# Unsupervised Cross-system Log Anomaly Detection via Domain Adaptation

20221031 이연재, 김미주

# Index

# Introduction

**Cold-start problem** : 새로 배포되는 system 은 anomaly log 를 자동적으로 탐지해야 하지만, 그러한 탐지 모델을 만들기 위해서는 충분한 데이터와 시간이 필요하다.

Solution : Transfer Learning-based Approach

- **기존 transfer learning 모델의 단점** : source 와 target system 의 normal 과 anomaly log data 둘 다를 요구한다.
  - 현실은 target system 에서 anomalous data 를 많이 모으기는 힘들다
- **LogTAD** : source 와 target system 둘 다에서 labeled 된 anomalous sample 을 요구하지 않음.
  - source 와 target system 의 normal log data 에 기반하여 system invariant center representation 을 끌어낸다.
  - source 와 target system 의 log sequence 를 비슷한 분포를 가지는 같은 '구' 로 mapping 하기 위해 domain adversarial 을 사용 ( 두 시스템의 **shared center** 를 끌어냄)

# Introduction

**<main contribution>**

1. normal samples 만을 사용해서 cross-system anomaly detection model 을 개발
2. LogTAD 는 target system 으로부터의 적은 수의 sample 을 요구함 -> cold-start problem of anomaly detection 를 해결
3. 실험결과는 LogTAD 가 source, target system 양쪽에서 좋은 성능을 보임을 확인한다.

# 2. FrameWork

# Framework

- S : source system
- T : target system
- X : system 으로부터 생성된 log sequence $\mathcal{X} = \{x_n\}_{n=1}^{N}$
  - xn : n 번째 log message
- dataset D : normal log sequence
  - Ds : source system 으로부터의 normal log sequence $\mathcal{D}^S = \{\mathcal{X}_i^S\}_{i=1}^{M_S}$
  - Dt : target system 으로부터의 normal log sequence (small number)

$$\mathcal{D}^T = \{\mathcal{X}_i^T\}_{i=1}^{M_T} \ (M_T << M_S)$$

# Framework

LogTAD 의 framework 는 두 part 로 구성된다.

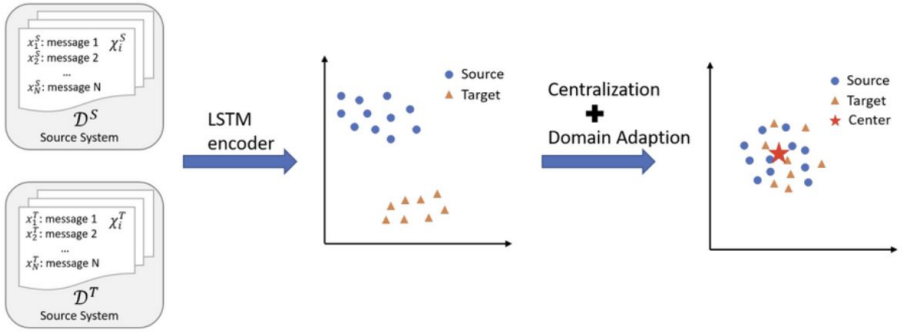



**Figure 1: Framework of LogTAD**

1. **encoder(LSTM)** : 양쪽 system 의 log sequence 들을 mapping
2. 양쪽 system 의 log representation 을 하나의 **구(hypersphere) 에 mapping**
   - 구는 shared **center** 를 가짐
   - normal sequence 들을 구의 중심에 가까이 mapping 하는데 DeepSVDD 의 idea 사용함

# Framework - Log sequence representation

**cross-system anomaly detection 의 template** : raw message 의 정보를 사용 **=>** system 간에 transfer 가능

1. **word embedding** : word2vec 사용 → log message 의 각 word 를 표현

2. 한 log message 의 word embedding 의 **평균(mean)** 을 낸다 → 하나의 log message 의 representation 을 얻기 위한 과정

3. **encoder(shared LSTM model)** 가 log message sequence 를 sequence **representation 으로 encoding** 한다.

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

   - xn : n 번째 feature vector
   - hn : LSTM 모델의 n 번째 hidden vector

4. → **마지막 hidden vector:hN** 은 전체 sequence 의 정보를 capture 한다 ⇒ **v = hN(log sequence representation)**

# Framework - Log sequence centralization

1. 모든 log sequence 의 center **c = 모든 log sequence 의 평균**

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

2. objective function

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

→ 위 objective function 을 **최소화** 시켜서 **normal log sequence 를 구의 중심에 가깝게** 만든다.

# Framework - System-agnostic representation via domain adversarial training

- **Adversarial training 의 objective function**

$$\mathcal{L}_{adv} = \min_G \max_D \left( \mathbb{E}_{\mathcal{X}^S \sim P_{\text{source}}} [\log D(G(\mathcal{X}^S))] + \mathbb{E}_{\mathcal{X}^T \sim P_{\text{target}}} [\log(1 - D(G(\mathcal{X}^T)))] \right),$$

  - Xs : source log sequences
  - Xt : target system 의 log sequences
  - **minmax optimization** 으로 training 함으로써 , shared LSTM 으로부터 생성된 representation 은 discriminator 를 mislead 할 수 있음
    - **min G** : log representation 을 생성하는 LSTM 모델의 generator G => 서로 다른 system 으로부터의 log representation 이 비슷한 분포를 갖도록 $\text{i.e., } \mathbf{v}^\epsilon = G(\mathcal{X}^\epsilon)$
    - **max D** : log representation 의 출처가 어딘지 예측하는 FCNN 모델의 discriminator D => target 인지 source 인지 판별이 불가능하도록 $\text{i.e., } D(\mathbf{v}^\epsilon) = \sigma(\mathbf{w}^T \mathbf{v}^\epsilon + b)$

# Framework - System-agnostic representation via domain adversarial training

- **마지막 training 의 objective function**

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

  - **λ** : two parts(encoder, adversarial)를 조정하는 하이퍼파라미터
  - 마지막 training 이 끝나면, 양쪽 system 의 sample 을 center c 에 임베딩할 수 있는 능력을 얻게 된다

- **Log Anomaly detection**
  - **decision boundary** 결정하기 : radius set 하기 → normal 과 anomaly 를 나누는 경계가 됨
    - radius γ 보다 distance 가 크면 anomalous sequence 로 판단
  - **각각의 system 에 validation set(**검증 셋) 을 둔다 → best radius 를 찾기 위해
    - validation set **:** 각 시스템의 one-day log sequence 로 둔다

$$\gamma^S \text{ and } \gamma^T$$

# 3. Experiments

# Experiments - Experimental Setup

Datasets.

**Table 1: 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 |

- 두 개의 supercomputer system 으로부터 얻은 dataset 사용
  1. BGL(Blue Gene/L) - Lawrence Livermore National Labs
  2. TB(Thunderbird) - Sandia National Labs

# Experiments - Experimental Setup

Datasets.

**Table 2: Statistics of Shared Words**

| | 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 |

→ BGL 과 TB 에서 공유되는 shared words 의 개수를 보여줌

ex.

- BGL 의 normal log sequence 의 unique word : 664
- TB 의 normal log sequence 의 unique word : 1753
- BGL, TB 의 normal log sequence 의 shared word : 254

# Experiments - Experimental Setup

Baselines.

5가지 anomaly detection 과 비교해서 LogTAD 의 성능을 측정

1. **PCA**(Principal Component Analysis) → a dimensional reduction based anomaly detection approach
2. **LogCluster** → clustering-based log anomaly detection approach
3. **Deeplog** → a deep learning based log anomaly detection
4. **DeepSVDD**(Deep support vector data description) → a deep learning based one-class classification model
5. **LogTransfer** → cross-system anomaly detection 이 가능하지만, to train a classifier 위해 두 시스템의 labeled data 가 필요

# Experiments - Experimental Setup

Implementation Details.

- sliding window (size :20) → short sequence 로 log file 을 자르기 위해
- Drain package → raw log message 에서 template 을 추출할 때 사용

모델

- LSTM encoder : ( 2 hidden layer with 128 dimensions ) i.e., $\mathbf{v}^{\epsilon} = G(\mathcal{X}^{\epsilon})$
- domain discriminator D : FCNN ( 2 hidden layer with 64 dimensions ) i.e., $D(\mathbf{v}^{\epsilon}) = \sigma(\mathbf{w}^{T}\mathbf{v}^{\epsilon} + b)$

training stage 에서 사용하는 normal sequence

- 100,000 개의 source system 의 normal sequence 사용
- 1,000 개의 target system 의 normal sequence 사용

adversarial training - objective function의 하이퍼파라미터 조정 → λ : 0.1

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

# Experiments - Experimental Results

**Table 3: Anomaly Detection on Source and Target Systems**

| | Method | Source | | Target | |
|---|---|---|---|---|---|
| | | F1 | AUC | F1 | AUC |
| BGL → TB | PCA w/ TB | 0.322 | 0.587 | 0.731 | 0.776 |
| | PCA w/o TB | 0.642 | 0.816 | 0.558 | 0.504 |
| | LogCluster w/ TB | 0.530 | 0.746 | 0.677 | 0.716 |
| | LogCluster w/o TB | 0.713 | 0.829 | 0.559 | 0.504 |
| | DeepLog w/ TB | 0.662 | 0.854 | 0.590 | 0.619 |
| | DeepLog w/o TB | 0.678 | 0.867 | 0.556 | 0.500 |
| | DeepSVDD w/ TB | 0.499 | 0.725 | 0.567 | 0.616 |
| | DeepSVDD w/o TB | 0.566 | 0.789 | 0.577 | 0.646 |
| | LogTransfer | 0.971 | 0.972 | 0.792 | 0.828 |
| | LogTAD | 0.926 | 0.964 | 0.758 | 0.804 |
| TB → BGL | PCA w/ BGL | 0.789 | 0.798 | 0.577 | 0.773 |
| | PCA w/o BGL | 0.760 | 0.779 | 0.229 | 0.658 |
| | LogCluster w/ BGL | 0.708 | 0.688 | 0.697 | 0.886 |
| | LogCluster w/o BGL | 0.724 | 0.716 | 0.223 | 0.500 |
| | DeepLog w/ BGL | 0.687 | 0.701 | 0.527 | 0.843 |
| | DeepLog w/o BGL | 0.660 | 0.677 | 0.223 | 0.500 |
| | DeepSVDD w/ BGL | 0.660 | 0.699 | 0.196 | 0.537 |
| | DeepSVDD w/o BGL | 0.794 | 0.808 | 0.195 | 0.497 |
| | LogTransfer | 0.995 | 0.995 | 0.788 | 0.833 |
| | LogTAD | 0.788 | 0.797 | 0.845 | 0.909 |

두 개의 시나리오 - F1score 와 AUC 측면에서 evaluate

1. BGL → TB : BGL 이 source, TB 가 target system
2. TB → BGL : TB 가 source, BGL 이 target system

AUC-ROC 곡선은 다양한 임계값에서 모델의 분류 성능에 대한 측정 그래프

- ROC(Receiver Operating Characteristic) = 모든 임계값에서 분류 모델의 성능을 보여주는 그래프
- AUC(Area Under the Curve) = ROC 곡선 아래 영역
  → AUC가 높다는 사실은 클래스를 구별하는 모델의 성능이 훌륭하다는 것을 의미

# LogTAD v.s. Unsupervised Anomaly Detection Approaches (PCA, LogCluster, DeepLog)

**Table 3: Anomaly Detection on Source and Target Systems**

| | Method | Source | | Target | |
|---|---|---|---|---|---|
| | | F1 | AUC | F1 | AUC |
| BGL → TB | PCA w/ TB | 0.322 | 0.587 | 0.731 | 0.776 |
| | PCA w/o TB | 0.642 | 0.816 | 0.558 | 0.504 |
| | LogCluster w/ TB | 0.530 | 0.746 | 0.677 | 0.716 |
| | LogCluster w/o TB | 0.713 | 0.829 | 0.559 | 0.504 |
| | DeepLog w/ TB | 0.662 | 0.854 | 0.590 | 0.619 |
| | DeepLog w/o TB | 0.678 | 0.867 | 0.556 | 0.500 |
| | DeepSVDD w/ TB | 0.499 | 0.725 | 0.567 | 0.616 |
| | DeepSVDD w/o TB | 0.566 | 0.789 | 0.577 | 0.646 |
| | LogTransfer | 0.971 | 0.972 | 0.792 | 0.828 |
| | LogTAD | 0.926 | 0.964 | 0.758 | 0.804 |
| TB → BGL | PCA w/ BGL | 0.789 | 0.798 | 0.577 | 0.773 |
| | PCA w/o BGL | 0.760 | 0.779 | 0.229 | 0.658 |
| | LogCluster w/ BGL | 0.708 | 0.688 | 0.697 | 0.886 |
| | LogCluster w/o BGL | 0.724 | 0.716 | 0.223 | 0.500 |
| | DeepLog w/ BGL | 0.687 | 0.701 | 0.527 | 0.843 |
| | DeepLog w/o BGL | 0.660 | 0.677 | 0.223 | 0.500 |
| | DeepSVDD w/ BGL | 0.660 | 0.699 | 0.196 | 0.537 |
| | DeepSVDD w/o BGL | 0.794 | 0.808 | 0.195 | 0.497 |
| | LogTransfer | 0.995 | 0.995 | 0.788 | 0.833 |
| | LogTAD | 0.788 | 0.797 | 0.845 | 0.909 |

PCA, LogCluster, and DeepLog 는 unsupervised models 이고 domain adaptation 을 위해 설계되지 않았으므로, 2가지 기준에 대해 평가

- **training with or without** using the log sequences from the **target system**.

**unsupervised models - performance**

1. target system 의 training samples 없이 source system 안에서는 양쪽 시나리오에서 괜찮은 성능 보임.
2. 그러나, training 에 target data 를 넣어서 학습(w)시키면 target system 에서는 괜찮은 성능을 보이지만 source system 에서는 안좋은 성능을 보임.

→ 기존 모델들은 시스템간 적응 능력 x, 데이터 섞으면 오히려 성능 저하
⇒ Using the log sequences from both source and target systems, normal samples 분포에 대해 혼동을 일으킴.

**LogTAD**

양쪽 시나리오(BGL → TB and TB → BGL ) 에서 source 와 target 둘 다 기존 모델들보다 성능이 잘 나옴.

# LogTAD v.s. DeepSVDD.

**Table 3: Anomaly Detection on Source and Target Systems**

| | Method | Source | | Target | |
|---|---|---|---|---|---|
| | | F1 | AUC | F1 | AUC |
| BGL → TB | PCA w/ TB | 0.322 | 0.587 | 0.731 | 0.776 |
| | PCA w/o TB | 0.642 | 0.816 | 0.558 | 0.504 |
| | LogCluster w/ TB | 0.530 | 0.746 | 0.677 | 0.716 |
| | LogCluster w/o TB | 0.713 | 0.829 | 0.559 | 0.504 |
| | DeepLog w/ TB | 0.662 | 0.854 | 0.590 | 0.619 |
| | DeepLog w/o TB | 0.678 | 0.867 | 0.556 | 0.500 |
| | DeepSVDD w/ TB | 0.499 | 0.725 | 0.567 | 0.616 |
| | DeepSVDD w/o TB | 0.566 | 0.789 | 0.577 | 0.646 |
| | LogTransfer | 0.971 | 0.972 | 0.792 | 0.828 |
| | LogTAD | 0.926 | 0.964 | 0.758 | 0.804 |
| TB → BGL | PCA w/ BGL | 0.789 | 0.798 | 0.577 | 0.773 |
| | PCA w/o BGL | 0.760 | 0.779 | 0.229 | 0.658 |
| | LogCluster w/ BGL | 0.708 | 0.688 | 0.697 | 0.886 |
| | LogCluster w/o BGL | 0.724 | 0.716 | 0.223 | 0.500 |
| | DeepLog w/ BGL | 0.687 | 0.701 | 0.527 | 0.843 |
| | DeepLog w/o BGL | 0.660 | 0.677 | 0.223 | 0.500 |
| | DeepSVDD w/ BGL | 0.660 | 0.699 | 0.196 | 0.537 |
| | DeepSVDD w/o BGL | 0.794 | 0.808 | 0.195 | 0.497 |
| | LogTransfer | 0.995 | 0.995 | 0.788 | 0.833 |
| | LogTAD | 0.788 | 0.797 | 0.845 | 0.909 |

- DeepSVDD 는 training dataset 에 **given target sequences** - target, source 에서 둘 다 낮은 성능(F1 score) 보임

  이유: mixed training data lead to diverse data distribution

  ⇒ 위의 분포에서 파생된 center 는 only use a small number of samples from the target system 때문에 anomalies 감지 능력 저하

- **without domain adaptation,**

  - the samples from the target systems 은 the source system 의 outliers 와 비슷함
    → the performance of DeepSVDD 도 source system 에서 좋지 않음

  - 따라서 domain adversarial adaptation 을 사용하면 문제 해결
    → samples have similar distribution

- **Another observation**

  - only using data from the source system 경우도 performance of DeepSVDD 안 좋음.
    → 이유: LogTAD derives the center by using more samples with the same distribution.

# LogTAD v.s. LogTransfer.

**Table 3: Anomaly Detection on Source and Target Systems**

| | Method | Source | | Target | |
|---|---|---|---|---|---|
| | | F1 | AUC | F1 | AUC |
| BGL → TB | PCA w/ TB | 0.322 | 0.587 | 0.731 | 0.776 |
| | PCA w/o TB | 0.642 | 0.816 | 0.558 | 0.504 |
| | LogCluster w/ TB | 0.530 | 0.746 | 0.677 | 0.716 |
| | LogCluster w/o TB | 0.713 | 0.829 | 0.559 | 0.504 |
| | DeepLog w/ TB | 0.662 | 0.854 | 0.590 | 0.619 |
| | DeepLog w/o TB | 0.678 | 0.867 | 0.556 | 0.500 |
| | DeepSVDD w/ TB | 0.499 | 0.725 | 0.567 | 0.616 |
| | DeepSVDD w/o TB | 0.566 | 0.789 | 0.577 | 0.646 |
| | LogTransfer | 0.971 | 0.972 | 0.792 | 0.828 |
| | LogTAD | 0.926 | 0.964 | 0.758 | 0.804 |
| TB → BGL | PCA w/ BGL | 0.789 | 0.798 | 0.577 | 0.773 |
| | PCA w/o BGL | 0.760 | 0.779 | 0.229 | 0.658 |
| | LogCluster w/ BGL | 0.708 | 0.688 | 0.697 | 0.886 |
| | LogCluster w/o BGL | 0.724 | 0.716 | 0.223 | 0.500 |
| | DeepLog w/ BGL | 0.687 | 0.701 | 0.527 | 0.843 |
| | DeepLog w/o BGL | 0.660 | 0.677 | 0.223 | 0.500 |
| | DeepSVDD w/ BGL | 0.660 | 0.699 | 0.196 | 0.537 |
| | DeepSVDD w/o BGL | 0.794 | 0.808 | 0.195 | 0.497 |
| | LogTransfer | 0.995 | 0.995 | 0.788 | 0.833 |
| | LogTAD | 0.788 | 0.797 | 0.845 | 0.909 |

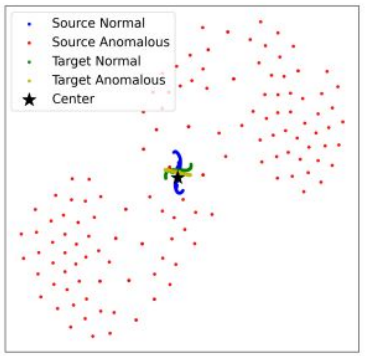
**LogTransfer** : supervised transfer learning method (the source and target systems 의 normal and anomalous data 를 채택하여 a cross-system anomaly detection classifier 훈련시킴)
→ 충분한 labeled data 를 사용할 수 있을 때 좋은 성능 달성

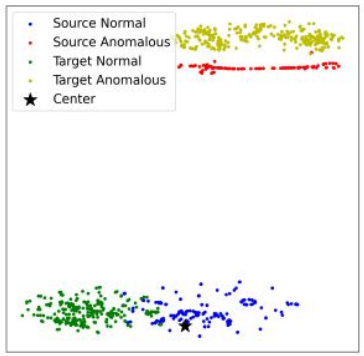
- **In this experiment,**
  - 목표: LogTAD 와 유사한 성능을 달성할 수 있도록 훈련하는데 how many labeled anomalous samples 필요한지 식별
  - train dataset
    - **100 anomalous sequences** from the source system
    - **10 anomalous sequences** from the target system
  - LogTransfer can achieve the best performance in the source system.

- **scenario BGL → TB,**
  LogTAD 보다 약간 더 나은 성능 달성

- **scenario TB → BGL,**
  using 10 anomalous sequences from the target system → LogTAD 보다 성능 향상 실패

# Visualization.




(a) Without domain adaptation (b) With domain adaptation

Figure 2: Log Sequence Visualization (BGL → Thunderbird)

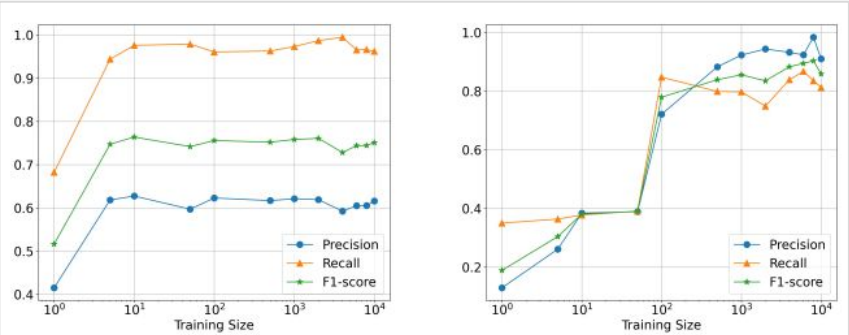
the effectiveness of our cross-system adaptation approach by visualizing sequence representations v.

1. sequence representations 을 2-dimensional space 에 매핑하기 위해 t-SNE algorithm 채택
2. For each dataset, 300 normal and anomalous sequences 랜덤 선택
3. BGL → TB 시나리오 사용

- **As shown in Figure 2(a),**
  normal sequences from the source and target systems 와
  anomalous sequences from the target system 가 섞여있음.

- **As shown in Figure 2(b),**
  normal sequences from both source and target systems ➡️ close to the center
  anomalous sequences from both systems ↔ the center region.

# Sample Size.




(a) BGL → Thunderbird (b) Thunderbird → BGL

**Figure 3: LogTAD performance with different training sizes from the target system. (x-axis is in log-scale)**

- **the scenario BGL → TB, 3(a)**
  only using 10 normal log sequences from the Thunderbird dataset → 좋은 성능 달성

- **the scenario TB → BGL, 3(b)**
  using around 100 samples can achieve reasonable performance

→ using a small number of normal sequences from the target system 함으로서 high anomaly detection performance on the target system 달성 가능

# 4. Conclusion

# Conclusion

LogTAD : It makes the normal log data in a **hypersphere** close to a center and utilizes the **domain adversarial adaptation** to make the log data from different systems follow similar distributions. (cross-system log anomaly detection framework)

- normal log data 를 구의 중심에 가깝게 하고 domain adversarial adaptation 을 활용하여 서로 다른 system 의 log data 들도 비슷한 분포를 갖도록 만듦.
- 이점: large distances to the center 로 different 시스템에서의 anomaly detection 가능

- 한계점 : source 와 target system 이 share similar patterns in terms of descriptive words 라는 가정
  → 앞으로 이 가정을 일반화 할 계획