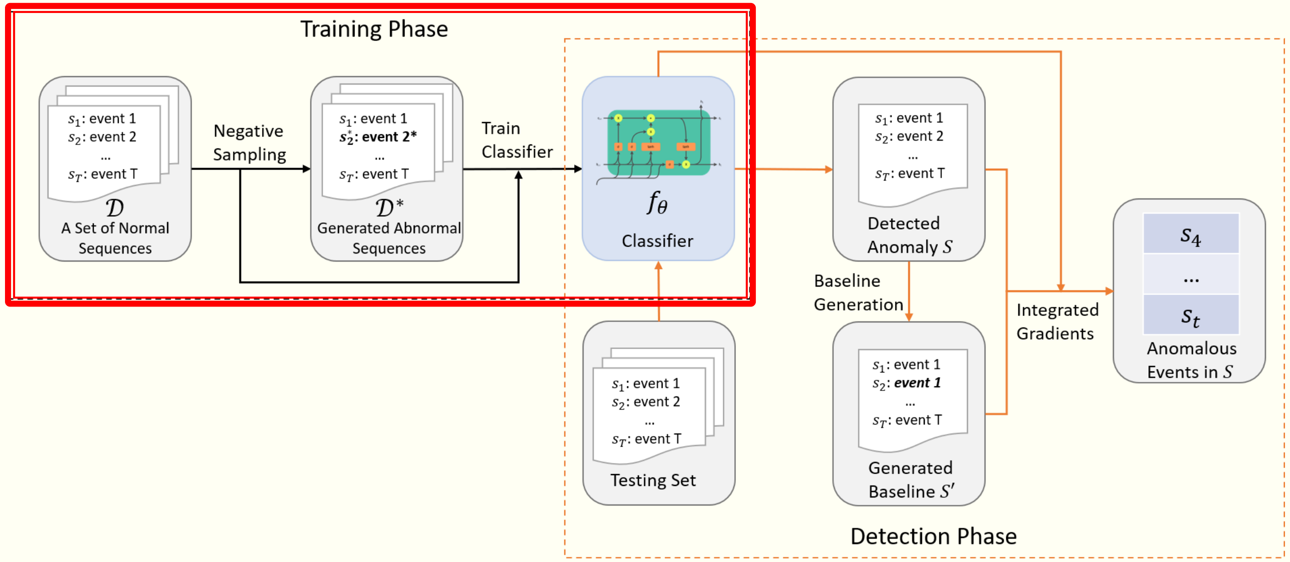
Logevent Template: $\mathcal{E}$
event1,dc2c74b7,PacketResponder <*> for block <*> terminating
event2,5d5de21c,BLOCK* NameSystem.addStoredBlock: blockMap updated: <*> is added to <*> size <*>

Framework of InterpretableSAD



Training Phase – Negative Sampling

- In order to train an accurate binary classifier, we aim to generate a dataset $\mathcal{D}^*$ with sufficient anomalous samples that can cover common anomalous scenarios.



Training Phase – Negative Sampling Cont.

- Two anomalous scenarios for anomalous log sequence generation:
  1. Anomalous events in the sequences
  2. Regular events happen in an unusual context

- Normal :

- Anomalous scenario 1:

- Anomalous Scenario 2:

Training Phase – Negative Sampling Algorithm

Algorithm 1: Negative Sampling

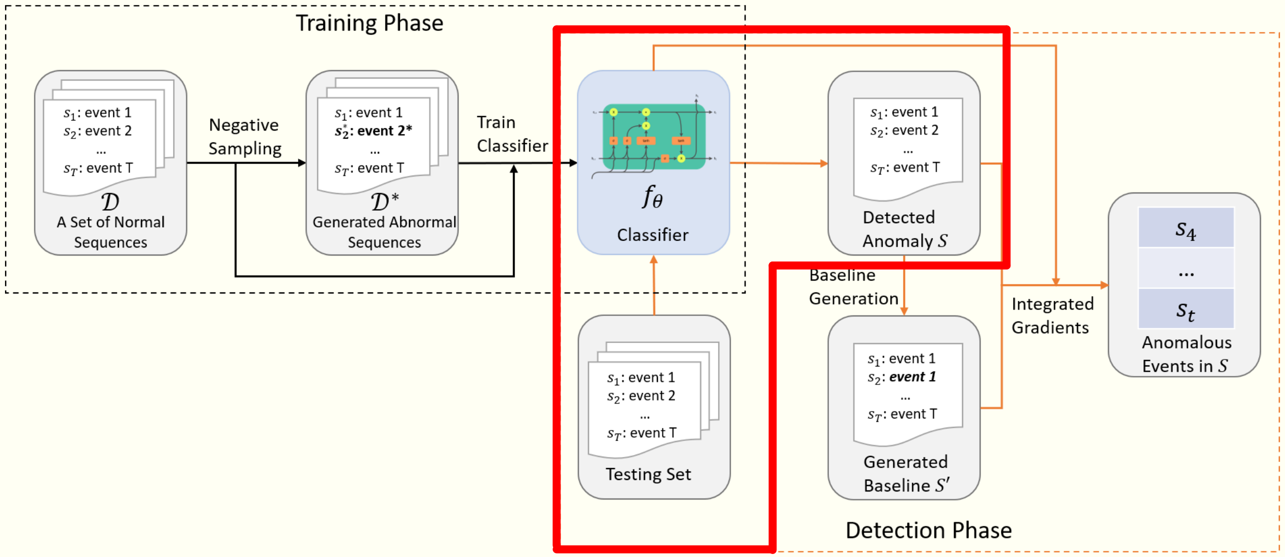
Input : Training set $\mathcal{D}$, Negative sample size M
Output: Negative sample set $\mathcal{D}^*$

Generate a bigram event dictionary $\mathcal{B}$ based on $\mathcal{D}$
for $i = 0$ **to** M **do**
    Randomly select S from $\mathcal{D}$
    $ind \leftarrow$ Randomly select r indices of events from S
    for t in ind **do**
        $(s_t, s^*_{t+1}) \leftarrow$ randomly select or generate a rare or never observed bigram in $\mathcal{B}$
        $(s_t, s_{t+1}) \leftarrow (s_t, s^*_{t+1})$
    $S^* \leftarrow S, \quad \mathcal{D}^* += S^*$
return $\mathcal{D}^*$

Detection Phase – Anomaly Detection on a Sequence Level

- After generating a set of anomalous sequences $\mathcal{D}^*$, we use both $\mathcal{D}$ and $\mathcal{D}^*$ to train a binary classification model $f : S \rightarrow [0, 1]$.
- Specifically, we use an LSTM with a linear layer as the classification model f.
- We further ada the cross-entropy loss to train the neural network:

$$\mathcal{L} = \sum_{j \in \mathcal{D}^* \cup \mathcal{D}} -y_j log\hat{y}_j - (1 - y_j)\log(1 - \hat{y}_j)$$



Detection Phase – Anomalous Event Detection

- Integrated Gradients (IG) is a model interpretable technique that can interpret prediction results by attributing input features.
- For example,

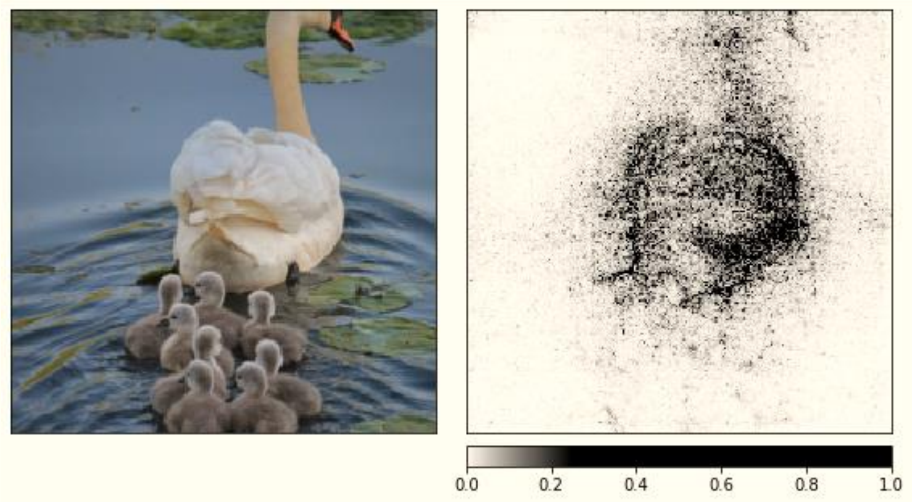


Image Classification

Legend: Negative Neutral Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
pos	pos (0.96)	pos	1.29	it was a fantastic performance ! #pad
pos	pos (0.87)	pos	1.56	best film ever #pad #pad #pad #pad
pos	pos (0.92)	pos	1.14	such a great show ! #pad #pad
neg	neg (0.29)	pos	-1.11	it was a horrible movie #pad #pad
neg	neg (0.22)	pos	-1.03	i 've never watched something as bad
neg	neg (0.07)	pos	-0.84	that is a terrible movie . #pad

Sentiment Analysis

Images from: https://captum.ai/tutorials/

Detection Phase – Anomalous Event Detection Cont.

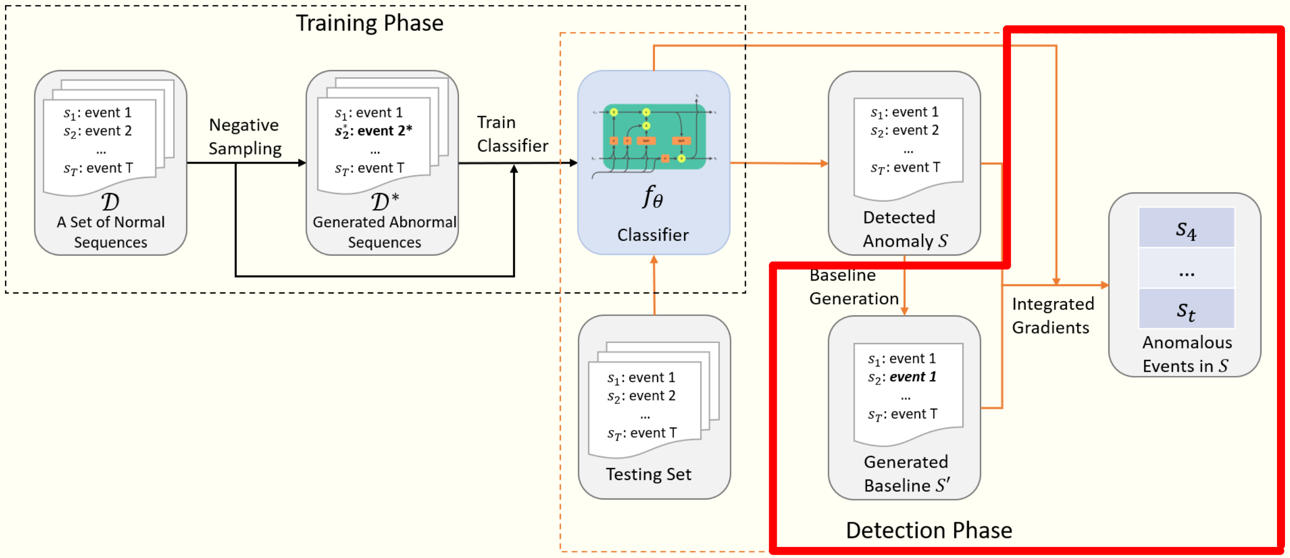
- Formally, given a neural network $f_\theta : S \rightarrow [0,1]$, integrated gradients are attributions of the prediction at input S relative to a baseline input S' as a vector $A_{f_\theta}(S, S') = (a_1, \ldots, a_T)$, where a_t is the contribution of s_t to the prediction $f_\theta(S)$.

EventID	Score
09a53393	-0.20
09a53393	-0.20
3d91fa85	-0.51
09a53393	-0.20
0567184d	1.73
d38aa58d	-0.62
e3df2680	0.17
0567184d	1.73
d38aa58d	-0.62
e3df2680	0.17
…	
5d5de21c	0.00
…	
d63ef163	-0.34
…	
dba996ef	-1.00

09a53393	Receiving block <*> src: <*> dest: <*>
3d91fa85	BLOCK* NameSystem. allocateBlock: <*> <*>
0567184d	Receiving empty packet for block <*>
d38aa58d	PacketResponder <*> for block <*> <*>
e3df2680	Received block <*> of size <*> from <*>
5d5de21c	BLOCK* NameSystem. addStoredBlock: blockMap updated: <*> is added to <*> size <*>
d63ef163	BLOCK* NameSystem.delete: <*> is added to invalidSet of <*>
dba996ef	Deleting block <*> file <*>

Detection Phase – Baseline Generation

- Finding a reasonable baseline is an essential step for applying the IG method.
- For the image classification models, the black image is widely used as a baseline, while the zero-embedding matrix is a common baseline for the text classification task.
- Therefore, we propose to generate a unique baseline for each sequence.



Detection Phase – Baseline Generation Cont.

- Sort the events based on their frequencies in the training set
- Replace the lowest frequent event with its preceding event
- Check whether the sequence is normal

Algorithm 2: Baseline Generation

Input : Neural network f_θ, Anomalous sample S, Training set $\mathcal{D}$, Replacement Threshold τ
Output: Baseline S'

$i = 0$
while $f_\theta(S)$ *is not normal* & $i < \tau$ **do**
  $s_t \leftarrow$ Select the event in S with the lowest frequency based on $\mathcal{D}$
  $s_t \leftarrow s_{t-1}$, $i += 1$
$S' \leftarrow S$
return S'

Experiment - Datasets

- Log parser – Drain; Window size – 100; Step size – 20.
- Training dataset consists of 100,000 normal log sequences and 2,000,000 generated anomalous sequences for each log dataset .

Dataset	# of Unique Log Keys	# of Log Sequences		# of Log Keys in Anomalous Sequences	
		Normal	Anomalous	Normal	Anomalous
HDFS	48 (19)	458,223	16,838	N/A	N/A
BGL	396 (318)	19,430	4,190	326,491	7,139
Thunderbird	806 (774)	22,538	76,189	6,866,417	479,883

TABLE I: Statistics of Test Datasets

Experiment - Baselines for Anomalous Log Sequence Detection

- Traditional machine learning models:
  - Principal Component Analysis (PCA)
  - One-Class SVM (OCSVM)
  - Isolation Forest (iForest)
  - LogCluster
- Deep learning models:
  - DeepLog
  - LogAnomaly

Experiment - Results on Anomalous Log Sequence Detection

Method	BGL			Thunderbird			HDFS		
	Precision	Recall	F-1 score	Precision	Recall	F-1 score	Precision	Recall	F-1 score
PCA	67.91	99.79	80.82	94.83	84.43	89.33	97.77	42.12	58.88
iForest	73.13	38.19	50.17	95.06	17.92	30.15	41.59	58.80	48.72
OCSVM	24.60	100	39.49	87.13	100	93.12	6.68	90.58	12.44
LogCluster	8.03	15.97	10.69	86.56	22.94	36.26	98.37	67.45	80.03
DeepLog	42.39	52.08	46.74	82.42	81.36	81.89	56.98	48.37	52.32
LogAnomaly	42.58	53.17	47.29	81.69	82.11	81.90	55.85	48.03	51.65
InterpretableSAD	94.25	88.47	**91.27**	97.31	96.42	**96.86**	92.31	87.04	**89.60**

Experiment - Results on Anomalous Log Sequence Detection

Method	BGL			Thunderbird			HDFS		
	Precision	Recall	F-1 score	Precision	Recall	F-1 score	Precision	Recall	F-1 score
PCA	67.91	99.79	80.82	94.83	84.43	89.33	97.77	42.12	58.88
iForest	73.13	38.19	50.17	95.06	17.92	30.15	41.59	58.80	48.72
OCSVM	24.60	100	39.49	87.13	100	93.12	6.68	90.58	12.44
LogCluster	8.03	15.97	10.69	86.56	22.94	36.26	98.37	67.45	80.03
DeepLog	42.39	52.08	46.74	82.42	81.36	81.89	56.98	48.37	52.32
LogAnomaly	42.58	53.17	47.29	81.69	82.11	81.90	55.85	48.03	51.65
InterpretableSAD	94.25	88.47	**91.27**	97.31	96.42	**96.86**	92.31	87.04	**89.60**

Experiment - Results on Anomalous Log Sequence Detection

Method	BGL			Thunderbird			HDFS		
	Precision	Recall	F-1 score	Precision	Recall	F-1 score	Precision	Recall	F-1 score
PCA	67.91	99.79	80.82	94.83	84.43	89.33	97.77	42.12	58.88
iForest	73.13	38.19	50.17	95.06	17.92	30.15	41.59	58.80	48.72
OCSVM	24.60	100	39.49	87.13	100	93.12	6.68	90.58	12.44
LogCluster	8.03	15.97	10.69	86.56	22.94	36.26	98.37	67.45	80.03
DeepLog	42.39	52.08	46.74	82.42	81.36	81.89	56.98	48.37	52.32
LogAnomaly	42.58	53.17	47.29	81.69	82.11	81.90	55.85	48.03	51.65
InterpretableSAD	94.25	88.47	**91.27**	97.31	96.42	**96.86**	92.31	87.04	**89.60**

Experiment - Baselines for Anomalous Event Detection

- Anchors
- Low-Freq
- Integrated Gradients without our IG baseline generation

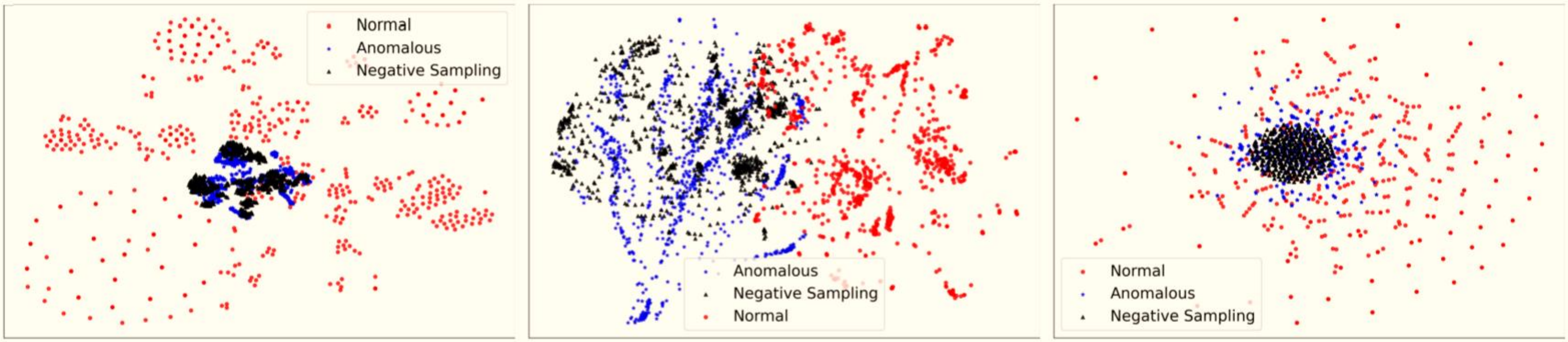
Experiment - Results on Anomalous Event Detection

- We consider two scenarios, with or without a validation set consisting of 10% anomalous sequences in the testing datasets to tune a detection threshold for anomalous event detection.

Method	BGL			Thunderbird		
	Precision	Recall	F-1 score	Precision	Recall	F-1 score
Anchors	0.31	8.56	0.60	4.58	14.62	6.98
Low-Freq	38.76	93.59	54.82	52.61	99.00	68.70
IG w/o val	6.56	90.27	12.23	10.36	85.65	18.49
IG w/ val	42.43	73.83	53.89	20.92	44.48	28.45
InterpretableSAD w/o val	50.87	89.23	64.80	94.98	86.79	90.70
InterpretableSAD w/ val	68.92	82.53	**75.11**	93.84	98.31	**96.02**

Experiment - Visualization of the normal, anomalous, and generated anomalous sequences

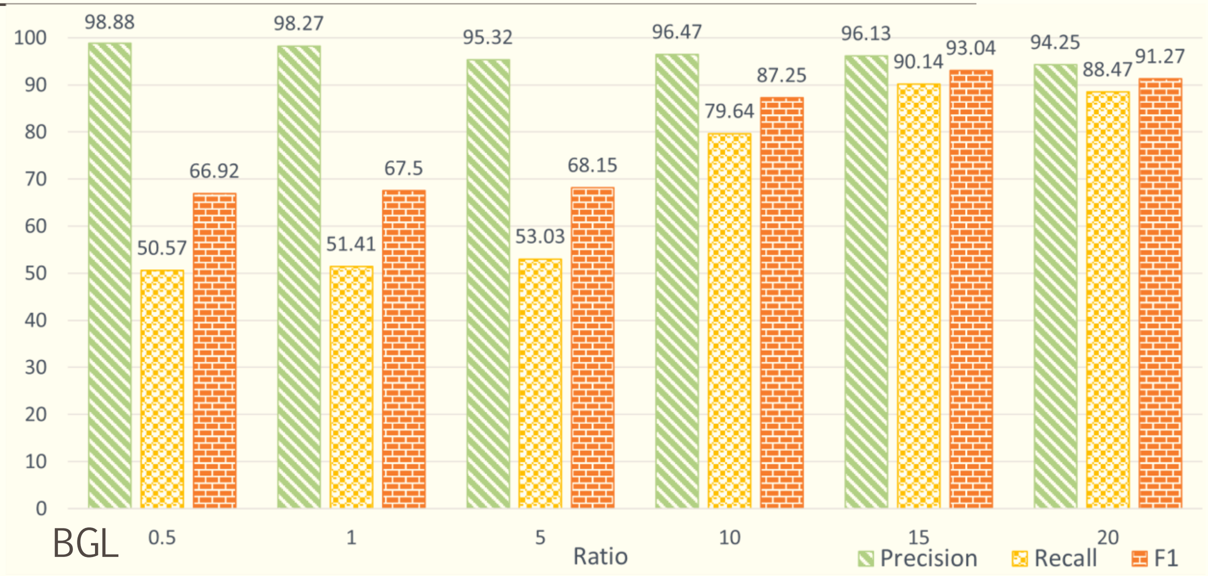
- We randomly select 1,000 sequences of normal, anomalous, and generated samples, separately.

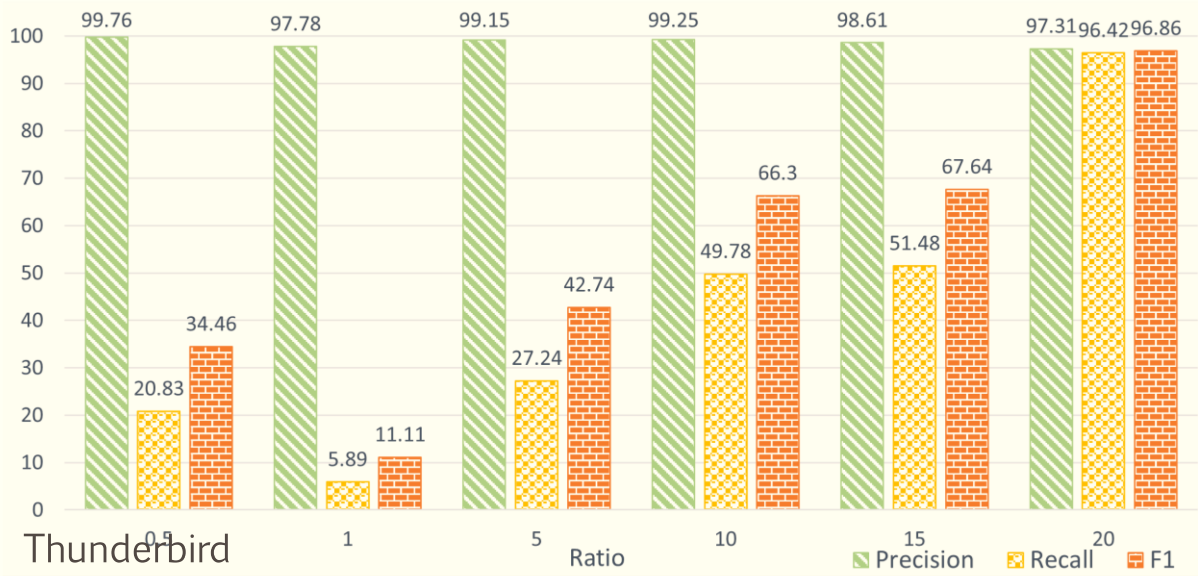


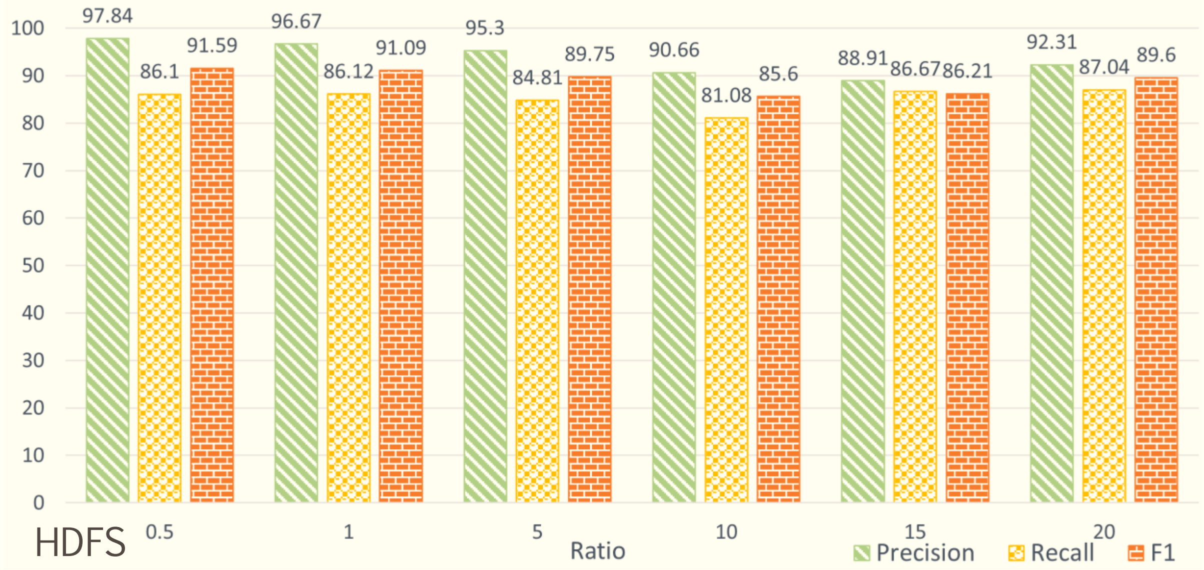
(a) BGL (b) Thunderbird (c) HDFS

Experiment – Sensitivity Analysis on the Size of Generated Anomalous Sequences

The ratios of the anomalous datasets to the training dataset are 0.5, 1, 5, 10, 15, 20, respectively.







Conclusions

- Propose a novel framework to detect anomalous sequences as well as anomalous events in the sequences.
- Propose a novel negative sampling algorithm that can accurately generate anomalous samples.
- Apply an interpretable machine learning technique, Integrated Gradients (IG), to detect potential anomalous events.
- Propose a novel feature attribution baseline generation algorithm.
- Experimental results on three log datasets show that our model can achieve state-of-the-art performance on the anomalous sequence and event detection.

Unsupervised Cross-system Log Anomaly Detection via Domain Adaptation

Xiao Han and Shuhan Yuan
Utah State University
In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM ‘21).

Why Do We Need LogTAD

Develop a cross-system anomaly detection model that only uses samples.

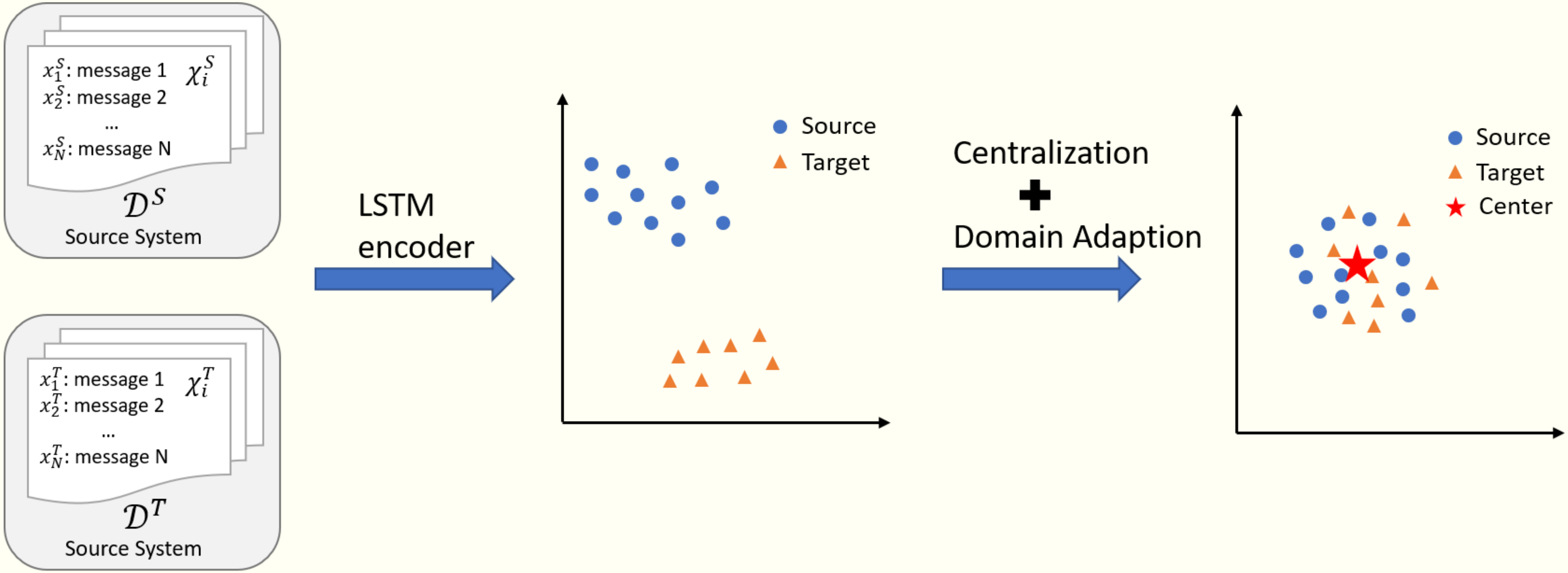
Require a small number of samples from the target system (newly deployed).

Problem Statement

- Prerequisites: A dataset $\mathcal{D}$ consisting of normal log sequences from the source system $\mathcal{D}^S$ and a small number of normal log sequences from the target system $\mathcal{D}^T$.

- Goal: Building an unsupervised and transferable log anomaly detection model to detect the anomalous log sequences from both source and target system.

Framework of LogTAD



Task I - Log Sequence Centralization

- Encodes the log messages in a sequence to a sequence representation,

$$\mathbf{h}_n = LSTM(\mathbf{x}_n, \mathbf{h}_{n-1}), \tag{1}$$

$$\mathbf{v} = \mathbf{h}_N. \tag{2}$$

- Inspired by the DeepSVDD that the normal log sequences should be in a hypersphere and close to the center in the embedding space,

$$\mathbf{c} = Mean(\mathbf{v}_i^{\epsilon}), \text{ where } \epsilon \in \{\mathrm{S}, \mathrm{T}\}. \tag{3}$$

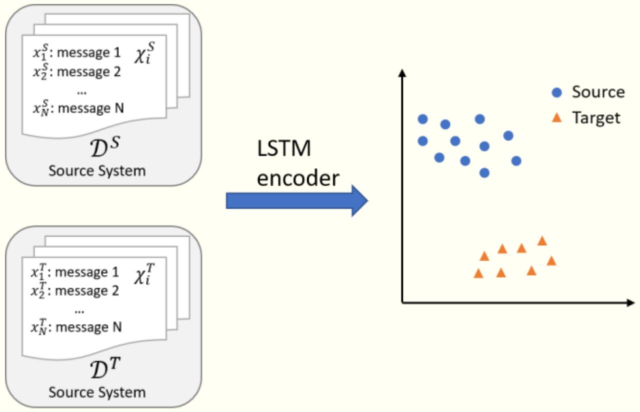
- To make the representation of normal log sequences close to the center $\mathbf{c}$, we develop the following objective function,

$$\mathcal{L}_{en} = \sum_{\epsilon \in \{\mathrm{S},\mathrm{T}\}} \sum_{i=1}^{M_\epsilon} || \mathbf{v}_i^{\epsilon} - \mathbf{c} ||^2, \tag{4}$$

where M_ϵ is the number of samples from the specific domain.

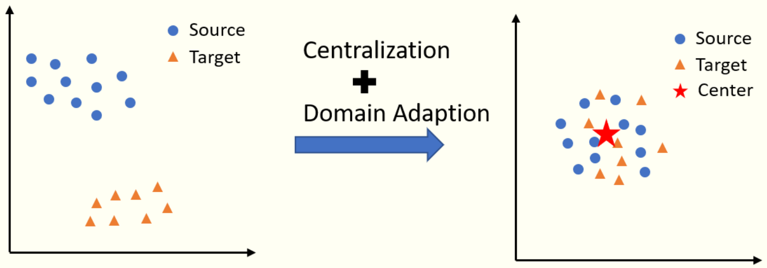
Task II - System-agnostic Representation

Although we adopt one shared LSTM model to map log sequences into a hypersphere, the representations of log sequences from different systems can be still located in different regions.



Hence, we propose an adversarial training method for cross-system data mapping.

In specific, we formulate the adversarial training with a discriminator D and a shared LSTM as a generator G.



Task II - System-agnostic Representation via Domain Adversarial Training

- Discriminator D is used to distinguish whether the representations of log sequences are from the source or target system,

$$D(\mathbf{v}^{\epsilon}) = \sigma(\mathbf{w}^{T}\mathbf{v}^{\epsilon} + b),$$

where $\epsilon \in \{S, \mathrm{T}\}$, $\sigma(\bullet)$ indicates the logistic function, $\mathbf{w}$ and b are the trainable parameters.

- The shared generator G is trained to make representations of log sequences,

$$\mathbf{v}^{\epsilon} = G(\chi^{\epsilon}),$$

where $\epsilon \in \{S, \mathrm{T}\}$.

Task II - System-agnostic Representation via Domain Adversarial Training Cont.

- With the adversarial training objective function,

$$\mathcal{L}_{adv} = \min_{G} \max_{D} (\mathbb{E}_{\chi^s \sim P_{source}} [\log D(G(\chi^S))] + \mathbb{E}_{\chi^T \sim P_{target}} [\log(1 - D(G(\chi^T))]),$$

our goal is to mix the distributions of source and target log sequences.

- Final objective function for LogTAD,

$$\mathcal{L} = \mathcal{L}_{en} + \lambda \mathcal{L}_{adv}.$$

Cross-system Log Anomaly Detection

- For a log sequence χ^{ϵ},

$$\hat{y}_{\chi^{\epsilon}} = \begin{cases} anomalous, & if\ ||G(\chi^{\epsilon}) - \mathbf{c}||^2 > \gamma^{\epsilon} \\ normal, & else \end{cases}$$

- where $\epsilon \in \{\mathrm{S}, \mathrm{T}\}$ and γ^{ϵ} can be derived from a small validation set.

Experiment - Datasets

- Statistics of the Datasets

Dataset	# of Logs	# of Log Sequences	
		Normal	Anomalous
BGL	1,212,150	265,583	37,450
TB	3,737,209	565,817	368,481

- Statistics of Shared Words Across Systems

	BGL Normal	BGL Anomalous	TB Normal	TB Anomalous
BGL Normal	664	133	254	25
BGL Anomalous	133	195	99	16
TB Normal	254	99	1753	49
TB Anomalous	25	16	49	54

Experiment - Baselines

- Unsupervised Log Anomaly Detection Approaches
  - PCA
  - LogCluster
  - DeepLog
  - DeepSVDD
- Supervised Transfer Learning Approach for Log Anomaly Detection
  - LogTransfer

Experiment - Results Compared with Unsupervised Approaches

- Training dataset contains 100,000 normal sequences from the source system and 1,000 normal sequences from the target system.

BGL -> TB				
Method	Source		Target	
	F1	AUC	F1	AUC
PCA w/o TB	0.642	0.816	0.558	0.504
LogCulster w/o TB	0.713	0.829	0.559	0.504
DeepLog w/o TB	0.578	0.867	0.556	0.500
DeepSVDD w/o TB	0.566	0.789	0.577	0.646
LogTAD	0.926	0.964	0.758	0.804

TB -> BGL				
Method	Source		Target	
	F1	AUC	F1	AUC
PCA w/o BGL	0.760	0.779	0.229	0.658
LogCulster w/o BGL	0.724	0.716	0.223	0.500
DeepLog w/o BGL	0.660	0.677	0.223	0.500
DeepSVDD w/o BGL	0.794	0.808	0.195	0.497
LogTAD	0.788	0.797	0.845	0.909

Experiment - Results Compared with Unsupervised Approaches Cont.

BGL -> TB				
Method	Source		Target	
	F1	AUC	F1	AUC
PCA w/ TB	0.322	0.587	0.731	0.776
LogCulster w/ TB	0.530	0.746	0.677	0.716
DeepLog w/ TB	0.662	0.854	0.590	0.619
DeepSVDD w/ TB	0.499	0.725	0.567	0.616
LogTAD	0.926	0.964	0.758	0.804

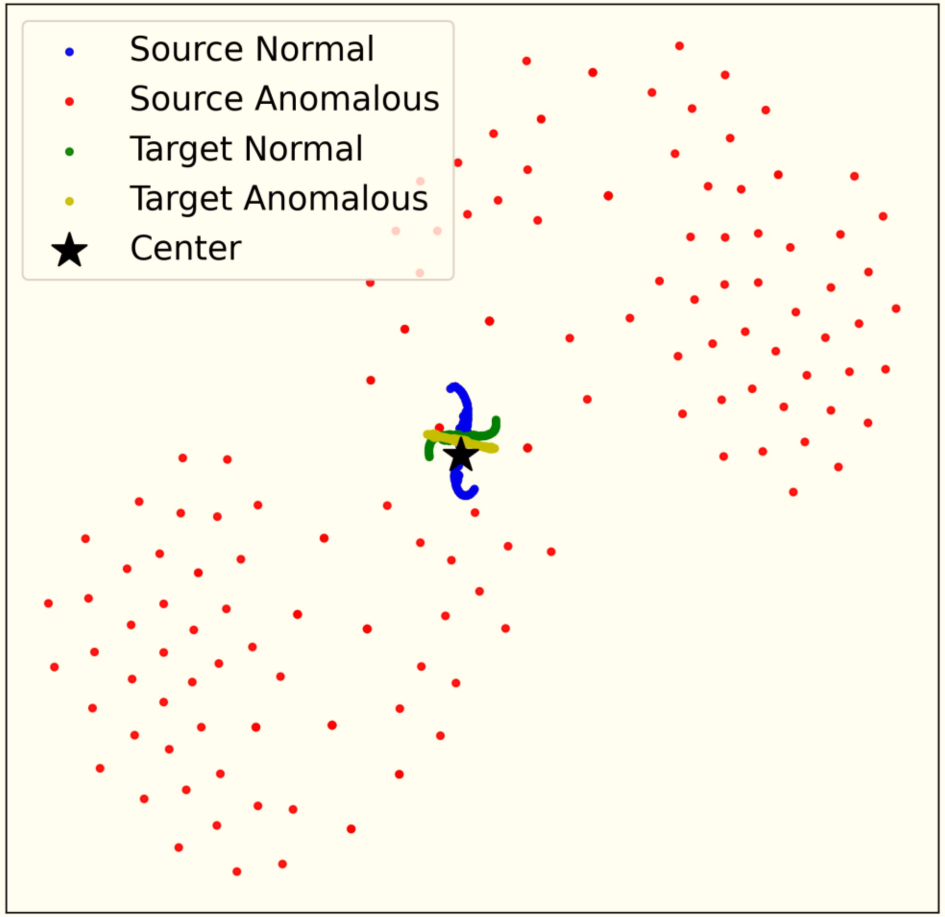
TB -> BGL				
Method	Source		Target	
	F1	AUC	F1	AUC
PCA w/ BGL	0.789	0.798	0.577	0.773
LogCulster w/ BGL	0.708	0.688	0.697	0.886
DeepLog w/ BGL	0.687	0.701	0.527	0.843
DeepSVDD w/ BGL	0.660	0.699	0.196	0.537
LogTAD	0.788	0.797	0.845	0.909

Experiment - Results Compared with Supervised Approach

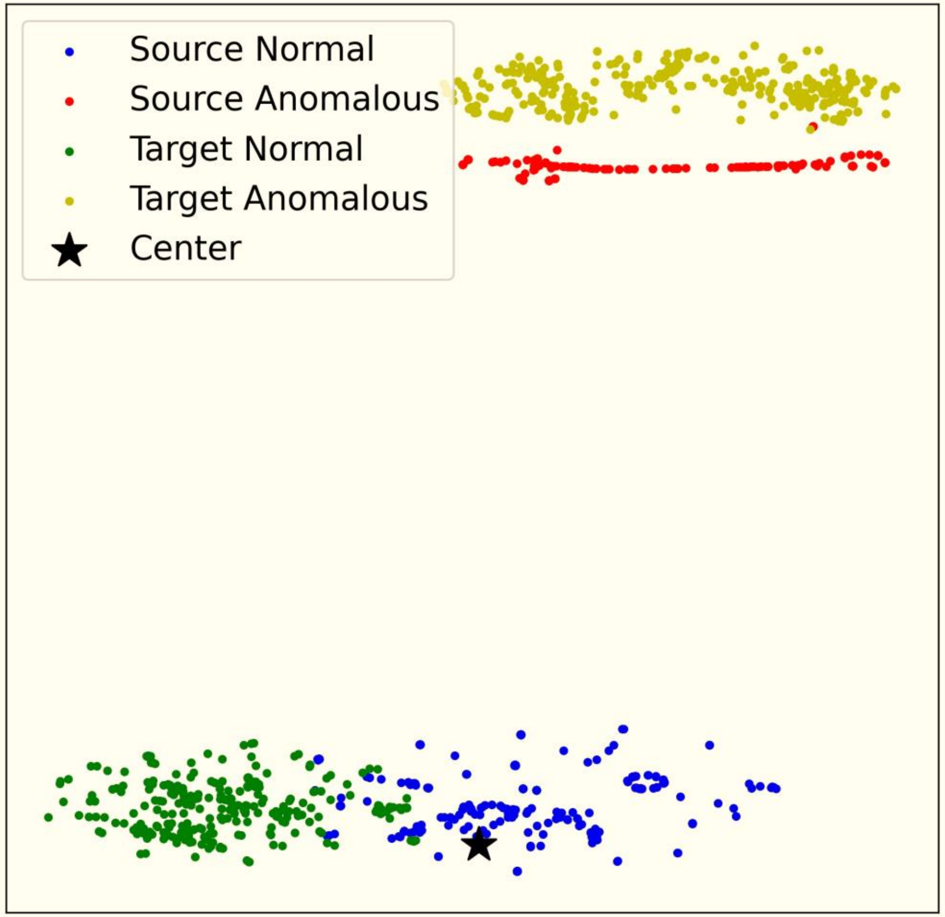
BGL -> TB				
Method	Source		Target	
	F1	AUC	F1	AUC
LogTransfer	0.971	0.972	0.792	0.828
LogTAD	0.926	0.964	0.758	0.804

TB -> BGL				
Method	Source		Target	
	F1	AUC	F1	AUC
LogTransfer	0.995	0.995	0.788	0.833
LogTAD	0.788	0.797	0.845	0.909

Experiment - Log Sequences Visualization



Without domain adaption



With domain adaption

Conclusions

- We propose an unsupervised cross-system log anomaly detection framework.
- LogTAD utilizes the domain adversarial adaption to make the log data from different systems follow similar distributions.
- LogTAD can detect anomalies in different systems with large distances to the center.
- The experiment results show the effectiveness of our framework.

THANK YOU FOR YOUR ATTENTION!

Any questions?