LEVERAGING LARGE LANGUAGE MODELS FOR SQL BEHAVIOR-BASED DATABASE INTRUSION DETECTION

A PREPRINT

Meital Shlezinger*
Huawei Tel Aviv Research Center
Tel Aviv, Israel
mshlezinger@gmail.com

Shay Akirav
Huawei Tel Aviv Research Center
Tel Aviv, Israel

Lei Zhou
Huawei Tel Aviv Research Center
Tel Aviv, Israel

Liang Guo
Huawei Technologies Co. LTD Shenzhen
Guangdong, China

Avi Kessel
Huawei Tel Aviv Research Center
Tel Aviv, Israel

Guoliang Li
Tsinghua University
Beijing, China

August 11, 2025

ABSTRACT

Database systems are extensively used to store critical data across various domains. However, the frequency of abnormal database access behaviors, such as database intrusion by internal and external attacks, continues to rise. Internal masqueraders often have greater organizational knowledge, making it easier to mimic employee behavior effectively. In contrast, external masqueraders may behave differently due to their lack of familiarity with the organization. Current approaches lack the granularity needed to detect anomalies at the operational level, frequently misclassifying entire sequences of operations as anomalies, even though most operations are likely to represent normal behavior. On the other hand, some anomalous behaviors often resemble normal activities, making them difficult for existing detection methods to identify. This paper introduces a two-tiered anomaly detection approach for Structured Query Language (SQL) using the Bidirectional Encoder Representations from Transformers (BERT) model, specifically DistilBERT, a more efficient, pre-trained version. Our method combines both unsupervised and supervised machine learning techniques to accurately identify anomalous activities while minimizing the need for data labeling. First, the unsupervised method uses ensemble anomaly detectors that flag embedding vectors distant from learned normal patterns of typical user behavior across the database (out-of-scope queries). Second, the supervised method uses fine-tuned transformer-based models to detect internal attacks with high precision (in-scope queries), using role-labeled classification, even on limited labeled SQL data. Our findings make a significant contribution by providing an effective solution for safeguarding critical database systems from sophisticated threats.

Keywords Database Security · Anomaly Detection · Machine Learning · LLM

1 Introduction

In recent years, database security has made significant advancements, driven by the increasing reliance on databases and the threat of targeted attacks. Data breaches and malicious activities have raised significant security and privacy concerns for organizations, emphasizing the critical need for robust database security. The 2023 Tesla data leak, which compromised the personal data of 75,000 of the company's employees, is a stark reminder of these vulnerabilities [1]. Tesla attributed the breach to 2 former employees, highlighting the importance of implementing more effective security measures. Protecting databases from leakage of sensitive information requires addressing internal and external threats.

*Corresponding author.

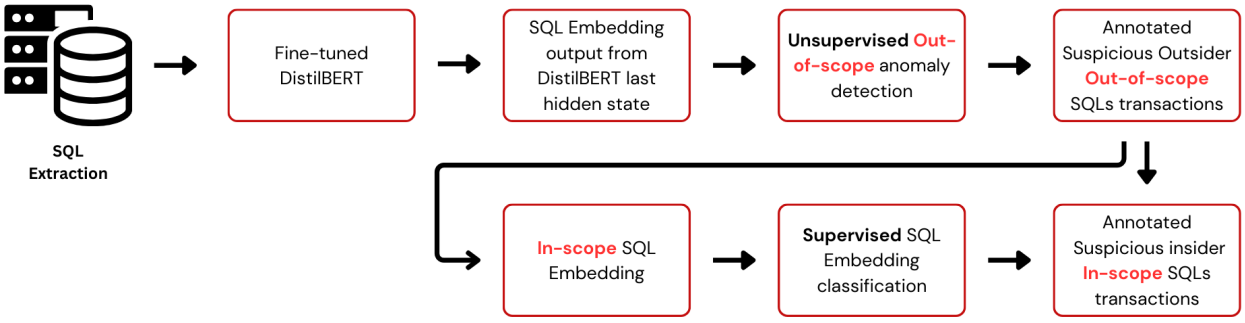

Figure 1: An overview of a two-tier anomaly detection framework composed of unsupervised and supervised approaches for external (out-of-scope queries) and internal (in-scope queries) database attacks.

Insider threats occur when legitimate system users abuse their access privileges to steal or leak sensitive data. In contrast, external threats originate from attackers outside the organization who exploit network or system vulnerabilities to gain access to the organization. Insider threats are particularly concerning, as they involve trusted individuals who have permission to access various data and services. According to the Ponemon Institute's 2023 report "The Cost of Insider Risk," 71% of the companies surveyed reported experiencing between 21 and over 40 insider incidents annually, a 4% increase from 2022 [2].

Traditional security measures, such as authentication, role-based access control, data encryption, and physical security, provide a foundational level of protection. However, protecting databases from legitimate system users abusing their access privileges poses a continuous challenge, emphasizing the need for effective security controls to mitigate insider threats. One particularly concerning type of attack is a masquerade attack, where an attacker uses stolen credentials to impersonate a legitimate employee and gain unauthorized access to resources, including databases. Masquerade attacks can occur in 2 ways: (i) an insider gains control of another employee's credentials with a different privilege level, or (ii) an outsider obtains a legitimate employee's credentials. Detecting such attacks requires specialized techniques like masquerade anomaly detection [3]. This approach aims to identify unauthorized users by analyzing deviations in user behavior between transactions, which may indicate an attacker's presence.

A significant body of literature has been dedicated to anomaly detection in database systems, particularly within Structured Query Language (SQL). These detection methods can be divided into 3 primary categories: syntax-based techniques [4, 5], context-based methods [6, 7, 8, 9], and data statistics-based approaches [10, 11]. Despite their contributions, these methods have notable limitations. One major drawback is their inability to fully capture SQL's underlying structure and syntax, often resulting in a relatively high rate of false positives. Natural Language Processing (NLP) has introduced new possibilities for addressing complex security challenges. In particular, transformer-based models, such as Bidirectional Encoder Representations from Transformers (BERT) [12], Large Language Model Meta AI (LLaMA), and LLaMA 2 [13, 14], have achieved state-of-the-art (SOTA) performance in various NLP tasks. These models can enhance computer security by enabling the development of more effective and adaptable anomaly detection systems capable of learning from large-scale and diverse data sources.

Detecting insider threats presents considerable challenges, particularly due to limitations in threat data availability and quality. Recent surveys indicate that advanced deep learning models, such as Long Short-Term Memory (LSTM) networks and transformer-based models, can address some of these data issues using techniques like anomaly detection [15]. Transformer-based models, known for their capability to handle large datasets and manage long-range

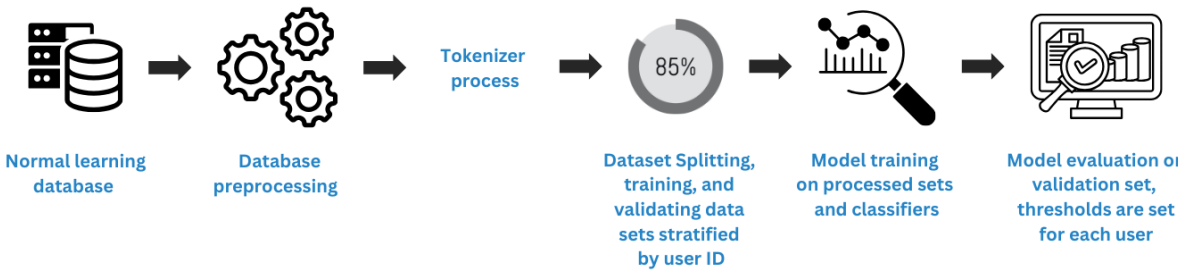

Figure 2: Supervised method – Learning period flow chart.

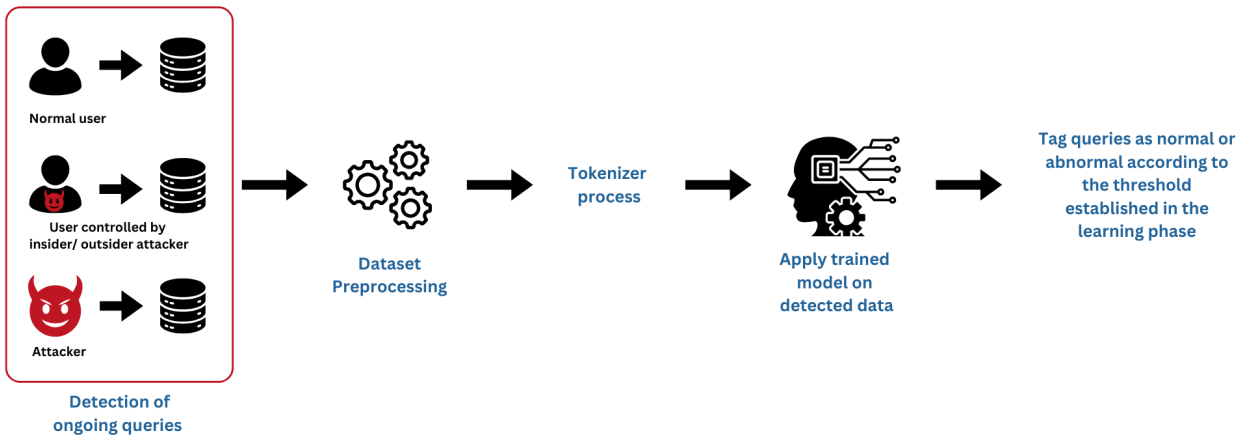

Figure 3: Supervised method - Detection period flow chart.

dependencies through multi-head attention, provide a promising alternative to traditional machine learning models. DistilBERT [16], a lighter and more efficient version of BERT [12], has shown exceptional performance across various NLP tasks. By fine-tuning a DistilBERT model on a SQL dataset, we can capture the underlying structure and syntax of SQL. This enables the model to detect deviations in users' SQL commands from normal activity with typical patterns. Consequently, it is crucial to assess whether these methods can accurately identify specific anomalous operations and report them to system operators for further action.

Our proposed system transcends the limitations of conventional rule-based approaches by harnessing the adaptive capabilities of machine learning to address evolving threats. This is achieved by applying a Large Language Model (LLM), a deep learning algorithm capable of performing various NLP tasks. LLMs, which use transformer models trained on large datasets, can recognize, translate, predict, or generate text or other content based on the attention and context of words. The proposed system aims to analyze query patterns and user behaviors in real time, enabling the early detection of anomalous activities indicative of malicious intent. Our current research primarily focuses on employing machine learning techniques to understand the semantics and context of SQL transactions. We aim to accurately detect anomalies and enhance overall system security using LLMs.

The main contributions of this paper are as follows: 1. We implement a two-tier anomaly detection framework for SQL that (i) utilizes the pre-trained DistilBERT model and ensemble anomaly detectors to address a significant issue in database security, including user behavior for external attacks (out-of-scope queries), and (ii) uses a role-labeled classification method to detect internal attacks by transformer-based models (see Fig. 1). 2. A role (user)-labeled classification system leveraging detailed behavioral profiles across several distinct roles (see Figs. 2-3), assuming all queries for all roles are normal in the learning period, while in the detection phase, the system identifies abnormal behavior using probabilistic embedding thresholds. This approach enables fine-grained detection of internal masqueraders, going beyond the binary 'Normal/Abnormal' labeling used in prior work. 3. In the detection phase, for internal threats, our supervised model identifies anomalies in two ways: (i) when a query is most likely associated with a different user than the user it was labeled with, and (ii) when a query matches the correct user but its probability score falls below the user's learned threshold (see Section 5.2). This strategy enables the detection of two distinct types of internal masquerade attacks. 4. We assess the performance of the supervised fine-tuned models on a few-shot set of labeled SQL data, emphasizing the adaptability and accuracy of our method.

Fig. 1 describes the general pipeline of the two-tier approach to detecting external and internal database attacks by using both unsupervised and supervised approaches. The rest of this paper is structured as follows: Section 2 reviews related work in SQL anomaly detection; Section 3 describes the data, including dataset details and data cleaning methods; Section 4 outlines our methodology, covering both unsupervised and supervised approaches; Section 5 presents the results along with examples of anomaly activities; and Section 6 concludes the paper.

2 Related Work

2.1 Database Anomaly Detection

Various methods have been suggested for identifying anomalies in databases, which can be categorized into 3 primary approaches: (i) syntax-based techniques, which principally use the syntax of SQL statements to pinpoint anomalies [5, 4]; (ii) context-based methods, which consider contextual features while modeling and learning patterns of normal behavior to detect deviations [6, 7, 8, 9]; and (iii) data statistics-based methods which identify anomalies by observing significant data changes caused by operational behavior [10, 11]. Additionally, some hybrid approaches combine elements from these categories (syntax-based, context-based, and data statistics-based) to enhance detection accuracy [9, 7]. Despite the advancements in anomaly detection techniques, traditional methods still face significant challenges. Specifically, these challenges include (i) an Inability to capture the complete structure and syntax of the SQL; (ii) difficulties in distinguishing between anomalous and normal behavior, especially when queries are similar but not identical, requiring advanced sentence processing to capture contextual information. For example: "select $*$ from employees where depid = ?" compared to "select $*$ from employees where depid = ? and managerid = ?", both queries have similar meaning, but they are not identical; and (iii) limitations in root cause analysis as most approaches fail to differentiate between external and internal attack methods.

Table 1: Supervised method output - Users Probability Matrix, the highest probability for each row (query) marked in bold red, and the significant probabilities marked in non-bold red, attributed to a specific user (labeled from 0 to 10).

	0	1	2	3	4	5	6	7	8	9	10
1	0.004128	0.003537	**0.956105**	0.005157	0.005614	0.005391	0.003488	0.003129	0.00824	0.003558	0.001616
2	0.006137	0.006842	0.004977	0.003585	0.003422	**0.936215**	0.005186	0.00384	0.009933	0.009418	0.010445
3	0.007922	0.003449	0.017589	0.004932	0.013318	0.008284	0.922157	0.007796	0.007715	0.003756	0.003101
4	0.000132	7.41E-05	5.91E-05	5.37E-05	0.000115	0.000122	0.000145	0.000994	0.000153	0.000468	**0.997685**
5	0.000785	0.002707	0.001228	**0.988784**	0.001185	0.001504	0.000663	0.000595	0.001566	0.000676	0.000307
6	0.000651	0.000365	0.000291	0.000265	0.000565	0.000601	0.000715	0.004897	0.000756	0.933976	0.056918
7	0.001395	0.0005	0.000629	0.000567	0.001362	0.001459	0.001522	0.981272	0.002977	0.002176	0.006141
8	0.966004	0.011226	0.004079	0.006021	0.007439	0.000944	0.001278	0.001049	0.001038	0.000505	0.000417
9	0.004952	0.003528	0.942357	0.005695	0.009944	0.009494	0.004185	0.003753	0.009885	0.004268	0.001939
10	0.008037	0.002881	0.004841	0.0041	0.013512	0.008404	**0.935533**	0.007909	0.007827	0.003811	0.003146
11	0.009278	0.004265	0.008231	0.004734	0.008598	0.009702	0.00841	0.021829	0.835873	0.08545	0.003632
12	0.001284	0.00072	0.000574	0.000521	0.001114	0.001184	0.001409	**0.98216**	0.00149	0.003892	0.005652
13	0.001291	0.000463	0.001145	0.000658	0.00217	0.00135	0.001408	0.980904	0.002754	0.002013	0.005846
14	0.002027	0.80957	0.007335	0.164282	0.003062	0.003887	0.001713	0.001536	0.004047	0.001747	0.000794
15	0.00041	0.000178	0.000501	0.000203	0.0004	0.000428	0.000447	0.003106	0.000874	0.000692	0.99276
16	0.00041	0.00023	0.000183	0.000166	0.000355	0.000378	0.00045	0.003106	0.000475	0.001581	0.992665
17	0.002728	0.003041	0.001954	0.778131	0.001521	0.198728	0.002305	0.001922	0.004415	0.004186	0.001068
18	0.00041	0.000178	0.000501	0.000203	0.0004	0.000428	0.000447	0.003106	0.000874	0.000692	0.99276
19	0.000132	7.41E-05	5.91E-05	5.37E-05	0.000115	0.000122	0.000145	0.000994	0.000153	0.000467	**0.997685**
20	0.010387	**0.913162**	0.006296	0.008674	0.09438	0.00811	0.008777	0.007319	0.00783	0.01594	0.004066
21	0.00041	0.00023	0.000183	0.000166	0.000355	0.000378	0.00045	0.003106	0.000475	0.001448	0.992798
22	0.006917	0.00388	0.003094	0.002809	0.006	0.006378	0.00759	0.943863	0.003674	0.013089	0.002708
23	0.11758	0.13109	0.867791	0.012224	0.014204	0.021015	0.009935	0.008286	0.019032	0.018044	0.004603
24	0.000891	0.000319	0.000402	0.000362	0.00087	0.000931	0.000971	0.006752	0.0019	0.963781	0.022821
25	0.004518	0.001619	0.002038	0.002241	**0.967406**	0.004001	0.00542	0.004446	0.0044	0.002142	0.001769
26	0.000643	0.002218	0.003479	0.988729	0.000971	0.00084	0.000543	0.000487	0.001284	0.000554	0.000252
27	0.967515	0.011243	0.002522	0.006031	0.007451	0.000945	0.00128	0.00105	0.001039	0.000506	0.000418
28	0.00041	0.0003	0.000496	0.000203	0.000355	0.000378	0.000449	0.003105	0.000475	0.001447	0.992381
29	0.000132	7.41E-05	5.91E-05	5.37E-05	0.000115	0.000122	0.000145	0.000994	0.000153	0.000467	**0.997685**
30	0.004498	0.001959	0.005498	0.002723	0.96324	0.03983	0.005396	0.004427	0.004381	0.002133	0.001761
31	0.004637	0.001662	0.002793	0.00289	0.966278	0.004106	0.003918	0.003514	0.004516	0.003871	0.001815
32	0.00041	0.0003	0.000496	0.000203	0.000355	0.000378	0.000449	0.003105	0.000475	0.001477	0.992381
33	0.00017	0.00019	0.000122	9.94E-05	9.49E-05	0.000779	0.000154	0.001029	0.005801	**0.987493**	0.004066
34	0.000149	8.35E-05	6.66E-05	6.05E-05	0.000107	0.000137	0.000135	0.0009	0.000173	0.985179	0.01301
35	0.001571	0.001752	0.001125	0.000918	0.000876	0.007192	0.001424	0.002848	0.950879	0.030799	0.000615
36	0.000528	0.000189	0.000238	0.000215	0.000516	0.000552	0.000576	0.004005	0.001127	0.000892	0.991161
37	**0.974423**	0.011324	0.000657	0.003961	0.004358	0.000952	0.001289	0.001058	0.001047	0.00051	0.000421
38	0.000221	0.000157	0.000159	0.013857	0.000139	0.001146	0.0002	0.001334	0.010568	0.966562	0.005657
39	0.001393	0.000607	0.001703	0.00069	0.001361	0.001457	0.00152	0.97999	0.002973	0.002173	0.006133
40	0.001146	0.000992	0.005687	0.001011	0.001062	0.002407	0.001039	0.002697	**0.974256**	0.009254	0.000449
41	0.967515	0.011243	0.002522	0.006031	0.007451	0.000945	0.00128	0.00105	0.001039	0.000506	0.000418
42	0.010232	0.00729	0.018894	0.11766	0.009483	0.892686	0.008646	0.007755	0.020424	0.008818	0.004006

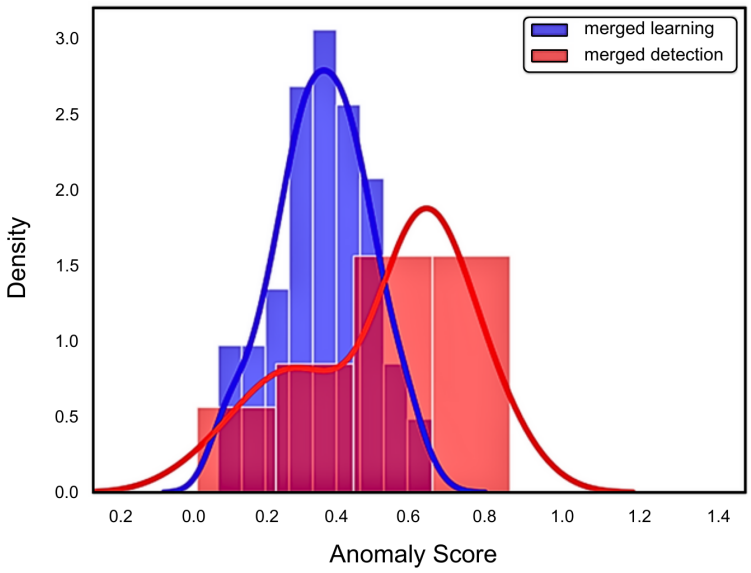

Figure 4: Distribution of averaged anomaly scores, normal (blue) and normal + abnormal out-of-scope queries (red).

2.2 Masqueraders and Masquerade Detection

Attacks can occur when an insider gains control of another employee's credentials with different privileges or when an outsider obtains a legitimate employee's credentials. Internal masqueraders often have greater organizational knowledge, making it easier to mimic employee behavior effectively. In contrast, external masqueraders may behave differently due to their lack of familiarity with the organization [17]. To address these threats, masquerade detection [3] serves as a specialized form of anomaly detection. In the context of SQL, external masquerade detection aims to identify out-of-scope queries that are not typical for database users. On the other hand, internal masquerade detection differentiates between a legitimate user's normal activities and a suspicious action indicative of a masquerader. Early methods for masquerade detection utilized traditional machine learning techniques such as Naive Bayes [18, 19, 20], Support Vector Machines (SVMs) [20, 21], and K-Nearest Neighbors (KNN) [22, 23]. More recently, deep learning approaches [24, 25], including RNN [26, 27], LSTM [28], and Bidirectional Long Short-Term Memory (Bi-LSTM) [29, 30], have been employed in masquerade detection, significantly enhancing detection accuracy. However, these masquerade detection methods are not well-suited to detect various types of SQL anomalies, such as data leaks, SQL injection (SQLi), and data sabotage. Additionally, they do not target the attack source, whether it originates externally or internally.

3 Data

3.1 Data generation

We generated SQL data from 2 datasets:

- Short sequence dataset - we simulated a relational database with 3 user groups (HR, Finance, and DBA), each having approximately 100 unique normalized SQL queries. This dataset comprises 9 tables, 6 views, and 50 attributes, with an average query length of 12 tokens. A unique challenge of our simulated data is the overlap between the different user regions, as multiple groups access the same tables and columns. In addition, the

Table 2: Out-of-scope queries examples: Data leaks, Data sabotage, and SQL injection

Data leak	`select sensitive_c1, sensitive_c2 from T1`
Data sabotage	`DROP TABLE T3` `UPDATE T1 SET COL1 = ?  WHERE COL2 = ?`
SQL injection	`SELECT * FROM T1 WHERE COL1 = ?  OR ? = ?` `SELECT * FROM T1 WHERE COL1 = ?  AND COL2 = ?  OR ? = ?`

Finance group has access to sensitive data, whereas the other user groups have limited access to sensitive data columns.

- Long sequence dataset – We used an open source, comprehensive Customer Relationship Management (CRM) system built on PostgreSQL, installed and configured locally. This dataset includes 3 user groups, with approximately unique normalized 500 SQL per user. The average query length is 200 tokens, with the longest queries reaching up to 1900 tokens.

3.2 Data preprocessing

The following stages were followed in data preprocessing:

3.2.1 Data normalization and cleaning:

- SQL queries were converted to their normalized form by replacing literal values with question marks '?', allowing the model to learn the basic form of the query without focusing on variable values.
- All queries were converted to lowercase.
- A fixed number of spaces was maintained between the tokens to create a uniform pattern.
- Duplicate queries are removed for each user to prevent the model from being biased toward frequent queries.

3.2.2 Overcome model embedding vector capacity limit

For long queries that produce extensive input vectors (after tokenization), we split the input into several vectors based on the model embedding capacity (e.g., 512 tokens for BERT and 1024 tokens for LLaMA). We then averaged the embedding vector chunks for each query based on the assumption that using the element-wise sum or mean of the word embeddings across all words in the sentence effectively preserves the encoded meaning [31].

4 Threat Model

To contextualize our detection performance, we define two primary adversary profiles targeting database systems:

External Adversary (Outsider Threats):

- Access Level: Entry via compromised credentials, network exploits, or misconfigured public interfaces.
- Behavior Patterns: Lack historical behavioral alignment with authorized users. Mimicking legitimate SQL queries is difficult for attackers, especially external ones, as it requires access to application code or risky trial-and-error behavior that tends to trigger anomaly detectors. Tend to issue queries outside normal operational scope, or SQL injection.
- Attack Goals: Data leakage, or destructive actions such as table dropping or schema sabotage.
- Detection Strategy: Unsupervised learning using ensemble anomaly detectors that flag embedding vectors distant from learned normal patterns.

Internal Adversary (Insider Threats):

- Access Level: Legitimate access with elevated privileges or stolen internal credentials.
- Behavior Patterns: Sophisticated mimicry of legitimate user behavior, different behavioral profile, often within expected schema and permission boundaries.
- Attack Goals: Subtle data exfiltration, privilege misuse, unauthorized report generation.
- Detection Strategy: Supervised learning using user-specific DistilBERT embeddings with probabilistic thresholds to catch behavior-role mismatches.

5 Method

Deep learning-based anomaly detection can be categorized into 3 types based on label availability: supervised, semi-supervised, and unsupervised deep anomaly detection [32]. Our approach utilizes supervised and unsupervised methods to detect 2 forms of masquerade attacks: external and internal. Supervised methods are used where both normal and abnormal data are present, enabling binary or multi-class classification [33, 34]. In contrast, unsupervised methods are

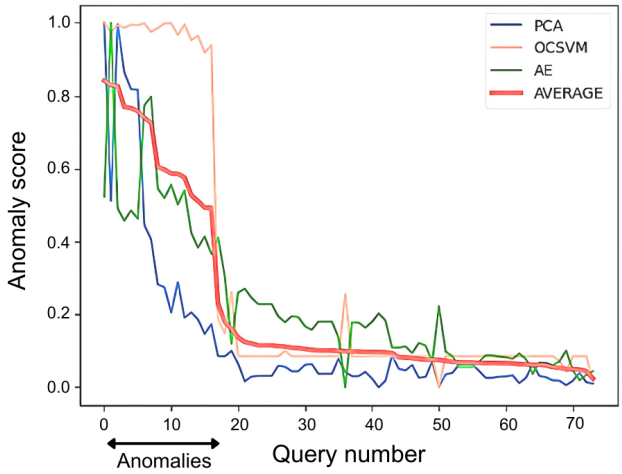

Figure 5: 3 outlier detectors normal + anomalies scores sorted by average score, long sequences.

employed when no labeled data is available or when anomalies need to be detected based on the internal properties of the data samples [35, 36, 37]. We used an unsupervised ensemble anomaly detector based on the fine-tuned DistilBERT model to detect out-of-scope queries. These queries, whose embeddings are significantly distant from the established embedding domain, are probably associated with external masqueraders, as their lack of familiarity with typical employee behaviors often results in anomalous activity [17]. We also evaluated several models for the supervised approach using labeled data. This approach may effectively detect internal masqueraders that can imitate the behavior of other employees within the organization [17]. We conducted distinct learning and detection periods for both methods. The learning period refers to a designated time frame when users are assumed to perform only regular, non-malicious queries. During this period, the system collects user queries, which are pre-processed and used to train the models. Once the learning period concludes, the detection period begins. In this phase, the system monitors new queries performed on the database and alerts any abnormal queries. A detailed description is provided in the following sections.

5.1 Unsupervised part for out-of-scope query detection

This part's primary objective is to detect distant vectors in the multidimensional vector embedding space. Such vectors are likely to fall outside the typical query domain of regular database users, indicating potential anomalous or out-of-scope queries. DistilBERT [16] was chosen for its strong performance and reduced computational requirements. Using a Masked Language Model (MLM) randomly masks tokens in the input and predicts their meaning based on context, effectively capturing the inherent structure and dependencies within the SQL sequences [12]. SQL is similar to human language, making it suitable for natural language processing techniques. In our unsupervised approach,

Table 3: Parameter setting for each model - BERT, LLaMA, DistilBERT, LSTM, BiLSTM, SetFit

Model	Parameter	Selected Value
BERT/LLaMA/DistilBERT	Number of iterations	20
	Batch size	16
	Number of epochs	6
	Learning rate	1e-5
	Activation	Softmax
LSTM/Bi_LSTM	Embedding_dim	The average sequence length
	Batch size	16
	Number of epochs	6
	Activation	Softmax
SetFit	Number of iterations	20
	Batch size	16
	Number of epochs	1
	Learning rate	1e-5

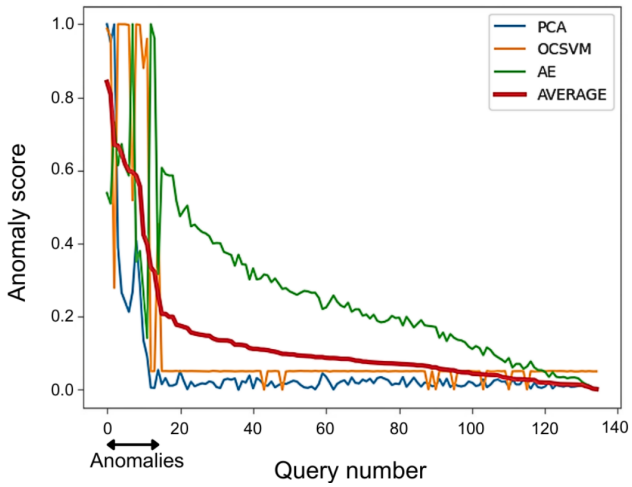

Figure 6: 3 outlier detectors normal + anomalies scores sorted by average score, short sequences.

we fine-tuned DistilBERT on SQL datasets, focusing first on queries collected during the learning period. We then extract the last hidden state embedding, resulting in a 768-dimensional representation. These embeddings serve as an input for creating an ensemble anomaly detector by applying 3 outlier detectors. The 3 detectors trained on the SQL embeddings are Principle Component Analysis (PCA) [38], Autoencoders (AE)[39], and One-Class Support Vector Machine (OCSVM) [40]. Reconstruction errors were calculated for both the AE and PCA models. The embeddings extracted from PCA dimensionality reduction [41], which preserves 98% of the common variance, were used as input for the OCSVM, with decision scores subsequently normalized. The final anomaly score was determined by averaging all 3 normalized scores. This score measured the deviation of an SQL query from the overall set. Out-of-scope queries, including data leaks, attacks such as SQLi attacks, and data sabotage, typically receive the highest anomaly scores. A threshold (with a confidence interval) determines the upper limit of our normal learning period queries. The entire process is repeated on the SQL queries from the detection period, using the threshold set during the learning period to detect out-of-scope queries.

5.2 Supervised part for in-scope query detection

In the supervised part of our study, the learning period serves as a phase in which a probabilistic classifier models the behavior of each role or user and assigns labels for them (for example, in our short sequence dataset – Finance, HR, and DBA are the user groups). A threshold probability is then determined for each role or user. These probabilities represent the likelihood that a given query belongs to a specific user based on the classifier's learned patterns. This learning period contains several key stages, as outlined in the scheme illustrated in Figure 2. The validation data set generates a probability matrix (Table 1). Each column corresponds to one of the 11 users (labeled from 0 to 10) and is associated with certain validation queries. Each row represents a set of user probabilities calculated by the probabilistic classifier for a given validation input instance (validation query) within a stratified validation set. The probabilities across each row are normalized, ensuring their sum equals 1. Each user's probability threshold is determined based on the classification results and the validation input dataset. The lowest significant probability value is determined as the probability threshold. During the detection period, queries are tagged as 'Normal' or 'Abnormal' according to the probability matrix produced by the classification layer.

Table 4: Evaluation results of 6 supervised models with different training sample sizes for long sequences dataset.

Sample Size	Fine-tuned DistilBERT			Fine-tuned BERT			Fine-tuned LLaMA		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
60K	0.34	0.43	0.38	0.42	0.57	0.48	0.14	0.38	0.20
240K	0.66	0.50	0.57	0.75	0.62	0.68	0.42	0.29	0.34
390K	0.87	0.65	0.74	0.82	0.71	0.76	0.63	0.47	0.54
540K	0.88	0.86	0.87	0.96	0.79	0.87	0.81	0.75	0.78
660K	0.96	0.94	**0.95**	0.90	0.93	0.91	0.94	0.91	0.92

The highest probability for the first validation query is 0.956105, associated with user 2 (first row, column 2 in Table 1). Similarly, the highest probability for a second validation query is 0.936215 and is associated with user 5 (second row, column 5 in Table 1). This approach identifies the highest probability value for each validation query as the significant probability. In Table 1, these significant values are marked in bold red. The second step determines the lowest significant probability value for each user. For example, for user 2, the lowest significant probability value is 0.867791 (row 23, column 2 in Table 1). Similarly, for user 5, the lowest significant probability value is 0.892686 (row 42, column 5 in Table 1). These lowest significant probability values are then used to establish a respective probability threshold for each user. In the detection period, a detection input instance based on a query enters a data repository like a database and is classified using the trained probabilistic classifier. The classifier assigns a probability indicating the likelihood that the query belongs to a certain user. This probability is then used to determine if the detection query is abnormal. The detection period contains several key stages described in the scheme illustrated in Figure 3. During the detection period, anomalies in the labeled datasets can be detected in 2 ways. First, an anomaly is flagged if the highest probability is associated with another user. Second, an anomaly is identified if the highest probability belongs to the current user but falls below the threshold (with confidence interval) established during the learning phase. We used all labeled data (SQL queries labeled by each user) to fine-tune various models, including the RNN models (LSTM, Bi-LSTM) and the LLM models (BERT, DistilBERT, LLaMA). Additionally, we fine-tuned the Sentence Transformer Fine-tuning (SetFit) [42], a novel approach to few-shot text classification. SetFit is significantly faster in inference and training compared to similar methods like T-FEW [43], ADAPET [44], and PERFECT [45] while also delivering strong performance with significantly smaller and more efficient base models.

6 Results

The following sections present the results of unsupervised and supervised anomaly detection methods applied to SQL. We first evaluate the unsupervised model, which uses fine-tuned DistilBERT embeddings alongside the ensemble anomaly detectors on unlabeled data. We then assess the performance of the supervised model using labeled SQL queries.

6.1 Unsupervised, out-of-scope query detection results

We conducted several analyses to evaluate the unsupervised model and gain insights into its performance. These include:

- Visualizing the distributions of anomaly scores compared to normal scores of embedding vectors to recognize the difference between them.
- Comparing the anomaly scores generated by the 3 outlier detectors and their averaged scores across the long and short sequences datasets.

Table 2 presents out-of-scope SQL query examples, including those that could result in data leaks, data sabotage, or SQLi. In anomaly or outlier detection [46, 32], it is generally assumed that only a small number of anomalous data instances exist in the distribution. Figure 4 shows the distribution of anomaly scores from the learning period (composed only of normal queries) compared to the detection period (consisting of normal and abnormal queries). These scores are derived from the combined output of the 3 normalized outlier detectors. The distribution appears to follow a near-normal pattern, with most SQL queries clustered around the mean, indicating typical anomaly scores. However, the distribution also reveals the presence of some outlier SQLs that exhibit significantly higher anomaly scores, marking them as potential anomalies. The anomaly scores from the 3 outlier detectors for the long and short SQL sequences consisting of normal and abnormal queries during the detection period are presented in Figures 5 and

Table 5: Evaluation results of 3 supervised LSTM, Bi-LSTM, and SetFit models with different training sample sizes for long sequences dataset.

Sample Size	LSTM			Bi-LSTM			SetFit		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
60K	0.22	0.11	0.15	0.22	0.35	0.27	0.58	0.65	0.61
240K	0.30	0.11	0.16	0.59	0.38	0.46	0.80	0.65	0.72
390K	0.22	0.12	0.16	0.77	0.60	0.67	0.90	0.74	0.81
540K	0.36	0.125	0.18	0.90	0.78	0.84	0.92	0.84	0.88
660K	0.35	0.13	0.19	0.90	0.93	0.91	0.97	0.98	**0.97**

6 (respectively). The red line represents the average of those 3 detectors, with scores sorted based on this average. Overall, the OCSVM tends to produce the highest anomaly scores, while the AE and PCA produce the lowest scores. However, in some cases, AE and PCA scores are more effective at revealing specific anomalies than OCSVM. Notably, in the short sequences graph (Figure 6), the AE scores appear more volatile than in the long sequences graph (Figure 5). Despite these differences, both graphs clearly distinguish between normal and abnormal queries when using the average of the 3 outlier detectors.

6.2 Supervised, in-scope query detection results

The supervised component of our approach uses labeled SQL data corresponding to different database users. A query is deemed abnormal if it is assigned to a different user or if it is assigned to the correct user but falls below the established probability threshold. Conversely, a query is considered normal if it is classified to the same labeled user and exceeds the established threshold. The labeled dataset is divided into training and testing sets in an 85:15 ratio. To ensure balance, the training data includes an equal number of SQL queries for each user, which are then combined to create a few-shot training set. Given the potential instability of evaluation results from a small-size training set, we conducted 5 experiments for each model and sample size per class. The parameter settings for each model are described in Table 3. The F1 scores (2 * (precision * recall) / (precision + recall)) are presented in Figures 7 and 8 for LSTM, Bi-LSTM, LLaMA, BERT, DistilBERT, and SetFit models for long and short SQL sequences, respectively. The training data sizes were examined in the long sequence dataset (Figure 7): 60K, 240K, 390K, 540K, and 660K. The training data sizes examined in the short sequence dataset (Figure 8) are 7K, 12K, and 17K. The averaged precision, recall, and F1 scores for short and long-sequence datasets are described in Tables 4, 5, 6 and 7. While Tables 4 and 5 describe the scores on long sequence datasets with training data sizes varying from 60K to 660K, Tables 6 and 7 describe the scores on short sequence datasets with training data sizes varying from 7K to 17K. As expected, the model's performance improves as the sample size increases. In addition, across all sample sizes, we noticed that LLaMA performances are lower than BERT/DistilBERT when using our limited data. Our findings align with Bumgardner's conclusion [47] that claims the BERT model achieves high performance on smaller datasets, whereas the LLaMA models excelled with the larger datasets. This discrepancy can be attributed to the simpler classification challenge of smaller datasets featuring fewer class labels and examples compared to the complexity of the larger dataset, which offers more complex training data. Another significant observation is that the Bi-LSTM models consistently outperformed LSTM models. This is because the Bi-LSTM processes input in both directions, leveraging contextual information from both sides, unlike LSTM, which processes data in a single direction. Additionally, the fine-tuned SetFit model yielded the best results when trained on the largest data sets, outperforming the fine-tuned DistilBERT with the same data size. Conversely, LSTM models produced the lowest performance, highlighting the advantages of using SetFit for fine-tuning pre-trained models when labeled data is limited. Overall, the experimental results demonstrate the effectiveness of creating a small set of manually labeled SQL, fine-tuning a pre-trained model with SetFit, and subsequently using it for automated SQL classification. We combined equal numbers of normal and abnormal sessions into a few-shot training set that resulted in improved performance as the number of samples per user increased.

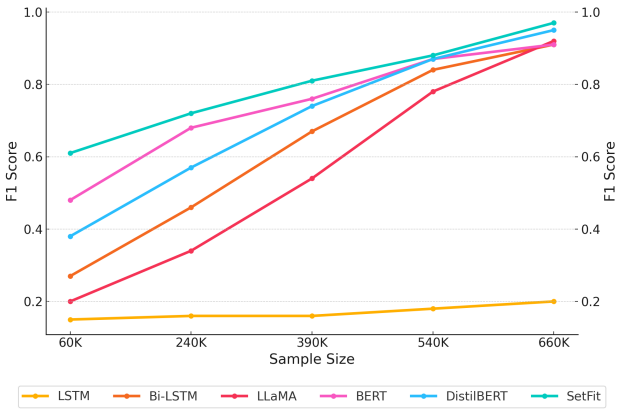

Figure 7: F1 scores of 6 supervised models with different training sizes for long sequences dataset.

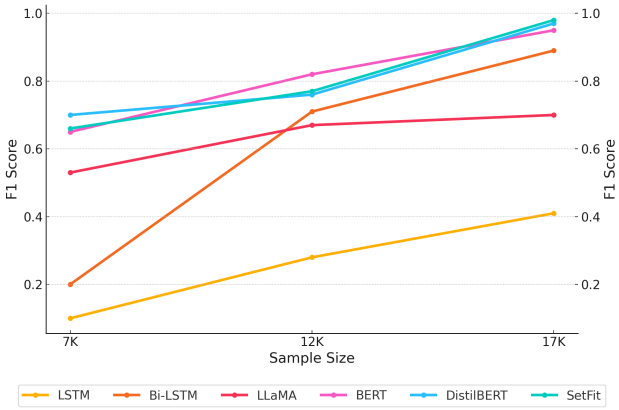

Figure 8: F1 scores of 6 supervised models with different training sizes for short sequences dataset.

7 Conclusions

Detecting abnormal database access behavior remains a critical challenge, especially given the limitations of existing approaches in processing complex statements, detecting abnormal behavior, performing root cause analysis, and maintaining precision. To address these challenges, we introduce an innovative approach that effectively detects abnormal database access behavior from both external and internal aspects. Our approach leverages advanced techniques for extracting semantic information from SQL operation statements and integrates 2 distinct detection techniques, unsupervised and supervised, to achieve highly accurate anomaly detection. Extensive experiments conducted on datasets from 2 different database scenarios indicate that our techniques outperform SOTA methods. The results show that our model enhances detection accuracy and provides a more comprehensive analysis of user behavior, making it well-suited for real-world applications. By refining anomaly detection with a blend of semantic understanding and advanced machine learning techniques, our approach sets a new standard for ensuring database security and integrity.

References

[1] CYFOR Forensics. Former employees behind tesla data breach, 2023. Accessed: 2025-03-20.

[2] Ponemon Institute and Sullivan Report. Cost of insider risks global report 2023, 2023. Accessed: 2025-03-20.

[3] Maximiliano Bertacchini and Pablo Fierens. A survey on masquerader detection approaches. In *Proceedings of V Congreso Iberoamericano de Seguridad Informática*, pages 46–60. Universidad de la República de Uruguay, 2008.

[4] Syed R. Hussain, Ahmed M. Sallam, and Elisa Bertino. Detanom: Detecting anomalous database transactions by insiders. In *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, pages 25–35. Association for Computing Machinery, 2015.

[5] Asmaa Sallam, Elisa Bertino, Syed Rafiul Hussain, David Landers, R. Michael Lefler, and Donald Steiner. DBSAFE—an anomaly detection system to protect databases from exfiltration attempts. *IEEE Systems Journal*, 11(2):483–493, 2017.

[6] Mahdi Alizadeh, Sander Peters, Sandro Etalle, and Nicola Zannone. Behavior analysis in the medical sector: theory and practice. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, pages 1637–1646, 2018.

Table 6: Evaluation results of 3 supervised Fine-tuned DistilBERT, BERT, and LLaMA models with different training sample sizes for short sequences dataset.

Sample Size	Fine-tuned DistilBERT			Fine-tuned BERT			Fine-tuned LLaMA		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
7K	0.81	0.62	0.70	0.72	0.60	0.65	0.79	0.40	0.53
12K	0.83	0.70	0.76	0.87	0.78	0.82	0.74	0.61	0.67
17K	0.97	0.98	**0.97**	0.94	0.96	0.95	0.69	0.72	0.70

Table 7: Evaluation results of 3 supervised LSTM, Bi-LSTM, and SetFit models with different training sample sizes for short sequences dataset.

Sample Size	LSTM			Bi-LSTM			SetFit		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
7K	0.06	0.25	0.10	0.14	0.37	0.20	0.62	0.70	0.66
12K	0.24	0.35	0.28	0.75	0.67	0.71	0.75	0.80	0.77
17K	0.40	0.42	0.41	0.88	0.90	0.89	0.98	0.975	**0.98**

[7] Ma'ayan Gafny, Asaf Shabtai, Lior Rokach, and Yuval Elovici. Poster: applying unsupervised context-based analysis for detecting unauthorized data disclosure. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 765–768, 2011.

[8] Muhammad Imran Khan and Simon N. Foley. Detecting anomalous behavior in dbms logs. In *Risks and Security of Internet and Systems - 11th International Conference, CRiSIS 2016, Roscoff, France, September 5-7, 2016, Revised Selected Papers*, pages 147–152. Springer International Publishing, 2017.

[9] Garfield Zhiping Wu, Sylvia L. Osborn, and Xin Jin. Database intrusion detection using role profiling with role hierarchy. In *Secure Data Management: 6th VLDB Workshop, SDM 2009, Lyon, France, August 28, 2009. Proceedings*, pages 33–48. Springer, 2009.

[10] Muhammad Imran Khan, Barry O'Sullivan, and Simon N. Foley. A semantic approach to frequency based anomaly detection of insider access in database management systems. In *Risks and Security of Internet and Systems*, pages 18–28. Springer International Publishing, 2018.

[11] Sunu Mathew, Michalis Petropoulos, Hung Q. Ngo, and Shambhu Upadhyaya. A data-centric approach to insider attack detection in database systems. In *Recent Advances in Intrusion Detection*, volume 6307 of *Lecture Notes in Computer Science*, pages 382–401. Springer Berlin Heidelberg, 2010.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[13] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[14] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[15] Haixuan Guo, Shuhan Yuan, and Xintao Wu. Logbert: Log anomaly detection via bert. In *2021 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2021.

[16] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[17] Muhammad Imran Khan, Simon N. Foley, and Barry O'Sullivan. Database intrusion detection systems (dids): Insider threat detection via behavioural-based anomaly detection systems—a brief survey of concepts and approaches. *arXiv preprint arXiv:2011.02308*, 2020.

[18] Roy A Maxion and Tahlia N Townsend. Masquerade detection using truncated command lines. In *Proceedings international conference on dependable systems and networks*, pages 219–228. IEEE, 2002.

[19] Roy A. Maxion. Masquerade detection using enriched command lines. In *Proceedings of the 2003 International Conference on Dependable Systems and Networks (DSN 2003)*, pages 5–14. IEEE Computer Society, 2003.

[20] Ke Wang and Salvatore J. Stolfo. One-class training for masquerade detection. In *Proceedings of the 2003 ACM Workshop on Data Mining for Security Applications (DMSA'03)*, pages 1–8. Association for Computing Machinery, 2003.

[21] Han-Sung Kim and Sung-Deok Cha. Empirical evaluation of svm-based masquerade detection using unix commands. *Computers & Security*, 24(2):160–168, 2005.

[22] Juan A. Recio-García, Mauricio Gabriel Orozco del Castillo, and Jose A. Soladrero. Case-based explanation of classification models for the detection of sql injection attacks. In *Proceedings of the Workshops at the 31st International Conference on Case-Based Reasoning (ICCBR-WS 2023)*, pages 200–215, 2023.

[23] Katarzyna A. Tarnowska and Araav Patel. Log-based malicious activity detection using machine and deep learning. In *Malware Analysis Using Artificial Intelligence and Deep Learning*, pages 581–604. Springer, 2021.

[24] Shuhan Yuan and Xintao Wu. Deep learning for insider threat detection: Review, challenges and opportunities. *Computers & Security*, 104:102221, 2021.

[25] Wisam Elmasry, Akhan Akbulut, and Abdul Halim Zaim. Deep learning approaches for predictive masquerade detection. *Security and Communication Networks*, 2018:Article ID 9327215, 2018.

[26] Fangfang Yuan, Yanan Cao, Yanmin Shang, Yanbing Liu, Jianlong Tan, and Binxing Fang. Insider threat detection with deep neural network. In *Computational Science–ICCS 2018: 18th International Conference, Wuxi, China, June 11–13, 2018, Proceedings, Part I 18*, pages 43–54. Springer, 2018.

[27] Shuhan Yuan, Panpan Zheng, Xintao Wu, and Qinghua Li. Insider threat detection via hierarchical neural temporal point processes. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1343–1350. IEEE, 2019.

[28] Adam Adenike Azeezat, Onashoga Sadiat Adebukola, Abayomi-Alli Adebayo, and Omoyiola Bayo Olushola. A conceptual hybrid model of deep convolutional neural network (dcnn) and long short-term memory (lstm) for masquerade attack detection. In *Information and Communication Technology and Applications: Third International Conference, ICTA 2020, Minna, Nigeria, November 24–27, 2020, Revised Selected Papers 3*, pages 170–184. Springer, 2021.

[29] Saiyu Hao, Jun Long, and Yingchuan Yang. Bl-ids: Detecting web attacks using bi-lstm model based on deep learning. In *International conference on security and privacy in new computing environments*, pages 551–563. Springer, 2019.

[30] Yuqi Yu, Guannan Liu, Hanbing Yan, Hong Li, and Hongchao Guan. Attention-based bi-lstm model for anomalous http traffic detection. In *2018 15th International Conference on Service Systems and Service Management (ICSSSM)*, pages 1–6. IEEE, 2018.

[31] Lyndon White, Roberto Togneri, Wei Liu, and Mohammed Bennamoun. How well sentence embeddings capture meaning. In *Proceedings of the 20th Australasian Document Computing Symposium*, pages 1–8, 2015.

[32] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint*, arXiv:1901.03407, January 2019.

[33] Raghavendra Chalapathy, Ehsan Zare Borzeshi, and Massimo Piccardi. Bidirectional lstm-crf for clinical concept extraction. *arXiv preprint*, arXiv:1610.05858, October 2016.

[34] Raghavendra Chalapathy, Ehsan Zare Borzeshi, and Massimo Piccardi. An investigation of recurrent neural architectures for drug name recognition. In *Proceedings of the Seventh International Workshop on Health Text Mining and Information Analysis*, pages 1–5, Austin, TX, 2016. Association for Computational Linguistics.

[35] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018.

[36] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2828–2837, 2019.

[37] Aaron Tuor, Samuel Kaplan, Brian Hutchinson, Nicole Nichols, and Sean Robinson. Deep learning for unsupervised insider threat detection in structured cybersecurity data streams. In *AAAI Workshops*, pages 224–231, 2017.

[38] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. In *Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop*, pages 172–179. IEEE Press, 2003.

[39] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, 1989.

[40] Zaruhi Alaverdyan, Julien Jung, Romain Bouet, and Carole Lartizien. Regularized siamese neural network for unsupervised outlier detection on brain multiparametric magnetic resonance imaging: Application to epilepsy lesion screening. *Medical Image Analysis*, 60:101618, 2020.

[41] Basna Mohammed Salih Hasan and Adnan Mohsin Abdulazeez. A review of principal component analysis algorithm for dimensionality reduction. *Journal of Soft Computing and Data Mining*, 2(1):20–30, 2021.

[42] Lewis Tunstall, Nils Reimers, Unso Eun Seo Jo, Luke Bates, Daniel Korat, Moshe Wasserblat, and Oren Pereg. Efficient few-shot learning without prompts. *arXiv preprint*, arXiv:2209.11055, 2022.

[43] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

[44] Derek Tam, Rakesh R Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. Improving and simplifying pattern exploiting training. *arXiv preprint arXiv:2103.11955*, 2021.

[45] Rabeeh Karimi Mahabadi, Luke Zettlemoyer, James Henderson, Lambert Mathias, Marzieh Saeidi, Veselin Stoyanov, and Majid Yazdani. Perfect: Prompt-free and efficient few-shot learning with language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3638–3652, Dublin, Ireland, 2022. Association for Computational Linguistics.

[46] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

[47] VK Cody Bumgardner, Aaron Mullen, Samuel E Armstrong, Caylin Hickey, Victor Marek, and Jeff Talbert. Local large language models for complex structured tasks. *AMIA Summits on Translational Science Proceedings*, 2024:105, 2024.