Unsupervised Contextual Anomaly Detection For Database Systems

Data Engineering Lab

Sunghyun Ahn
skd@yonsei.ac.kr

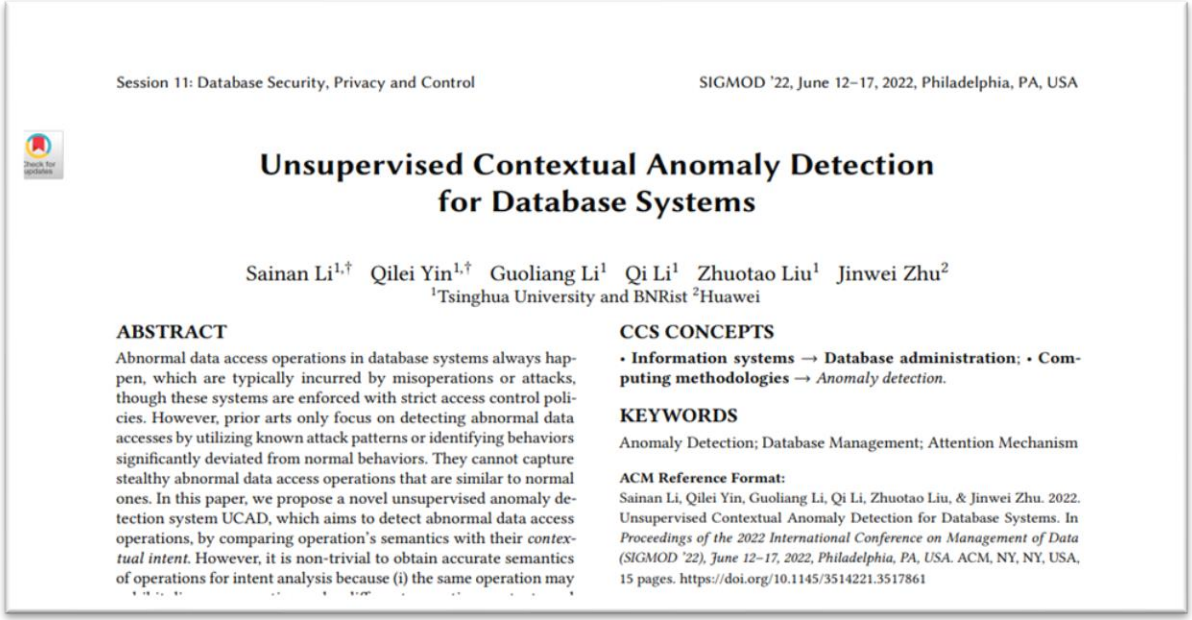
<2023/05/09>

Contents

1 Introduction

Unsupervised Contextual Anomaly Detection For Database Systems

- 2022년 SIGMOD 학회에 게재된 논문
- 데이터베이스에서 이상 쿼리문을 탐지하는 딥 러닝 모델 개발, Tsinghua University & Huawei
- Stealthy abnormal operation에도 강인한 이상치 탐지, real-word data에서 baseline 성능을 능가함



Unsupervised Contextual Anomaly Detection for Database Systems

Sainan Li[1,†] Qilei Yin[1,†] Guoliang Li[1] Qi Li[1] Zhuotao Liu[1] Jinwei Zhu[2]
[1]Tsinghua University and BNRist [2]Huawei

ABSTRACT

Abnormal data access operations in database systems always happen, which are typically incurred by misoperations or attacks, though these systems are enforced with strict access control policies. However, prior arts only focus on detecting abnormal data accesses by utilizing known attack patterns or identifying behaviors significantly deviated from normal behaviors. They cannot capture stealthy abnormal data access operations that are similar to normal ones. In this paper, we propose a novel unsupervised anomaly detection system UCAD, which aims to detect abnormal data access operations, by comparing operation's semantics with their *contextual intent*. However, it is non-trivial to obtain accurate semantics of operations for intent analysis because (i) the same operation may

CCS CONCEPTS

• **Information systems** → **Database administration**; • **Computing methodologies** → *Anomaly detection.*

KEYWORDS

Anomaly Detection; Database Management; Attention Mechanism

ACM Reference Format:

Sainan Li, Qilei Yin, Guoliang Li, Qi Li, Zhuotao Liu, & Jinwei Zhu. 2022. Unsupervised Contextual Anomaly Detection for Database Systems. In *Proceedings of the 2022 International Conference on Management of Data (SIGMOD '22), June 12–17, 2022, Philadelphia, PA, USA.* ACM, NY, NY, USA, 15 pages. https://doi.org/10.1145/3514221.3517861

1 Introduction

데이터베이스 보안 위협

Privilege Abuse

→ 권한을 남용하여 데이터베이스에 대한 불법적인 액세스를 시도하는 것,
ex) 은행에서 권한을 가진 직원이 개인적인 금전적 이득을 얻기 위해 은행 고객들의 개인정보를 불법적으로 조회함

Credential Stealing

→ 합법적인 사용자의 자격 증명 정보(ID 및 PW)를 탈취하여 데이터베이스에 액세스하는 것,
ex) 피싱, 스팸 메일 또는 악성 코드와 같은 방식으로 정보 탈취 -> 사용자가 매일 수행하는 일반적인 작업들 사이에 비정상적인 데이터 삭제 작업을 숨겨서 수행함 (Stealthy abnormal operation)

Misoperations

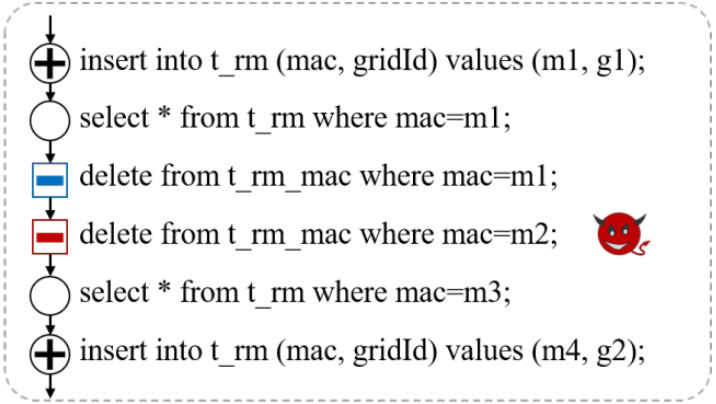
→ 경험이 부족한 직원들이 실수로 잘못된 작업을 수행하여 데이터 카오스(예: 데이터 유출)를 초래하는 것,
정해진 규칙에 따라서 수행되는 일반적인 작업과 달리 논리적으로 일관성이 없음

1 Introduction

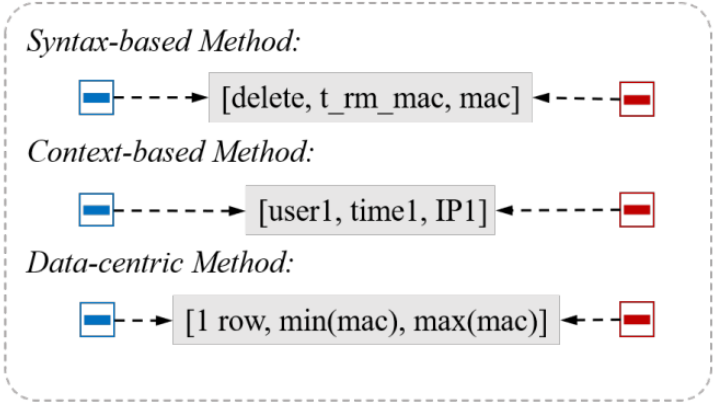
Stealthy abnormal operation

- 일반적인 작업들 사이에 비정상적인 데이터 삭제 작업을 숨겨서 수행하는 것
- Syntax-based Method: SQL 문장의 구문을 분석하여 이상한 패턴이 있는지를 찾는 방법
- Context-based Method: 데이터베이스 접근 도중 생성된 시스템 및 사용자 정보를 활용하여 정상적인 패턴에서 벗어나는 이상 동작을 감지하는 방법
- Data-based Method: 자주 변경되는 데이터의 통계(최댓값, 최솟값)를 사용하여 데이터 변경으로 인한 이상 동작을 감지하는 방법

→ 기존 방식들은 잘 알려진 공격이나 비정상적인 데이터 접근에 초점이 맞춰져서 Stealthty abnormal operation을 구분 못 함



(a) A session having one stealthy abnormal operation.

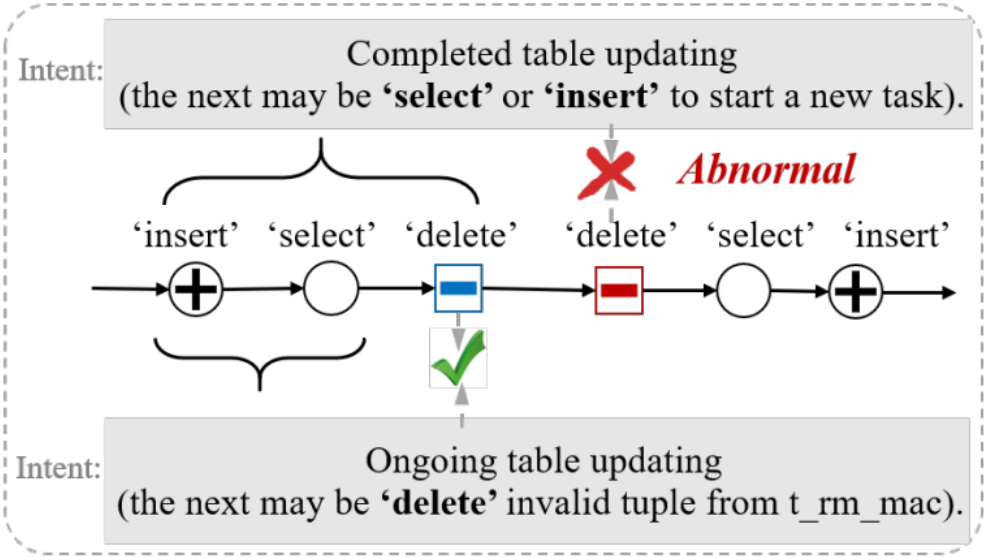


(b) The limitation of traditional detection methods.

1 Introduction

Comparing the semantics with the contextual intent

- 다음에 발생할 쿼리문의 의도(intent)와 실제 발생되는 쿼리문의 의미(semantics)를 비교하는 방법
- 다음에 발생할 쿼리문의 의도는 이전에 발생된 쿼리문들로부터 예측되므로 contextual intent로 표현함
- contextual intent와 semantic에 대한 차이가 발생하면 Abnormal이라고 간주함



1 Introduction

Three Challenges for capturing the semantics and contextual intent

- 같은 operation이라도 다른 의미를 지닐 수 있음
  - ex) select 이후 delete는 '선택한 것을 삭제'한다고 해석 가능하지만, 아무 맥락없는 delete는 단순히 '삭제'의 의미를 지님
  - → 흔한 semantic extraction apporoch (e.g. word embedding)은 적합하지 않음

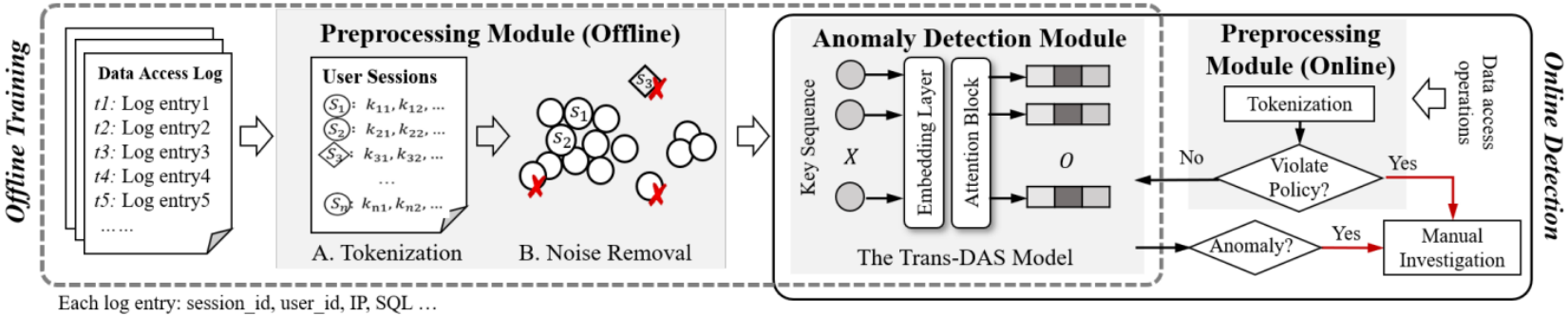
- 다른 operation 순서를 지닌다고 하더라도 같은 의미를 지닐 수 있음
  - ex) select 이후 delete한 것과 delete 이후 select한 것은 같은 의미를 지닐 수 있음
  - → LSTM과 같이 순서 정보를 고려한 semantic extraction 모델은 적합하지 않음 -> Bidrectional한 Transformer를 사용함

- 노이즈 데이터가 의미 해석에 악 영향을 줄 수 있음
  - ex) accidental misoperations (not necessarily malicious) – 일관성이 없는 쿼리
  - → Preprocessing Module을 개발해서 노이즈 데이터를 먼저 제거할 필요가 있음

2 Architecture

UCAD (Unsupervised Contextual Anomaly Detection)

- Preprocessing Module과 Anomaly Detection Module로 구성되어 있음
- Preprocessing Module은 User Sessions($s_1, s_2, \cdots, s_n$) 각각을 토큰화하고 노이즈를 제거함
- Anomaly Detection Module은 각 세션에 담긴 키($k_1, k_2, \cdots, k_l$)들을 입력으로 받아 semantic extraction을 함 (Trans-Das)
- Trans-Das라는 Transformer 모델을 설계해서 future key semantics를 예측함 → ($k_1, k_2, \cdots k_l$) -> ($k_{2'}, k_{3'}, \cdots, k_{l+1'}$)
  ex) 첫 번째 노드의 출력, $k_{2'}$는 이전 키(k_1)와 이후 키들($k_3, \cdots, k_l$)의 semantics로 획득한 future key semantics (target은 k_2)

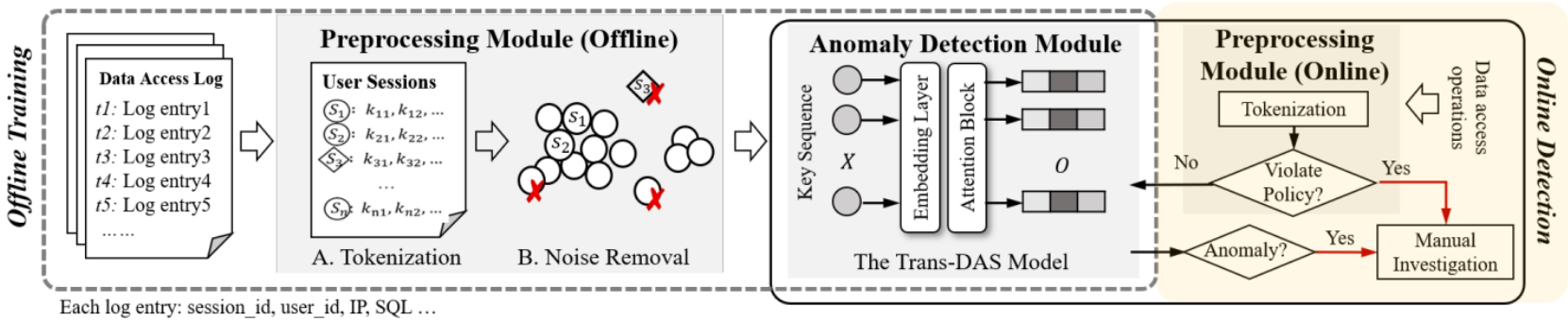


Session: 특정 사용자가 데이터베이스에 접근하는 동안 실행하는 데이터 접근 작업

2 Architecture

Anomaly Detection

- 각 세션에 담긴 operation들을 토큰화하고, 토큰 중에 정책에 위반되는 잘 알려진 공격이 존재하면 Anomaly라고 간주
- 현재 토큰, x_t의 Anomaly 여부를 탐지하기 위해 $x_{t-L},\cdots,x_{t-1}$을 Trans-DAS에 입력 ->
  출력으로 나온 x_t'을 Contextual intent로 지정 -> Contextual intent와 x_t를 비교해서 차이가 크면 Anomaly라고 간주
  (Contextual intent: 다음에 나올 쿼리문의 의미와 의도)



2 Architecture

Anomaly Detection

- 현재 토큰, x_t의 Anomaly 여부를 탐지하기 위해 $x_{t-L},\cdots,x_{t-1}$을 Trans-DAS에 입력 ->
  출력으로 나온 $x_{t'}(O_L^{(B)})$을 Contextual intent로 지정 -> Contextual intent와 x_t를 비교해서 차이가 크면 Anomaly라고 간주
  - → 좋은 효과를 발휘하지 못 함 (유저 패턴이 너무 다양하므로 예측된 $x_{t'}$가 정답 x_t와 일치하지 않는 경우가 다수 발생)

- 위 문제를 해결하기 위해, 출력인 $O_L^{(B)}$와 입력인 $\{x_{t-L,\cdots},x_{t-1}\}$의 유사도를 계산해서 랭킹(Top-p)을 매김 ->
  $O_L^{(B)}$와 x_t의 유사도가 랭킹 안에 속한다면 Normal로 간주, 그렇지 않다면 Anomaly로 간주

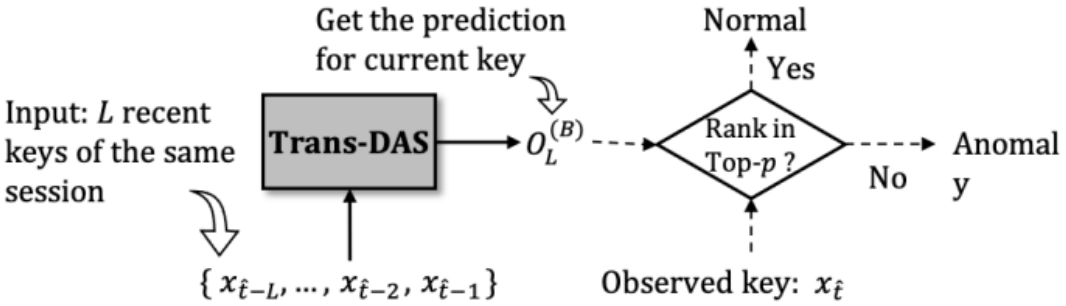


Figure 5: The detection stage of Trans-DAS model.

3 Method

Noise Removal

attribute-based access control policies

속성 기반 접근 제어 정책을 활용하여 데이터 접근 중 알려진 공격 패턴을 직접 필터링한다
ex) 사용자 ID와 주소 속성을 사용하여 이전에 알려지지 않은 주소에서 DB에 접근했다면 노이즈로 판단한다.

Clustering

(1) n-gram features를 사용하여 각 세션을 프로파일링한다
(2) Jaccard Index를 사용하여 세션 간의 유사도를 계산한다
(3) DBSCAN을 이용하여 밀도 기반 클러스터링을 진행한다

Post Processing

(1) 클러스터에서 언더샘플링을 수행한다 (샘플링 비율은 모든 클러스터의 중앙값으로 결정됨)
(2) 중앙값보다 사이즈가 작은 클러스터는 제거한다
(3) 각 클러스터의 세션 평균 길이(키의 개수)를 확인하고, 평균보다 작은 세션을 제거한다

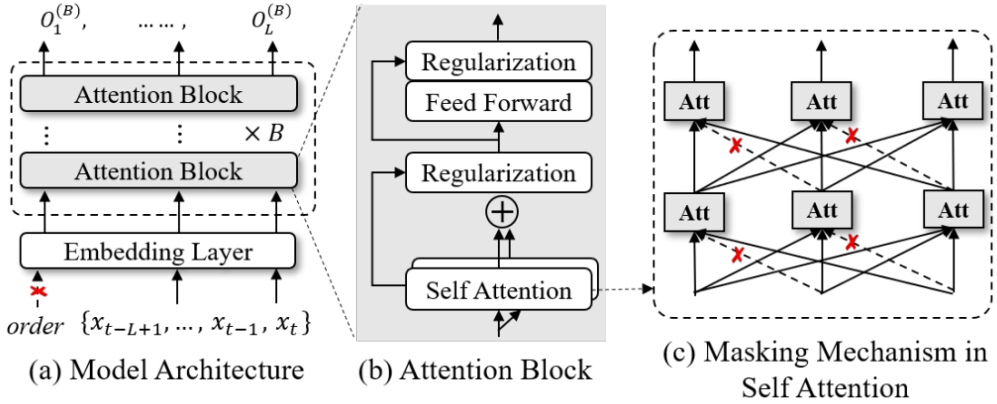
작은 클러스터에서 발생하는 패턴이 노이즈로 판단되는 것을 방지,
잘 발생하지 않은 드문 패턴(accidental misoperations 등) 제거,
문맥적 의도를 해석하기에는 너무 짧은 세션을 제거,
Trans-Das는 정상 데이터만으로 학습이 가능 (Unsupervised Training)

3 Method

Trans-Das (Transformer for Data Access Semantics)

- order information 제거
  - 일반적인 Transformer와 달리 Embedding 과정에서 Positional Encoding을 수행하지 않는다
    DB 사용자의 패턴이 워낙 다양하기 때문에 순서 정보를 학습하는 것은 오히려 악영향을 주기 때문이다

- 새로운 Masking Mechanism
  - Transformer Decoder의 Masking은 다음 단어를 예측하기 위해 이전 단어 정보만을 사용하도록 한다
    DB에서 발생하는 쿼리는 모든 쿼리와 연관성을 지니므로, 예측할 쿼리에 대해서만 마스킹을 해준다



(a) Model Architecture (b) Attention Block (c) Masking Mechanism in Self Attention

3 Method

Training Method

- Future key semantics를 예측하기 위해 output인 O(B)와 Target인 x간의 유사도를 높이는 방향으로 학습
- Triplet Loss와 Cross Entropy, L2 Norm을 결합한 목적함수를 제작함
- Triplet Loss를 통해 학습하면, 두 개의 임베딩이 서로 다른 라벨을 가지면 그 거리를 최대화하고(유사도를 낮춤), 같은 라벨을 가지면 그 거리를 최소화(유사도를 높임)하여 임베딩 간의 차이를 극대화할 수 있음
- 다른 라벨을 지닌 임베딩(x_1',⋯,x_L')을 이용하기 위해 Negative Sampling을 진행하였음

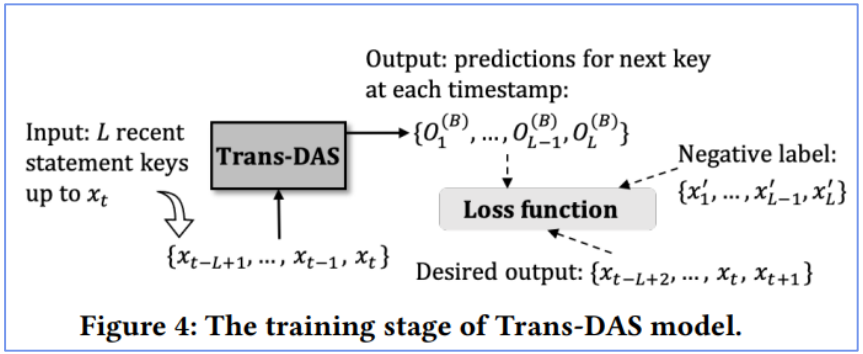


Figure 4: The training stage of Trans-DAS model.

$$z_i^j = Sigmoid\left(O_i^{(B)} \cdot \mathbf{M}\left(x_j\right)\right), \tag{10}$$

$$\mathcal{L} = \sum_{\mathcal{T}} \sum_{i=1}^{L} \max\left(z_i^- - z_i^+ + g, 0\right) - \log\left(z_i^+\right) + ||\theta||_2, \tag{11}$$

M: 임베딩을 위한 행렬 (LxH)
(10): 유사도 계산 공식
z^-: 서로 다른 라벨을 가진 두 개의 임베딩 간의 유사도
z^+: 서로 같은 라벨을 가진 두 개의 임베딩 간의 유사도

4 Experiments

Dataset

- 두 가지 시나리오에 대한 데이터셋을 수집함
  (1) 온라인 댓글 어플리케이션 (삽입, 삭제, 업데이트 작업을 많이 수행함)
  (2) 위치 서비스 어플리케이션 (선택 작업을 주로 많이 수행함)

- 각 시나리오에 대해 V1, V2, V3 테스트 데이터를 제작함 (서로 다른 사용자의 패턴에도 강인한지 테스트)
  (V1) 전처리 모듈을 통해 정제된 데이터셋 중 20% <80%는 훈련에 사용>
  (V2) V1 데이터셋의 각 세션에서 키를 부분적으로 교환
  (V3) V1 데이터셋의 각 세션에서 키를 부분적으로 삭제

- 각 시나리오에 대해 A1, A2, A3 테스트 데이터를 제작함 (데이터베이스 보안 위협을 잘 감지하는지 테스트)
  (A1) Privilege Abuse에 대한 데이터를 제작하기 위해 비정상적으로 많은 Select문이 담긴 세션을 생성함 <V1에 추가>
  (A2) Credential Stealing에 대한 데이터를 제작하기 위해 무작위로 Delete문이 담긴 세션을 생성함 <V1에 추가>
  (A3) Misoperations에 대한 데이터를 제작하기 위해 일관성이 없는 작업들이 담긴 세션을 생성함 <V1에 추가>

4 Experiments

Results

두 시나리오에 대한 성능 결과

제안 모델의 FPR과 FNR이 V1, V2, V3와 A1, A2, A3에 대해 평균적으로 낮은 수치를 보임,
Precision, Recall, F1 Score 같은 경우 baseline 모델들보다 높은 수치를 보임

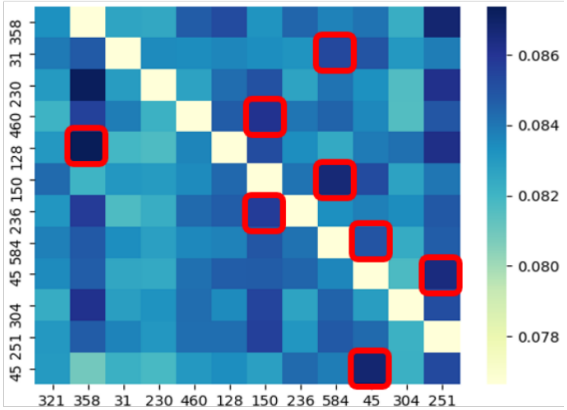
Scenario	Methods	FPR			FNR			P	R	F1
		$\mathcal{V}1$	$\mathcal{V}2$	$\mathcal{V}3$	$\mathcal{A}1$	$\mathcal{A}2$	$\mathcal{A}3$			
I Commenting Application	OneClassSVM[67]	0.02247	0.02247	0.02247	0.0494	0.75281	0.0	**0.97029**	0.73408	0.83582
	iForest [48]	0.26966	0.26966	0.22472	0.20225	0.19101	0.0	0.77333	0.86891	0.81834
	Mazzawi et al. [52]	0.05618	0.05618	0.07865	0.44944	1.0	0.0	0.89032	0.51685	0.65403
	DeepLog [21]	0.38202	0.57303	0.38202	0.21348	0.01124	0.0	0.67486	0.92509	0.78041
	USAD [11]	0.22472	0.20225	0.30337	0.08989	0.34831	0.0	0.77816	0.85393	0.81429
	Ours	0.12360	0.15730	0.14607	0.19101	0.02247	0.0	0.86713	**0.92884**	**0.89693**
II Location Service	OneClassSVM[67]	0.14475	0.13226	0.01613	0.0	0.84194	0.0	0.88609	0.71935	0.79407
	iForest [48]	0.03619	0.03226	0.02258	0.5	0.08925	0.0	0.96513	0.80358	0.87698
	Mazzawi et al. [52]	0.00844	0.01505	0.02043	0.44086	0.99247	0.55914	0.95223	0.33584	0.49656
	DeepLog [21]	0.34861	0.75591	0.69677	0.0	0.16022	0.0	0.61691	0.94659	0.74699
	USAD [11]	0.18938	0.26667	0.17097	0.0	0.34839	0.0	0.81386	0.88387	0.84742
	Ours	0.04222	0.03871	0.03118	0.0	0.00430	0.0	**0.96535**	**0.99857**	**0.98168**

4 Experiments

Results

어텐션 맵 시각화 결과

(Key:358)와 (Key:128)는 높은 연관성을 지닌 것으로 확인됨,
전문가가 Statement를 확인해봤더니 t_cell_fp_9 테이블에서 연속되는 쿼리였다고 함
(Key:460)과 (Key:150)도 마찬가지



T	Key	Statement
t1	321	INSERT INTO t_cell_fp_9(pnci, gridId, fps) VALUES ($1, $2, $3)
t2	358	SELECT * FROM t_cell_fp_9 WHERE pnci=$1 and gridId IN ($2, ... $36)
t3	31	INSERT INTO t_cell_fp_3(pnci, gridId, fps) VALUES ($1, $2, $3)
t4	230	SELECT * FROM t_cell_fp_3 WHERE pnci=$1 and gridId IN ($2, ... $13)
t5	460	SELECT * FROM t_cell_fp_3 WHERE pnci=$1 and gridId IN ($2, ... $9)
t6	128	INSERT INTO t_cell_fp_9(pnci, gridId, fps) VALUES ($1, $2, $3) ...($31, $32, $33)
t7	150	SELECT * FROM t_cell_fp_3 WHERE pnci=$1 and gridId IN ($2, ... $16)

4 Experiments

Results

Base Transformer와의 비교

제안한 Embedding Layer, Masking mechanism, training objective를 Base Transformer에 적용한 결과, Precision과 F1-Score에서 높은 성능 향상을 보였음

Scenario	Model Variants	FPR			FNR		P	R	F1
		$\mathcal{V}1$	$\mathcal{V}2$	$\mathcal{V}3$	$\mathcal{A}1$	$\mathcal{A}2$			
I Commenting Application	Base Transformer	0.20225	0.22472	0.21348	0.19101	0.02247	0.81311	0.92884	0.86713
	Our embedding layer	0.17977	0.19101	0.20225	0.20225	0.02247	0.82886 ↑	0.92509	0.87434 ↑
	Our masking mechanism	0.16265	0.16725	0.15791	0.19101	0.02247	0.84360 ↑	0.92884	0.88417 ↑
	Our training objective	0.15730	0.11236	0.13483	0.22472	0.02247	0.87189 ↑	0.91760	0.89416 ↑
	Trans-DAS	0.12360	0.15730	0.14607	0.19101	0.02247	0.86713 ↑	0.92884	**0.89693** ↑
II Location Service	Base Transformer	0.09530	0.09140	0.08602	0.0	0.00538	0.91945	0.99821	0.95721
	Our embedding layer	0.09771	0.09677	0.09570	0.0	0.00538	0.91461	0.99821	0.95458
	Our masking mechanism	0.06996	0.06452	0.05699	0.0	0.00215	0.94221 ↑	0.99928 ↑	0.96991 ↑
	Our training objective	0.03257	0.03763	0.03763	0.04624	0.03333	0.96550 ↑	0.97312	0.96930 ↑
	Trans-DAS	0.03860	0.04194	0.03226	0.0	0.00322	0.96535 ↑	0.99857 ↑	**0.98168** ↑

Thank You

Data Engineering Lab

Sunghyun Ahn
skd@yonsei.ac.kr

<2023/05/09>