CYBERSECURITY THREAT DETECTION BASED ON A UEBA FRAMEWORK USING DEEP AUTOENCODERS

A PREPRINT

José Fuentes
Galician Research and Development Center
in Advanced Telecommunications (Gradiant)
36214 Vigo, Spain
jfuentes@gradiant.org

Ines Ortega-Fernández*
Galician Research and Development Center
in Advanced Telecommunications (Gradiant)
36214 Vigo, Spain
iortega@gradiant.org

Nora M. Villanueva
Universidade de Vigo, Dep. of Statistics and O.R.
& SiDOR Group, 36310 Vigo (Spain)
nmvillanueva@uvigo.gal

Marta Sestelo
Galician Centre for Mathematical Research
and Technology (CITMAga),
Santiago de Compostela (Spain)
Universidade de Vigo, Dep. of Statistics and O.R.
& SiDOR Group, 36310 Vigo (Spain)
sestelo@uvigo.gal

June 8, 2025

ABSTRACT

User and Entity Behaviour Analytics (UEBA) is a broad branch of data analytics that attempts to build a normal behavioural profile in order to detect anomalous events. Among the techniques used to detect anomalies, Deep Autoencoders constitute one of the most promising deep learning models on UEBA tasks, allowing explainable detection of security incidents that could lead to the leak of personal data, hijacking of systems, or access to sensitive business information. In this study, we introduce the first implementation of an explainable UEBA-based anomaly detection framework that leverages Deep Autoencoders in combination with Doc2Vec to process both numerical and textual features. Additionally, based on the theoretical foundations of neural networks, we offer a novel proof demonstrating the equivalence of two widely used definitions for fully-connected neural networks. The experimental results demonstrate the proposed framework capability to detect real and synthetic anomalies effectively generated from real attack data, showing that the models provide not only correct identification of anomalies but also explainable results that enable the reconstruction of the possible origin of the anomaly. Our findings suggest that the proposed UEBA framework can be seamlessly integrated into enterprise environments, complementing existing security systems for explainable threat detection.

Keywords Anomaly detection · UEBA · Autoencoders · Deep learning · Cybersecurity · Cyber threat

1 Introduction

In the current digital era, cybersecurity and the reliability of both physical and logical systems have become of increased importance for industry and academia alike. The exponential growth of interconnected devices, the increasing volume of sensitive data, and the complexity of technological infrastructures highlight the need for robust algorithms that ensure security and resilience. Mathematical models and methods play a central role in this task by offering formal frameworks to identify cyberattacks, develop cryptographic protocols, simulate potential incidents under a wide range of scenarios,

*Corresponding author

and design defence strategies against adversarial threats. As these challenges intensify, mathematical approaches are becoming more crucial to guarantee robust system performance and to mitigate future cyber risks.

User and Entity Behaviour Analytics (UEBA) Shashanka et al. (2016a) is a powerful methodology for identifying cyber threats by creating models of normal behavioural patterns and detecting deviations that may indicate malicious or negligent activities. To profile the behaviour of an entity, these models allow the incorporation of multiple sources of information such as sensor readings, network traffic, system logs, security alerts, email information, and even geo-positioned or biometric data. UEBA uses advanced statistical learning techniques to model the behaviour of users, employees, and customers, as well as machines, such as servers, switches, and personal systems. By analysing anomalies in the behaviour of users and devices, UEBA can detect intrusions, impersonation attacks, or negligent users Maher (2017).

In parallel, Explainable Artificial Intelligence (xAI) has emerged as a key enabler in cybersecurity scenarios by addressing the growing need for interpretability and trust in complex AI-driven security systems. Since modern cybersecurity solutions are relying on deep learning models, the inherent lack of transparency can prevent the analysts' ability to understand and respond to the detected threats effectively. xAI techniques enable cybersecurity experts to interpret and understand why a system has flagged a specific event as anomalous behaviour, facilitating root-cause analysis and informed decision-making. Therefore, by integrating xAI techniques, UEBA-based cybersecurity tools not only improve technical performance but also enhance compliance with regulatory frameworks that demand transparency in AI-based decision-making, especially in sensitive areas such as finance.

Several methods have been proposed in the literature related to the use of these techniques. In Shashanka et al. (2016b), UEBA models based on Mahalanobis distance and Singular Value Decomposition are implemented to identify anomalous behaviour in users accessing a server. Voris et al. Voris et al. (2019) use Gaussian mixture models for each computer, gathering file system, process launching, and network behaviour data, in addition to setting up a series of trap files to attract and identify attackers. They also apply UEBA to continuously identify the user in the system by monitoring their activity. Another example is found in Pusara and Brodley (2004), where decision trees are applied to mouse movement data to identify the user. Similarly, in Slipenchuk and Epishkina (2019), results of applying UEBA with mouse movement data, keyboard typing dynamics, and event sequences in the context of online banking operations are compared. In Meng et al. (2018), a similar approach based on the Radial Basis Function Network (RBFN) classifier with Particle Swarm Optimisation (PSO) is applied to verify user identity through touchscreen usage and other biometric data during web browsing. Moreover, the combination of static (logins, cookies, system type, etc.) and dynamic (mouse, keyboard, microphone, network usage, etc.) data to build user models was explored in Martin et al. (2021), where UEBA models are also used to guarantee user coherence when performing authentication with identity federations.

UEBA can be considered a use case for anomaly detection (or outlier detection) with personalised models. Once the users are identified, each model has to detect data points that do not conform to the expected behaviour. As anomaly detection has grown in popularity, a wide array of methods and techniques has emerged. Among these, autoencoders Rumelhart et al. (1986) are a type of artificial neural network used for this purpose. Their goal is to learn (in an unsupervised way) a representation of the dataset by filtering out insignificant data or noise. Recent work in xAI for cybersecurity highlights the necessity of designing inherently interpretable (ante-hoc) models, prioritizing explainability principles from model conception through training Ortega-Fernandez et al. (2024). A key advantage of the use of autoencoders for cybersecurity is that they tend to be more explainable than other deep learning models Morales-Forero and Bassetto (2019); Gonzalez-Muniz et al. (2022). Moreover, in cases where the Well-Defined Anomaly Distribution (WDAD) assumption does not hold Görnitz (2019); Flynn et al. (2023), autoencoders can be trained on data assumed to be normal (even if slightly contaminated) Mauritz et al. (2021).

Autoencoders have been used for anomaly detection since the work of Hawkins et al. Hawkins et al. (2002), and a general overview can be found in Wang et al. (2019). Examples include using Variational Autoencoders (VAEs) to construct anomaly scores Xiao et al. (2020) and convolutional autoencoders for video signal anomaly detection Ribeiro et al. (2018). In Sakurada and Yairi (2014) both incomplete and overcomplete autoencoders are applied in satellite data, while Wang et al. (2022) proposes a VAE coupled with a transformer architecture to account for dependencies in satellite data. In industrial anomaly detection, Zhou and Paffenroth (2017) uses a norm-regularised autoencoder, and Morales-Forero and Bassetto (2019) combines an autoencoder with an LSTM network. Also, Gonzalez-Muniz et al. (2022) utilises VAEs to classify anomalies in engineering systems. Finally, Ortega-Fernandez et al. (2023) showed that autoencoders outperform other algorithms in the detection of Denial of Service cyberattacks in industrial scenarios.

Our work makes a twofold contribution. First, we provide novel theoretical results by proving the equivalence of two common definitions of fully-connected neural networks. Second, to the best of our knowledge, we present the first implementation of an explainable UEBA-based anomaly detection framework using autoencoders. Our methodology includes the use of text encoding models (Doc2Vec) alongside autoencoders to leverage both numerical and textual data, training with contaminated (unlabelled) data, and using model residuals for explainability.

This paper is structured as follows. Section 2 details the proposed methodology, including theoretical results and a description of the methods employed for feature extraction, residual space analysis and the proposed architecture for UEBA-based anomaly detection. Section 3 presents the results of the application of the proposed methodology to a real use-case of cybersecurity in a financial institution. Finally, Section 4 outlines the main conclusions and future research directions.

2 Materials and methods

In this work, we propose a novel UEBA-based anomaly detection framework based on Deep Autoencoders and the Doc2Vec algorithm for the pre-processing of text features. In the following Subsections 2.1-2.4, we describe the mathematical foundations of the used algorithms and present our theoretical contributions with a new proof of the equivalence of two common definitions of neural networks. Moreover, Subsection 2.5 describes the architectures of the UEBA-based anomaly detection framework.

2.1 Neural Networks

Neural networks constitute a large set of learning models that originate from the early work on the Rosenblatt perceptron Rosenblatt (1958). They approximate functions by interleaving affine transformations with non-linear activation functions. A feed-forward deep neural network use longer chains of concatenated affine and activation functions to improve the representation of the target function. This definition of a feed-forward deep neural can be formally expressed in the following manner.

Definition 1. *Let $\mathcal{F} \subset \{\varphi : \mathbb{R} \to \mathbb{R}\}$ be a set of activation functions. Given $d \geq 2$ and an input dimension $n^{(0)} = n$, for each $l = 1, \ldots, d$, let $n^{(l)} \in \mathbb{Z}^+$ with $n^{(d)} = m$, and let $A^{(l)} : \mathbb{R}^{n^{(l-1)}} \to \mathbb{R}^{n^{(l)}}$ be affine transformations. Define $\Phi^{(l)} : \mathbb{R}^{n^{(l)}} \to \mathbb{R}^{n^{(l)}}$ by*

$$\Phi^{(l)}(\mathbf{x}) = (\varphi_1(x_1), \ldots, \varphi_{n^{(l)}}(x_{n^{(l)}})),$$

with each $\varphi_j \in \mathcal{F}$ for $j = 1, \ldots, n^{(l)}$. Then, a feed-forward deep neural network with $d-1$ hidden layers is the function

$$\hat{f} = \Phi^{(d)} \circ A^{(d)} \circ \Phi^{(d-1)} \circ A^{(d-1)} \circ \cdots \circ \Phi^{(1)} \circ A^{(1)} : \mathbb{R}^n \to \mathbb{R}^m.$$

However, as noted in Mhaskar and Poggio (2019), this definition does not uniquely determine the network structure and makes it difficult to formalise concepts such as sparsity and convolutions. An alternative, more constructive description is based on a layered graph where each node (or neuron) implements a simple function, as formalized by Mhaskar and Poggio (2016); Cano-Córdoba et al. (2017). For brevity, we omit the definitions layered graph and neuron and they can be found in Cano-Córdoba et al. (2017).

Definition 2. *Given a layered graph $\mathfrak{G}$, a feed-forward deep neural network with structure $\mathfrak{G}$ is any function defined on $\mathfrak{G}$ ($\mathfrak{G}$-function) such that each constituent function is a neuron.*

Neural networks of fixed depth or fixed width can approximate a wide range of functions modeling real-life processes Leshno et al. (1993); Kidger and Lyons (2020). The universality property is crucial for any application; hence, it's significant to prove that both definitions are equivalent. Below, we provide a mathematical proof (Proposition 1) demonstrating this equivalence:

Proposition 1. *Definitions 1 and 2 are equivalent.*

Proof. We prove the equivalence of Definitions 1 and 2.

(Definition 1 $\Rightarrow$ Definition 2): Let $\hat{f} = \Phi^{(d)} \circ A^{(d)} \circ \cdots \circ \Phi^{(1)} \circ A^{(1)}$ with

$$\hat{f}^{(i)} = \Phi^{(i)} \circ A^{(i)} : \mathbb{R}^{n^{(i-1)}} \to \mathbb{R}^{n^{(i)}}, \quad n^{(0)} = n,\ n^{(d)} = m.$$

For $\mathbf{x} \in \mathbb{R}^n$, set

$$\mathbf{h}^{(0)} = \mathbf{x}, \quad \mathbf{h}^{(i)} = (\hat{f}^{(i)} \circ \cdots \circ \hat{f}^{(1)})(\mathbf{x}).$$

Since each coordinate

$$\pi_j\big(\hat{f}^{(i)}(\mathbf{h}^{(i-1)})\big) = \varphi_j\Big(W_j^{(i)} \cdot \mathbf{h}^{(i-1)} + b^{(i)}\Big)$$

defines a neuron (i.e., the function $\pi_j \circ \hat{f}^{(i)}$), we construct a layered graph $\mathfrak{G} = (V, E)$ by:

1. *Input layer:* $V^{(0)}$ consists of n nodes (the input coordinates $\mathbf{x}$).
2. *Layers* 1 *to* d*:* For each i, let $V^{(i)}$ consist of $n^{(i)}$ nodes, where each node $v_j \in V^{(i)}$ is assigned the function $\pi_j \circ \hat{f}^{(i)}$ and has incoming edges
$$I_v^- = \{(u, v) : u \in V^{(i-1)}\}.$$

Thus, $\hat{f}$ is a $\mathfrak{G}$-function.

(Definition 2 ⇒ Definition 1): Conversely, let $\mathfrak{G} = (V, E)$ be a layered graph with layers
$$V^{(i)} = \{v_1, \ldots, v_{n^{(i)}}\} \quad (n^{(0)} = n,\ n^{(d)} = m),$$
and let each node $v \in V^{(i)}$ have an associated function f_v. For $\mathbf{h}^{(i-1)} \in \mathbb{R}^{n^{(i-1)}}$, let $\mathbf{z}_v$ be the subvector of inputs corresponding to the predecessors of v. Define
$$\hat{f}^{(i)}(\mathbf{h}^{(i-1)}) = \big(f_{v_1}(\mathbf{z}_{v_1}), \ldots, f_{v_{n^{(i)}}}(\mathbf{z}_{v_{n^{(i)}}})\big).$$
Then, the overall network can be written as $\hat{f} = \hat{f}^{(d)} \circ \cdots \circ \hat{f}^{(1)}$, which is of the form given in Definition 1.

□

2.2 Autoencoders

An autoencoder is a model that approximates the identity function under a constraint that forces the model to capture the most salient features of the input. In our case, for a random sample $X = (\mathbf{x}_1, \ldots, \mathbf{x}_m)$ with $\mathbf{x}_i \in \mathbb{R}^n$, we define an autoencoder $\mathbf{AE}_n^p$ as follows.

Definition 3. *Given positive integers n and p (with $p < n$), an autoencoder $\mathbf{AE}_n^p$ is a tuple*
$$(n, p, f, g, \mathcal{E}, \mathcal{D}, X, \Delta)$$
where:

- *$\mathcal{E}$ and $\mathcal{D}$ are sets of functions from $\mathbb{R}^n$ to $\mathbb{R}^p$ and from $\mathbb{R}^p$ to $\mathbb{R}^n$, respectively;*
- *$f \in \mathcal{E}$ is the encoder and $g \in \mathcal{D}$ is the decoder;*
- *Δ is a dissimilarity measure (typically a metric) on $\mathbb{R}^n$.*

The latent space is the codomain of f, where the compressed representation of $\mathbf{x}$ is stored. The reconstruction error is defined as:

Definition 4. *The reconstruction error of the autoencoder is given by*
$$E_{f,g}(X) = \sum_{i=1}^{m} \Delta(g(f(\mathbf{x}_i)), \mathbf{x}_i).$$

Training an autoencoder involves finding functions f and g that minimise $E_{f,g}(X)$. In our work, both encoder and decoder are implemented as fully connected (regularised) neural networks. We focus on under-complete autoencoders ($p < n$) to force a compressed representation, which in turn leads the model to learn the dominant patterns of normal behaviour. Since anomalous data points are rare, the model prioritises the reconstruction of normal samples, making the reconstruction error an effective anomaly score.

2.3 Doc2Vec

To process text-based variables (e.g., lists of executed processes), we use the Doc2Vec model Le and Mikolov (2014). Doc2Vec is a neural network-based embedding method that learns vector representations of documents in an unsupervised manner. Similar to Word2Vec Mikolov et al. (2013), it clusters similar texts in the vector space. In the same way that Word2Vec embeds related words like synonyms or topics close in the vector space, Doc2Vec also clusters similar texts together, for example by identifying the topic of the text or by finding similar words between texts. Two main algorithms exist for training Doc2Vec: Distributed Bag of Words (DBOW) and Distributed Memory (DM). In DBOW, the document vector is used to predict random word vectors from the document; in DM, both the document

vector and word vectors are used to predict the next word in a sequence. The resulting document embeddings capture semantic similarities that are later used in our UEBA framework.

In this work, we apply Doc2Vec trained with DBOW to extract information from the lists of processes recovered from the activity logs. This way, we can capture when programs are run at the same time, and similarities in processes from their executable path.

2.4 t-distributed Stochastic Neighbor Embedding

The t-distributed Stochastic Neighbor Embedding (t-SNE) Van der Maaten and Hinton (2008) is a dimensionality reduction technique used primarily for visualizing high-dimensional data in two or three dimensions. It preserves local structures by mapping similar points from high-dimensional space to nearby points in the lower-dimensional embedding, while distant points remain separated. t-SNE works by first computing pairwise conditional probabilities based on distances between points in the high-dimensional space, and modelling the local similarities as Gaussian distributions. It then maps these points into a lower-dimensional space using a Student's t-distribution to model the similarity of the embeddings. The algorithm iteratively adjusts embedding positions by minimizing the Kullback-Leibler divergence between these two distributions:

$$\mathrm{KL}(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}},$$

where P and Q represent the joint probability distributions of pairwise similarities between data points in the high-dimensional and low-dimensional spaces, respectively. This lower dimensionality embedding allows us to identify patterns that are characterized by their local structure, such as clusters and anomalies in high dimensional datasets.

We use t-SNE to analyse the dataset and validate the autoencoder model's behaviour. We apply t-SNE to the test data to analyse the presence of clusters indicative of different user groups and behaviours, and validate the feasibility of a UEBA-based approach. Moreover, we also use t-SNE to study the residuals of the reconstruction error of the model to verify that anomalous data points remain distinguishable within the residual space.

2.5 UEBA-based anomaly detection framework

The dataset used in this study is derived from multiple real data sources, including Windows events of user activity, emails, and antivirus logs of a financial institution. For this reason, preprocessing plays a crucial role in preparing the raw data for effective anomaly detection. The preprocessing steps involve cleaning, transforming, and encoding the data using the Doc2Vec model described in Section 2.3 to derive features into a format suitable for the anomaly detection model.

The data is ingested as time-series from logs, and is aggregated into fixed windows, summarizing them into key statistics, like total counts and average time intervals. In addition, we perform feature engineering to derive new variables that better capture behavioural patterns, including metrics such as the average time between logins, the ratio of failed to successful logins, and the frequency of antivirus alerts. The set of derived features is detailed in Table 1, with 2 indexing variables and 19 features.

For handling missing values, a lack of activity is assumed to correspond to a zero count, while timing variables (e.g., `avg_sec_bet_logins`) are imputed using the duration of the aggregation window in seconds as a maximum time.

Meanwhile, text data (specifically, the executable names from processes executed within each window) are combined into a single field (`process_list`) and encoded using a Doc2Vec model (trained with DBOW) to generate a 64-dimensional embedding vector for each window. These embeddings are concatenated to the derived numerical features, obtaining the final input vector with 83 features.

Finally, all numerical features are normalized using a robust scaler followed by a min-max scaler to ensure that each variable contributes equally to the model. At the end of this preprocessing pipeline, the dataset is clean, structured, and ready for use in the UEBA models and anomaly detection framework.

Figure 1 illustrates the overall architecture of the proposed UEBA-based anomaly detection framework. The process starts with data collection from multiple sources (e.g., Windows events, emails, antivirus, firewall), which are stored in Splunk Enterprise, a common choice in the industry to collect, index, search, analyse, and visualize large volumes of machine-generated data in real time, as time-series events. The feature extraction pipeline aggregates and processes these events into summary variables and the text data is encoded using Doc2Vec (see Table 1). Once the raw data has been transformed into security events and the features have been computed, this aggregated data is grouped into entities based on business roles (e.g., Customer Management and Executive Positions).

Table 1: Variables collected in the dataset.

Variable	Description	Type
`time`	Date and time when the data was generated. (used for window aggregation and indexing only)	date
`CallerUser`	Username (used for role aggregation and indexing only).	factor
`WorkstationName`	Machine name where the data was generated.	factor
`num_new_process`	Number of new processes created (e.g., Windows event 4688).	numeric
`num_logins`	Number of successful logins (e.g., Windows event 4624).	numeric
`avg_sec_bet_logins`	Average time in seconds between successful logins.	numeric
`num_f_logins`	Number of failed login attempts (e.g., Windows event 4625).	numeric
`avg_sec_bet_f_logins`	Average time between failed logins.	numeric
`num_antivirus_alerts`	Number of incidents detected by the antivirus.	numeric
`num_firewall_alerts`	Number of incidents detected by the firewall.	numeric
`sent_emails`	Number of emails sent by the user.	numeric
`received_emails`	Number of emails received by the user.	numeric
`incident_emails`	Number of emails flagged as incidents.	numeric
`sent_emails_size`	Total size of sent emails (body and attachments).	numeric
`received_emails_size`	Total size of received emails (body and attachments).	numeric
`sent_email_files`	Number of file attachments in sent emails.	numeric
`received_email_files`	Number of file attachments in received emails.	numeric
`sent_email_links`	Number of web links in sent emails.	numeric
`received_email_links`	Number of web links in received emails.	numeric
`4100_events`	Number of PowerShell errors (e.g., Windows event 4100).	numeric
`4104_events`	Number of remote PowerShell commands executed (e.g., Windows event 4104).	numeric

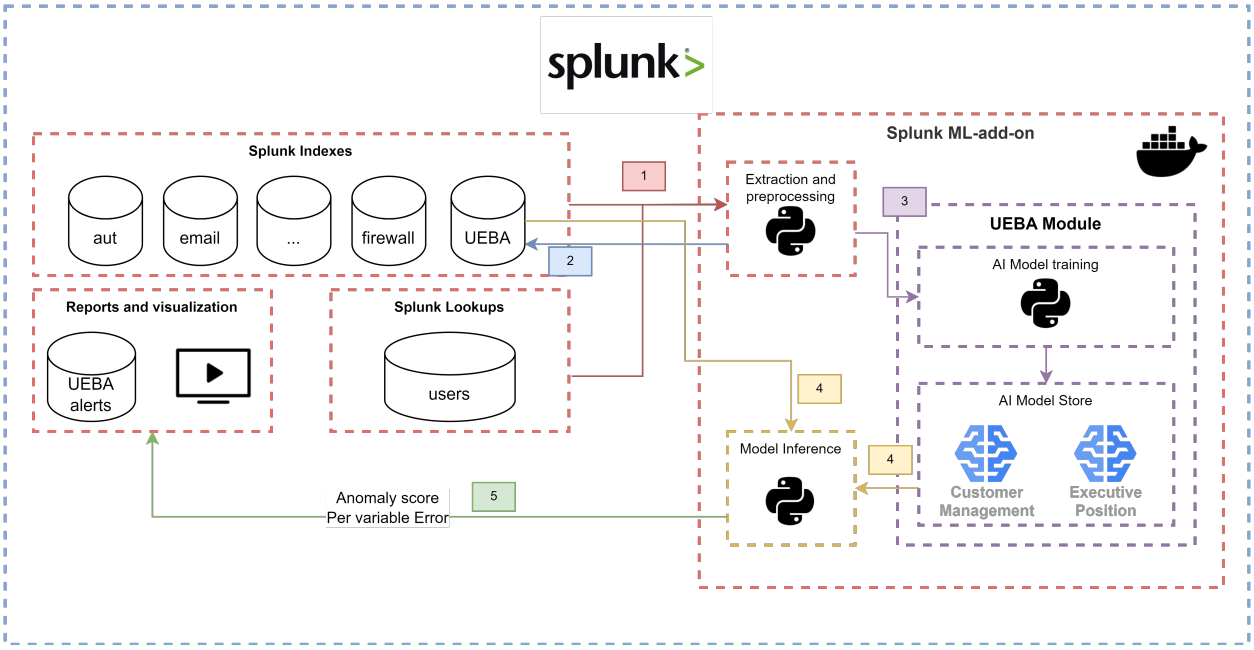


Figure 1: Architecture of the UEBA-based anomaly detector.

We generated two 1 year historical data sets, one for the Customer Management group (25313 records) and the other for the Executive Positions group (9804 records). Both datasets contain records of user behaviour that have been cleared as normal behaviour by the existing security filters from the institution; however, they may contain a small number of anomalies that evaded these measures. For this reason, we assume the data to be contaminated data (Tian et al., 2023), and thus unlabelled. Both datasets are split into training, validation, and testing sets using a standard split: 20% of the records are held out as the testing set, and the remaining 80% is used for training, of which 20% is further reserved as a validation set. We train a separate deep autoencoder model for each of the user groups. The model architecture starts with 83 input features (including the 19 aggregated features and the 64 components of the Doc2Vec transformation) and

compresses them to an 8-dimensional latent space via three hidden layers (with 64, 32, and 16 neurons, respectively). The encoder and decoder networks use ELU activations (except in the first and last layers, which use `tanh`). Training employs the Adam optimiser, early stopping, and L_1 regularisation. A decision threshold is determined as the 95th percentile of the reconstruction error on the validation set.

Trained models are stored using MLFlow and later deployed to analyse incoming data. The anomaly score (based on the reconstruction error) and auxiliary statistics are sent back to Splunk for reporting and further analysis.

3 Results and Discussion

We evaluate the proposed UEBA framework in two complementary settings. First, we conduct experiments on real-world data collected from a financial institution on Section 3.1, providing a strong validation of the proposed framework under real operational conditions. Next, we extend these findings on Section 3.2 using simulated data sampled from real attack scenarios, allowing us to systematically probe the system's response to specific threat vectors and anomalous behaviour of different intensities. By combining these two perspectives –actual enterprise data and controlled simulations– our analysis offers a robust demonstration of how UEBA can enhance security monitoring, highlight anomalous user or entity behaviour, and detect sophisticated cyber threats in an explainable manner.

In the sections that follow, we present the results of these evaluations, focusing on detection rate, residual-space analysis using t-SNE projections, performance under synthetic anomalies, and overall explainability.

3.1 Performance on real data

Evaluating unsupervised models is challenging in the absence of labelled data in real scenarios. To evaluate the anomaly detection capabilities of the proposed framework, we will assess its ability to learn normal behavioural patterns by analysing the positive rate on the test set. The positive rate is set at 5% based on a decision threshold corresponding to the 95th percentile of the reconstruction error on the validation set. Table 2 shows the positive rates for both user groups (Customer Management and Executive Positions). We can observe how positive rates for both groups are close to the 5% value, indicating that the models are well-calibrated and not overly sensitive to noise in the data. This balance helps reduce the risk of false positives while maintaining a high detection sensitivity.

Table 2: Positive rates for the two autoencoder models.

UEBA Model	Positive Rate
Customer Management	5.09%
Executive Positions	4.61%

Moreover, to better understand the model's learning effectiveness, we perform a t-SNE projection of both the original test data and the corresponding residuals. Figure 2 shows a t-SNE projection of normal test data from the Customer Management group (left) and the corresponding residuals (right). We can observe how the residual space shows less dependency clustering compared to the original data, confirming that the model has successfully captured the dominant patterns of normal behaviour. The residual distribution reveals anomalies as scattered points on the edges.

(a) Normal data.

(b) Residuals.

Figure 2: t-SNE representation of Customer Management. (a) Normal test data and (b) corresponding residuals.

3.2 Performance on synthetic anomalies

To assess the model's capability in detecting real-world anomalies, we conducted an experiment with synthetic anomalies generated from 10 real attack scenarios provided by the financial institution (login anomalies, antivirus incidents, email anomalies, and process-related anomalies). These synthetic anomalies are generated by taking convex combinations of the real anomalies with normal behaviour data. This procedure allows us to increase the sample size of the test set, using the variability of normal behaviour to provide more varied anomalies and study the model's detection capability as a function of an anomaly intensity factor, λ_k. Particularly, for each $j = 1, \ldots, 10$, and for $k = 1, \ldots, 100$, we obtain a synthetic test set as follows

$$\mathbf{a}_k^{*j} = \mathbf{z}_j(1 - \lambda_k) + \lambda_k \mathbf{a}_j,$$

where $\mathbf{z}_j$ is a randomly sampled element with normal behaviour data from the test set, $\lambda_k \in [0, 1]$ is the anomaly intensity factor, which takes values in steps of 0.01, and $\mathbf{a}_j$ is a real-type anomaly. Note that, with this procedure, we obtain a synthetic test set of sample size 1000.

Figure 3 and 4 presents the detection rate as a function of the anomaly intensity factor λ for both user groups (Customer Management and Executive Positions). The results demonstrate that the models reliably detect anomalies when $\lambda > 0.7$ for all anomaly types. For specific types, such as login and antivirus anomalies, detection occurs at much lower intensity levels ($\lambda > 0.2$). In contrast, process anomalies require higher intensity levels for reliable detection, primarily due to the complexity of encoding text-based data.

Figure 5 shows a t-SNE embedding of the test set. Each colour represents a different anomaly type (login, email, antivirus and process), with colour and saturation indicating the type and intensity of anomalies. The results confirm that anomalies become increasingly distinguishable as their intensity (λ) increases. Notably, anomalies with high λ values form clear clusters in the residual space, confirming the model's capability to separate abnormal behaviour from normal patterns. Figure 5a illustrates how synthetic anomalies are embedded alongside normal data in the original feature space, while Figure 5b shows the corresponding residuals, where anomalies show clearer clusters, thus being easier to identify. Process-related anomalies appear less distinct due to their complexity and dependence on text-based feature encoding.

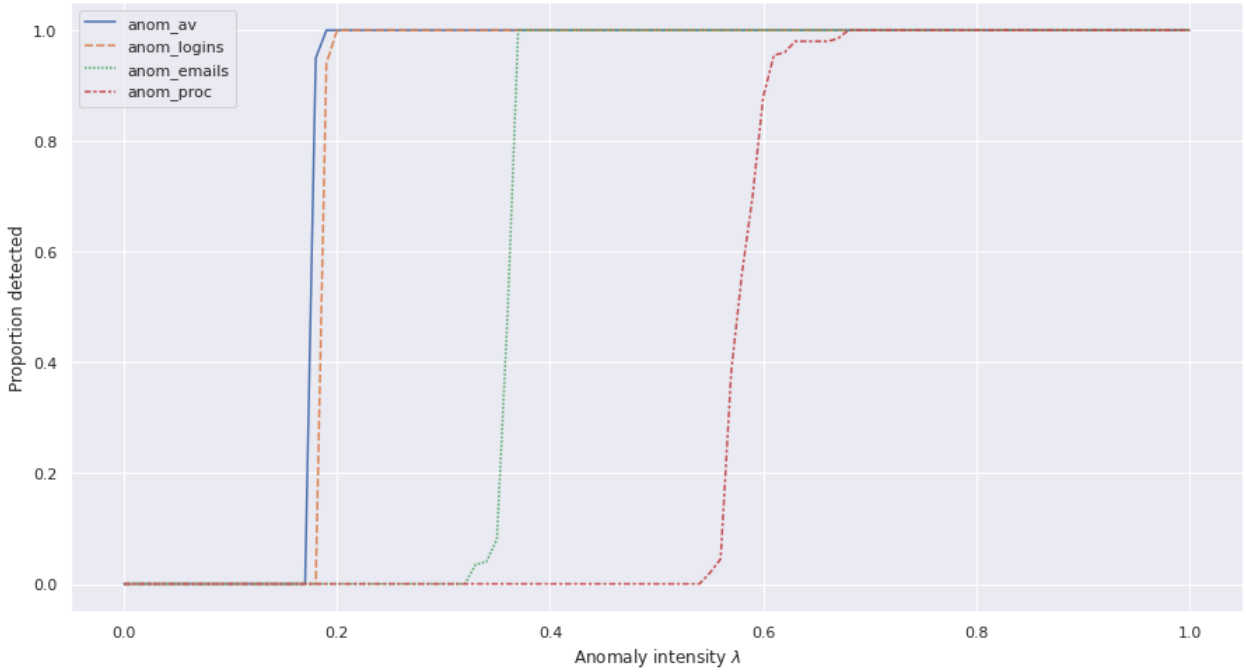


Figure 3: Anomaly detection rates as a function of anomaly intensity for each model for the Customer Management model.

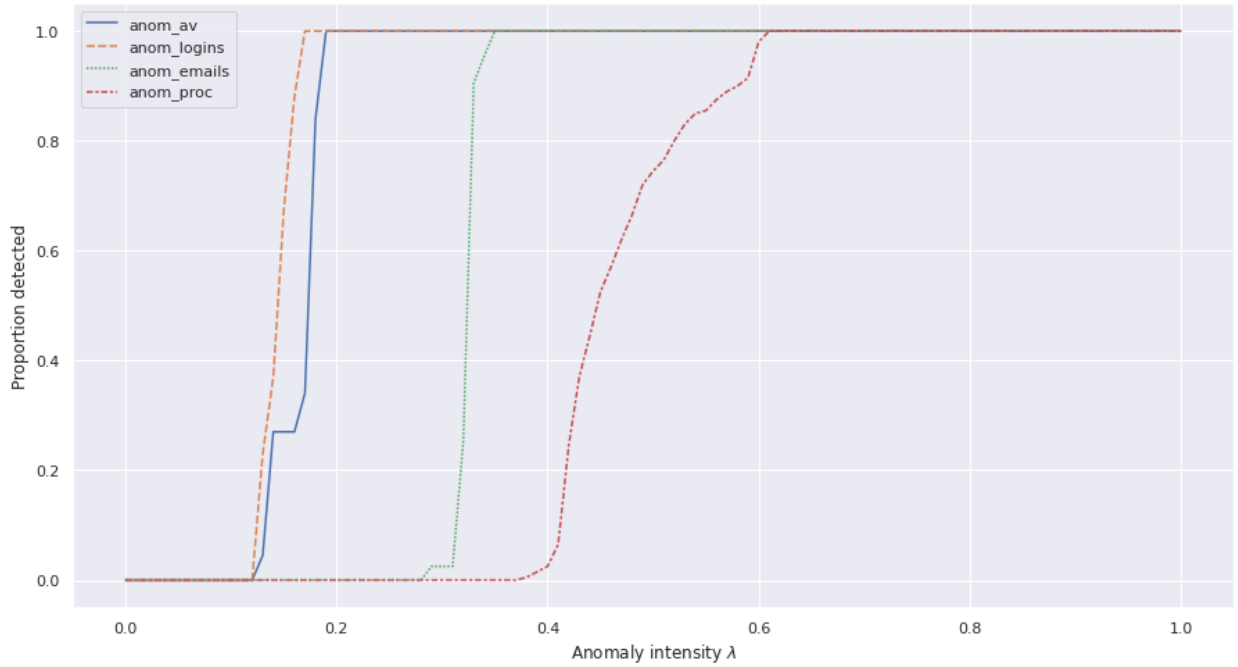


Figure 4: Anomaly detection rates as a function of anomaly intensity for each model for the Executive Positions model.

At last, we will assess the proposed methodology's ability to provide explainable model results through the per-feature reconstruction error. Figure 6 shows the logarithm of the per-feature reconstruction error for each model for fully anomalous data ($\lambda = 1$). We can observe how that are easier to detect and show this effect more clearly, with higher errors on features related to the anomaly. For instance, the email-related anomalies show the highest reconstruction error on the `sent_email_*` variables. In addition, for the login anomalies, we get the highest error on the antivirus and login variables.

However, for anomalies that are more challenging, such as process anomalies, the per-variable reconstruction error alone may not suffice in identifying the origin of the anomaly. We speculate that the reason for these anomalies being harder to detect and explain is that they heavily depend on the process encoding by Doc2Vec. However, we have to remark that, when removing the text variables, the detection performance worsened, so the text encoding is providing valuable information for the detection of these anomalies, even if itś not enough for clear interpretation.

We can observe how features related to login patterns, email attachment size, and failed login attempts exhibit the highest contributions to the anomaly score, offering clear indications of potential security incidents. Anomalies in email activity, such as unusually large attachments or an abnormal volume of sent emails, are immediately recognizable. Process-related anomalies, on the other hand, show a more distributed error pattern due to the diverse and complex nature of process command sequences.

These results highlight the strength of the proposed framework in detecting a wide range of anomalies with high accuracy while being explainable. The combination of Deep Autoencoders and Doc2Vec allows for an effective integration of numerical and text-based features, providing a comprehensive view of user behaviour. The ability to visualize residuals and analyse feature-level contributions significantly enhances explainability, making the framework more practical and effective for its use in cybersecurity applications.

4 Conclusions

This study presented a UEBA-based anomaly detection framework that leverages Deep Autoencoders to identify suspicious activities in a real cybersecurity use-case of a financial institution. By integrating both numerical and text-based features through Doc2Vec embeddings, the proposed approach can capture complex behavioural patterns to detect anomalies that may otherwise remain undetected by more traditional methods. Among the main contributions is a novel theoretical result that proves the equivalence of two common definitions of fully-connected neural network, and a set of experimental evaluations to showcase how advanced deep learning techniques can be employed for explainable, behaviour-based anomaly detection.

Experimental evaluations showed that the the proposed anomaly detection framework achieves a high detection rate, even in challenging conditions with contaminated training data. Additional experiments with synthetic anomalies–reflecting anomalous login patterns, email activity anomalies, and antivirus alerts–confirmed the framework's robustness and adaptability to diverse cybersecurity threats. A key advantage of our method lies in the ability to perform a residual-based analysis, which enhances explainability by pinpointing specific features that deviate from the normal baseline profile. This explainability is essential in practical cybersecurity contexts where timely responses and a deep understanding of anomalies can significantly improve investigation efficiency and reduce false alarms.

While the proposed framework demonstrated a strong performance, several areas offer opportunities for future improvement. Enhancing text-based feature encoding with more advanced language models (e.g., transformer-based architectures) could improve the detection of process-related anomalies or expand the approach to other text-based sources. Furthermore, integrating explainability techniques like Shapley values and attention mechanisms can provide more insights into feature contributions. Another promising direction is the use of ensemble models to combine multiple anomaly detection algorithms, improving overall performance and reliability.

Author contributions

Conceptualization, J.F, I.O-F, N.M.V and M.S; software, J.F; validation, I.O-F, N.M.V and M.S; formal analysis, J.F, I.O-F, N.M.V and M.S; investigation, J.F; resources, I.O-F, M.S; data curation, J.F; writing-original draft, J.F, I.O-F, N.M.V; writing-review & editing, I.O-F, M.S, and N.M.V; visualization, J.F; supervision, I.O-F, M.S, and N.M.V; project administration, I.O-F; funding acquisition, I.O-F, M.S.

Acknowledgments

This work was supported by the Ayudas Cervera para Centros Tecnológicos grant of the Spanish Centre for the Development of Industrial Technology (CDTI) under project CICERO (CER-20231019) and by the grant PID2020-118101GB-I00 from the Ministerio de Ciencia e InnovaciÃşn (MCIN/AEI/10.13039/501100011033).

References

Cano-Córdoba, F., Sarma, S., and Subirana, B. (2017). Theory of intelligence with forgetting: Mathematical theorems explaining human universal forgetting using "forgetting neural networks". Technical report, Center for Brains, Minds and Machines (CBMM).

Flynn, J. S., Giannetti, C., and Van Dijk, H. (2023). Anomaly detection of dc nut runner processes in engine assembly. *AI*, 4(1):234–254.

Gonzalez-Muniz, A., Diaz, I., Cuadrado, A. A., Garcia-Perez, D., and Perez, D. (2022). Two-step residual-error based approach for anomaly detection in engineering systems using variational autoencoders. *Computers and Electrical Engineering*, 101:108065.

Görnitz, N. (2019). *One-Class Classification in the Presence of Point, Collective, and Contextual Anomalies / Ein-Klassen-Klassifikation in Der Gegenwart von Punkt-, Kollektiv- Und Kontext-Anomalien*. PhD thesis, Technische Universität Berlin. AAI27610132.

Hawkins, S., He, H., Williams, G., and Baxter, R. (2002). Outlier detection using replicator neural networks. In Kambayashi, Y., Winiwarter, W., and Arikawa, M., editors, *Data Warehousing and Knowledge Discovery*, pages 170–180, Berlin, Heidelberg. Springer Berlin Heidelberg.

Kidger, P. and Lyons, T. (2020). Universal approximation with deep narrow networks.

Le, Q. V. and Mikolov, T. (2014). Distributed representations of sentences and documents.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867.

Maher, D. (2017). Can artificial intelligence help in the war on cybercrime? *Computer Fraud & Security*, 2017(8):7–9.

Martin, A. G., Beltran, M., Fernandez-Isabel, A., and de Diego, I. M. (2021). An approach to detect user behaviour anomalies within identity federations. *Computers & Security*, 108:102356.

Mauritz, R. R., Nijweide, F. P. J., Goseling, J., and van Keulen, M. (2021). A probabilistic database approach to autoencoder-based data cleaning.

Meng, W., Wang, Y., Wong, D. S., Wen, S., and Xiang, Y. (2018). Touchwb: Touch behavioral user authentication based on web browsing on smartphones. *Journal of Network and Computer Applications*, 117:1–9.

Mhaskar, H. and Poggio, T. (2016). Deep vs. shallow networks : An approximation theory perspective.

Mhaskar, H. N. and Poggio, T. (2019). Function approximation by deep networks.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space.

Morales-Forero, A. and Bassetto, S. (2019). Case study: A semi-supervised methodology for anomaly detection and diagnosis. In *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1031–1037.

Ortega-Fernandez, I., Sestelo, M., Burguillo, J. C., and Piñón-Blanco, C. (2023). Network intrusion detection system for ddos attacks in ics using deep autoencoders. *Wireless Networks*.

Ortega-Fernandez, I., Sestelo, M., and Villanueva, N. M. (2024). Explainable generalized additive neural networks with independent neural network training. *Statistics and Computing*, 34(1):6.

Pusara, M. and Brodley, C. E. (2004). User re-authentication via mouse movements. In *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, pages 1–8, New York, NY, USA. Association for Computing Machinery.

Ribeiro, M., Lazzaretti, A. E., and Lopes, H. S. (2018). A study of deep convolutional auto-encoders for anomaly detection in videos. *Pattern Recognition Letters*, 105:13–22.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation, parallel distributed processing, explorations in the microstructure of cognition. *Biometrika*, 71:599–607.

Sakurada, M. and Yairi, T. (2014). Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, pages 4–11, New York, NY, USA. Association for Computing Machinery.

Shashanka, M., Shen, M.-Y., and Wang, J. (2016a). User and entity behavior analytics for enterprise security. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1867–1874.

Shashanka, M., Shen, M.-Y., and Wang, J. (2016b). User and entity behavior analytics for enterprise security. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1867–1874.

Slipenchuk, P. and Epishkina, A. (2019). Practical user and entity behavior analytics methods for fraud detection systems in online banking: A survey. *Advances in Intelligent Systems and Computing*, pages 83–93.

Tian, B., Su, Q., and Yu, J. (2023). Leveraging contaminated datasets to learn clean-data distribution with purified generative adversarial networks.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

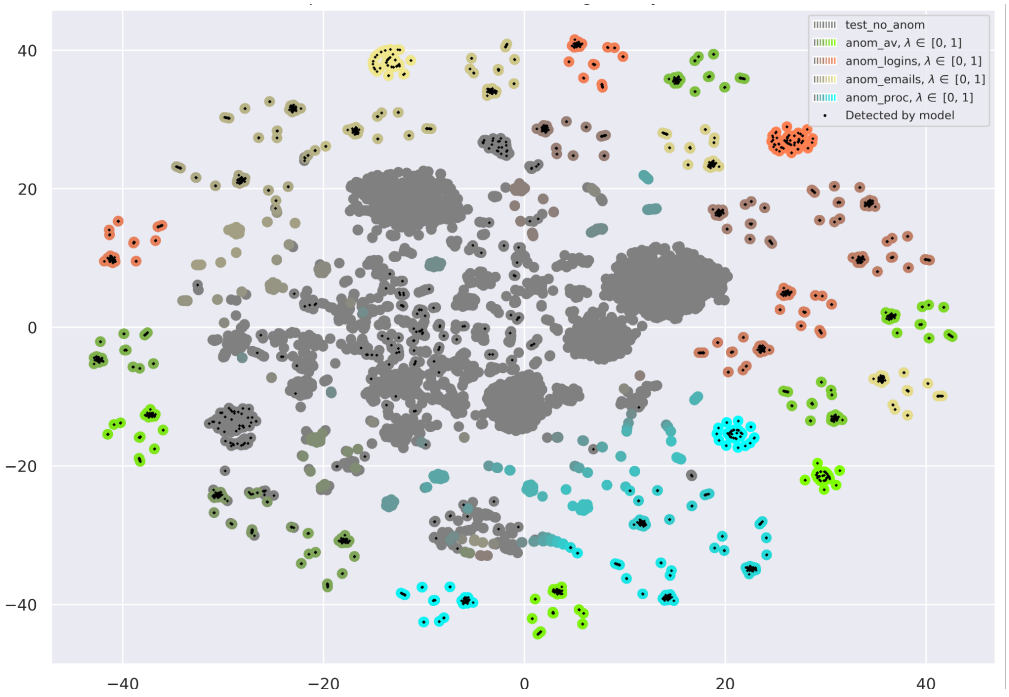
Voris, J., Song, Y., Salem, M. B., Hershkop, S., and Stolfo, S. (2019). Active authentication using file system decoys and user behavior modeling: results of a large scale study. *Computers & Security*, 87:101412.

Wang, H., Bah, M. J., and Hammad, M. (2019). Progress in outlier detection techniques: A survey. *IEEE Access*, 7:107964–108000.

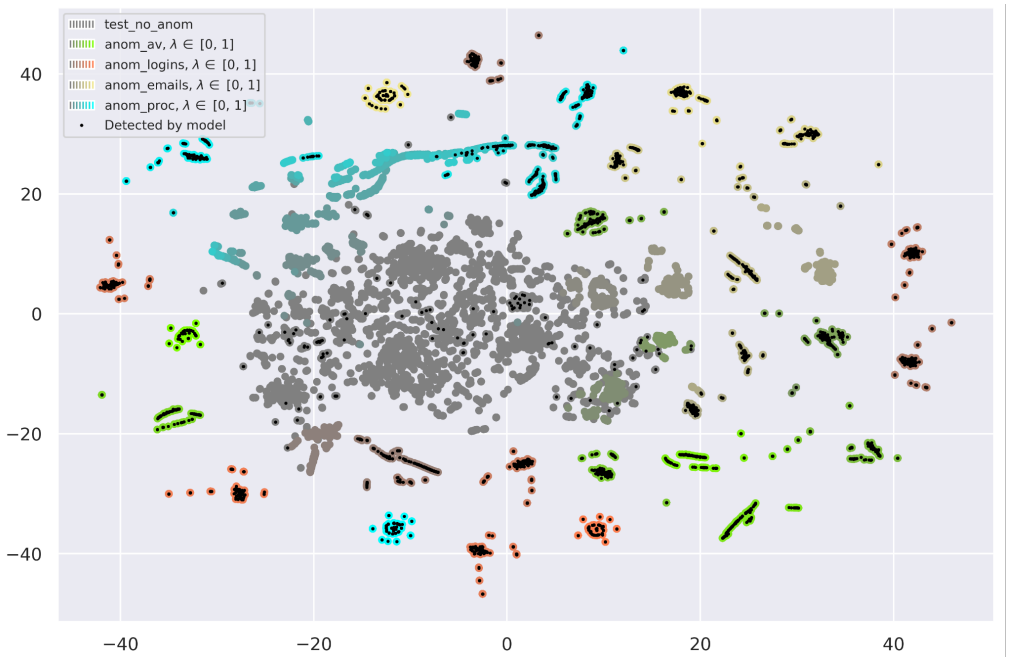
Wang, X., Pi, D., Zhang, X., Liu, H., and Guo, C. (2022). Variational transformer-based anomaly detection approach for multivariate time series. *Measurement*, 191:110791.

Xiao, Z., Yan, Q., and Amit, Y. (2020). Likelihood regret: An out-of-distribution detection score for variational auto-encoder.

Zhou, C. and Paffenroth, R. C. (2017). Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 665–674, New York, NY, USA. Association for Computing Machinery.

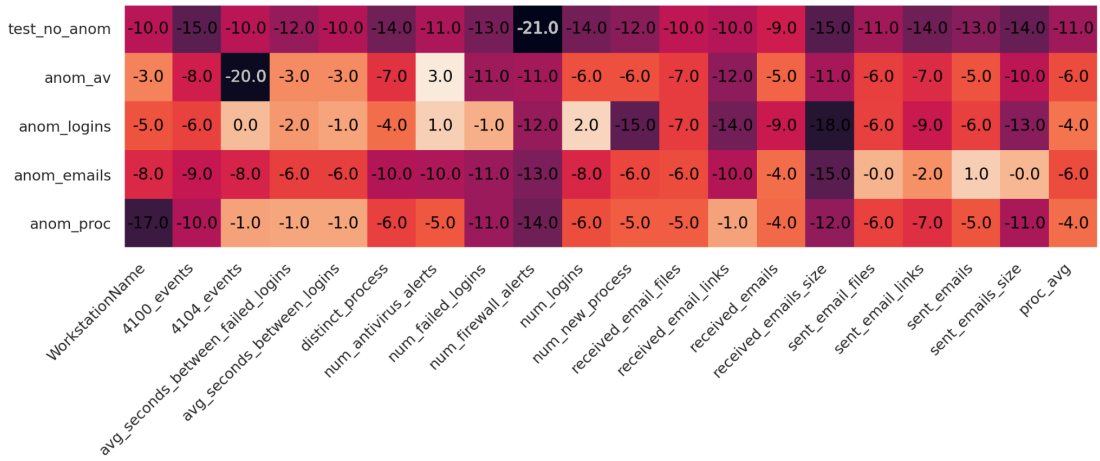


(a) Sample of test data with anomalies.



(b) Corresponding residuals.

Figure 5: t-SNE representations of data and residuals for the Customer Management group. Dotted points indicate instances flagged as anomalies. Saturation indicates the intensity of anomalies.



(a) Customer Management.

(b) Executive Positions.

Figure 6: Logarithm of the reconstruction error per feature for each model.