Universiteit Leiden

# Master Computer Science

Behavior-Based Broadcast Piracy Detection Using Weakly Supervised Anomaly Detection


| | |
|---|---|
| Name: | Xiangyuan Tao |
| Student ID: | s4256778 |
| Date: | [15/6/2026] |
| Specialisation: | Artificial Intelligence |
| 1st supervisor: | Daan Pelt |
| 2nd supervisor: | Hazel Doughty |


Master's Thesis in Computer Science


Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

# Abstract

Broadcast piracy detection is an important but challenging problem for subscription-based media services. In practice, the task is not simply to classify customers as piracy or non-piracy, but to rank accounts by risk so that a limited investigation team can focus on the most suspicious cases. This work studies piracy detection from customer subscription behavior as a weakly supervised anomaly ranking problem. Multiple operational tables, including customer, financial-account, subscription-event, migration, and package-channel records, are integrated into a customer-level dataset in which each row represents one customer. The final dataset is highly imbalanced and only partially labeled, with confirmed piracy accounts available for only a small subset of customers.

Several anomaly-oriented models are investigated under the same evaluation setting. First, a reconstruction-based tabular autoencoder is used as a baseline. Next, two weakly supervised anomaly detection methods, FeaWAD and PReNet, are applied. Finally, a unified hybrid model is proposed to combine FeaWAD-style reconstruction-aware feature encoding with PReNet-style pairwise ranking supervision within a shared architecture.

The results show that the baseline autoencoder provides only limited early-ranking performance, although it captures some useful anomaly signal. In contrast, the weakly supervised methods substantially improve ranking quality. FeaWAD and PReNet both outperform the baseline by a large margin, and the unified hybrid model achieves the strongest performance over most practically relevant top-$K$ budgets. These findings indicate that customer subscription behavior contains meaningful piracy-related signals, that weak supervision is highly valuable in this setting, and that combining reconstruction-based and pairwise ranking objectives is a promising direction for anti-piracy investigation support.

# Contents

# 1 Introduction

Broadcast television piracy poses a persistent challenge for subscription-based media services [1, 2, 3]. Service providers distribute encrypted content to authorized customers through controlled access infrastructures, but unauthorized access and redistribution remain common in practice. Piracy may take several forms, such as signal interception, tampering with access mechanisms, card sharing, and unauthorized rebroadcasting or restreaming of protected content. These activities can cause financial losses for operators, undermine legitimate subscription models, and reduce the effectiveness of existing content-protection measures [4, 5]. In large-scale operational settings, where millions of customer accounts may need to be monitored, purely manual investigation is costly and difficult to scale. As a result, there is a strong practical need for data-driven methods that can screen customer behavior and prioritize suspicious accounts for further investigation [6, 7, 8].

Traditional anti-piracy actions, such as manual investigation, technical enforcement, and legal countermeasures, remain necessary but are often reactive, slow, and difficult to scale [9]. In real-world settings, service providers may need to monitor millions of accounts, while only a limited number of suspicious cases can be investigated in depth. For this reason, the practical objective is often not to make a final automated decision, but to produce a ranked list of suspicious accounts so that limited investigative resources can be focused on the most promising cases.

This thesis addresses the problem of detecting piracy from users' subscription behavior. More specifically, the task is to identify suspicious customers from large-scale behavioral and account-level records, using only partial piracy labels. This setting is challenging for several reasons. First, confirmed piracy cases are rare compared with the full customer population. Second, the unlabeled population cannot be assumed to be fully normal, because it may contain undetected piracy cases. Third, piracy is not directly observable from the available data and must instead be inferred indirectly from behavioral proxies derived from subscription history, product changes, account characteristics, and usage patterns.

These characteristics make the problem closely related to anomaly detection, especially weakly supervised anomaly detection [10]. Prior work on anomaly detection has shown that rare and heterogeneous abnormal behavior is difficult to detect reliably, particularly under strong class imbalance and incomplete labels [11, 12, 13]. Autoencoder-based anomaly detection methods are often used as a baseline because they can learn compact representations of normal behavior and detect deviations through reconstruction error [14, 15]. However, reconstruction-based methods are often limited when the training data are contaminated or when the anomalies of interest are subtle. To address such limitations, weakly supervised anomaly detection methods have been proposed to exploit a small set of labeled anomalies together with a large unlabeled dataset [16]. In particular, methods such as DevNet [17], FeaWAD [18], and PReNet [19] show that partial anomaly labels can be used more effectively through deviation-based learning or pairwise relation learning.

Motivated by this literature, this thesis studies broadcast piracy detection as a weakly supervised anomaly ranking problem on large-scale tabular customer-behavior data. The overall approach is to transform raw operational records, including subscription, account, migration, and product information, into customer-level behavioral features, and then evaluate anomaly-

oriented models for ranking suspicious customers. A reconstruction-based autoencoder is first used as an unsupervised reference model. Next, weakly supervised anomaly detection methods, including FeaWAD and PReNet, are investigated to examine whether limited piracy labels can improve ranking performance. Finally, a unified FeaWAD–PReNet hybrid model is proposed to study whether reconstruction-aware feature encoding and pairwise ranking supervision can be combined effectively for piracy detection.

The main research question of this thesis is:

> *How effectively can weakly supervised anomaly detection methods identify and rank suspicious customers for broadcast piracy detection in a large-scale, highly imbalanced, and partially labeled operational dataset?*

This main question is addressed through the following research questions:

- **RQ1:** How can large-scale raw tabular business data be integrated, processed, and transformed into customer-level behavioral features suitable for anomaly detection?
- **RQ2:** To what extent can anomaly detection models identify piracy-related behavioral patterns under production-scale data, strong class imbalance, and partial labeling?
- **RQ3:** Can a new weakly supervised anomaly detection model be designed to improve suspicious-customer ranking beyond existing weakly supervised methods in this application setting?

The main contributions of this thesis are:

- **C1:** A production-scale data processing pipeline is developed to transform large-scale raw tabular business data into customer-level behavioral features suitable for anomaly detection.
- **C2:** The broadcast piracy detection task is formulated as a weakly supervised anomaly ranking problem under strong class imbalance, partial labeling, and potentially contaminated unlabeled data.
- **C3:** A comparative empirical study is conducted to evaluate reconstruction-based and weakly supervised anomaly detection models, including an autoencoder baseline, FeaWAD, and PReNet, under a unified ranking-based evaluation setting.
- **C4:** A new weakly supervised anomaly detection model is proposed by combining reconstruction-aware feature encoding with pairwise ranking supervision in a shared architecture.
- **C5:** The thesis provides practical insight into how anomaly ranking can support anti-piracy investigation by prioritizing suspicious customers under limited inspection capacity.

The remainder of this thesis is organized as follows. Section 2 reviews the literature most relevant to this work. Section 3 describes the data used in the project. Section 4 presents the methods developed in this thesis. Section 5 and Section 6 report the experimental settings and results. Section 7 discusses the findings, implications, and limitations. Finally, Section 8 concludes the thesis and outlines directions for future work.

# 2 Related Work

This section reviews the literature most relevant to this thesis. The problem studied in this work is closely related to anomaly detection, because confirmed piracy customers are rare, heterogeneous, and only partially observed. More specifically, the task belongs to weakly supervised anomaly detection, since a small set of confirmed piracy labels is available while the majority of customers remain unlabeled. The discussion first introduces anomaly detection and deep anomaly detection in general, then focuses on reconstruction-based methods and weakly supervised methods that are most relevant to the models evaluated in this thesis.

## 2.1 Anomaly detection and deep anomaly detection

Anomaly detection aims to identify observations that deviate substantially from the dominant patterns in a dataset. Given an input sample $x \in \mathbb{R}^d$, most anomaly detection methods can be viewed as learning or defining an anomaly scoring function

$$s : \mathbb{R}^d \to \mathbb{R},$$

where a larger value of $s(x)$ indicates that the sample is more likely to be anomalous. This formulation is especially suitable for applications where the goal is not necessarily to make an automatic binary decision, but to rank suspicious cases for further inspection.

Classical anomaly detection methods include statistical, distance-based, density-based, and classification-based approaches [20]. Statistical methods usually assume that normal samples follow a certain probability distribution and identify low-probability observations as anomalies [21, 22]. Distance-based methods regard samples as anomalous when they are far from their nearest neighbors or from the majority of the data [23, 24]. Density-based methods compare the local density of a sample with that of its surrounding region [25, 26]. Classification-based methods instead attempt to learn a decision boundary that separates normal and abnormal regions, or that describes the normal class compactly [27, 28].

These classical methods provide the foundation for anomaly detection, but they often depend on manually designed distance functions, density assumptions, or shallow feature representations. Such assumptions can be restrictive in high-dimensional and structurally complex data. This limitation is relevant to customer subscription behavior, where suspicious patterns may be distributed across many related features, such as subscription changes, account characteristics, migration behavior, product exposure, and temporal activity patterns.

Deep anomaly detection addresses this limitation by learning representations directly from data. Instead of applying anomaly scores only to raw input features, deep methods first learn a representation

$$h = f_\theta(x),$$

and then define or learn an anomaly score based on this representation. Existing deep anomaly detection methods are commonly grouped into reconstruction-based, one-class, self-supervised, adversarial, and hybrid approaches [29]. The advantage of this representation-learning view is that complex behavioral patterns can be captured in a latent space, rather than relying only on hand-crafted feature distances.

For the present thesis, two families of deep anomaly detection methods are especially relevant. The first is reconstruction-based anomaly detection, represented by autoencoders, where abnormality is measured through reconstruction error. This provides a natural unsupervised baseline when labels are scarce. The second is weakly supervised anomaly detection, where a small set of labeled anomalies is used together with a large unlabeled population. This setting matches the piracy detection problem studied in this thesis, because only a limited number of confirmed piracy customers are available, while the remaining customers are unlabeled rather than guaranteed to be normal.

## 2.2 Reconstruction-based anomaly detection

Reconstruction-based anomaly detection is one of the most widely used families of deep anomaly detection methods [30, 31, 32]. The basic assumption is that a model trained to reconstruct common patterns in the data will reconstruct normal samples well, while anomalous samples will be reconstructed less accurately. Autoencoders are a typical implementation of this idea.

Given an input sample $x \in \mathbb{R}^d$, an autoencoder consists of an encoder and a decoder. The encoder maps the input into a latent representation,

$$h = f_{\text{enc}}(x), \tag{1}$$

and the decoder reconstructs the input from this representation,

$$\hat{x} = f_{\text{dec}}(h). \tag{2}$$

The model is usually trained by minimizing the reconstruction loss

$$\mathcal{L}_{\text{rec}} = \|x - \hat{x}\|_2^2, \tag{3}$$

or a normalized variant of this loss. After training, the reconstruction error can be used as an anomaly score:

$$s(x) = \|x - \hat{x}\|_2^2. \tag{4}$$

Samples with larger reconstruction errors are regarded as more suspicious, because they are less well represented by the learned reconstruction function.

This approach is attractive in settings where anomaly labels are scarce or unavailable, because the model can be trained without requiring a large labeled anomaly set. It is also suitable for tabular behavioral data, where the latent representation may capture common customer behavior patterns and the reconstruction error may reveal deviations from these patterns [14, 15].

However, reconstruction error is only an indirect proxy for abnormality. A sample may receive a high reconstruction error because it is rare, noisy, or difficult to reconstruct, even if it is not related to the target anomaly. Conversely, some anomalous samples may be reconstructed well if they share common feature patterns with the dominant population, or if the training data are contaminated by hidden anomalies. This limitation is especially relevant in weakly supervised settings, where the unlabeled population cannot be assumed to be completely normal.

For this reason, reconstruction-based methods provide a useful baseline, but they may be insufficient when the practical goal is to rank a small number of high-risk cases near the top of a large population. This motivates weakly supervised methods that use limited anomaly labels to shape the scoring function more directly.

## 2.3 Weakly supervised anomaly detection

Weakly supervised anomaly detection addresses settings in which only limited anomaly labels are available, while the majority of samples remain unlabeled. A recent survey organizes weak supervision in anomaly detection into incomplete, inexact, and inaccurate supervision [16]. The setting studied in this thesis mainly corresponds to incomplete supervision: only a small subset of piracy cases is confirmed, while the remaining customer population is unlabeled rather than verified normal.

Let $\mathcal{A}$ denote the set of labeled anomalies and $\mathcal{U}$ denote the unlabeled set. In this setting, $\mathcal{A}$ is usually much smaller than $\mathcal{U}$, and $\mathcal{U}$ may still contain hidden anomalies. Therefore, it is inappropriate to treat all unlabeled samples as clean normal data. Instead, weakly supervised anomaly detection methods aim to use the limited labeled anomalies to guide representation learning or score learning, while still exploiting the large unlabeled population.

Compared with purely unsupervised reconstruction-based methods, weakly supervised methods can use labeled anomalies to shape the anomaly score more directly. For example, Deep SAD extends one-class representation learning by incorporating limited labeled anomaly information [33]. It learns a representation in which normal samples are encouraged to stay close to a center in latent space, while labeled anomalies are pushed away from that center. This illustrates the general value of weak supervision: even a small number of labeled anomalies can provide useful information about the direction in which abnormal behavior should be separated.

This property is particularly relevant to the piracy detection task. Confirmed piracy labels are rare, but they are operationally meaningful. The key question is therefore how to use these limited labels without assuming that the much larger unlabeled set is completely normal. The following subsections discuss two representative weakly supervised directions that are central to this thesis: deviation-based learning and pairwise relation learning.

### 2.3.1 Deviation-based learning: DevNet and FeaWAD

Deviation Networks, or DevNet, directly optimize anomaly scores using a small number of labeled anomalies and a reference score distribution [17]. Instead of relying only on a reconstruction error or a post-hoc anomaly score, DevNet learns a scoring function

$$s(x) = f_\theta(x),$$

where larger values indicate stronger abnormality.

The main idea is to compare each sample score with a reference distribution that represents typical score values. Let $\mu$ and $\sigma$ denote the mean and standard deviation of this reference score distribution. The normalized deviation score is defined as

$$d(x) = \frac{s(x) - \mu}{\sigma}. \tag{5}$$

A larger positive value of $d(x)$ means that the sample receives a score far above the reference distribution and is therefore more likely to be anomalous.

Given a binary label $y \in \{0, 1\}$, where $y = 1$ denotes a labeled anomaly and $y = 0$ denotes an unlabeled or reference sample, the deviation-based loss can be written as

$$\mathcal{L}_{\text{dev}} = (1 - y)|d(x)| + y \max(0, a - d(x)), \tag{6}$$

where $a > 0$ is a margin parameter. For unlabeled samples, the term $|d(x)|$ encourages scores to remain close to the reference distribution. For labeled anomalies, the hinge term $\max(0, a - d(x))$ encourages their normalized scores to exceed the margin $a$. In this way, DevNet explicitly pushes labeled anomalies into the high-score region instead of treating anomaly scoring as an indirect consequence of representation learning.

FeaWAD builds on this idea by combining deviation-based supervision with autoencoder-derived feature representations [18]. Rather than using only the scalar reconstruction error, FeaWAD constructs an anomaly representation from three components: the latent representation, the feature-wise reconstruction residual, and the overall reconstruction magnitude.

Following the autoencoder formulation introduced in Equation (1) and (2), let $h$ denote the latent representation and $\hat{x}$ denote the reconstruction of input $x$. FeaWAD defines the reconstruction residual and reconstruction magnitude as

$$r = x - \hat{x}, \qquad e = \|x - \hat{x}\|_2. \tag{7}$$

These components are then concatenated into the FeaWAD representation:

$$z = [h, r, e]. \tag{8}$$

A scoring network maps this representation to an anomaly score:

$$s(x) = f_{\text{score}}(z). \tag{9}$$

This representation is more specific than a scalar reconstruction error. The latent vector $h$ captures the compressed behavioral pattern of the input, the residual vector $r$ preserves feature-wise deviation information, and the scalar $e$ summarizes the overall reconstruction magnitude. Therefore, FeaWAD can distinguish not only how large the reconstruction error is, but also which feature dimensions contribute to the deviation. This is useful for tabular behavioral data, where suspicious behavior may appear through specific combinations of customer-level features rather than through a uniformly large reconstruction error.

Overall, DevNet and FeaWAD show how limited anomaly labels can be used to directly shape the anomaly score space. DevNet introduces deviation-based score supervision, while FeaWAD enriches the input to the scoring function with reconstruction-aware features. These ideas are highly relevant to this thesis because the piracy detection task contains a small number of confirmed piracy labels and a large unlabeled customer population.

### 2.3.2 Pairwise relation learning: PReNet

PReNet represents another direction in weakly supervised anomaly detection. Instead of learning only from individual samples, PReNet learns from pairwise relations between labeled anomalies and unlabeled samples [19]. The motivation is that, when labeled anomalies are scarce, constructing pairs can provide richer supervision than treating each labeled anomaly independently. This is especially useful when the goal is to learn relative abnormality, namely whether one sample should be ranked as more suspicious than another.

Given an input sample $x$, PReNet first maps it into a latent representation:

$$h = f_{\text{enc}}(x). \tag{10}$$

For a pair of samples $(x_i, x_j)$, with latent representations $h_i$ and $h_j$, PReNet constructs a pairwise relation representation by combining the two embeddings, their absolute difference, and their element-wise product:

$$\phi(h_i, h_j) = [h_i,\ h_j,\ |h_i - h_j|,\ h_i \odot h_j], \tag{11}$$

where $\odot$ denotes element-wise multiplication. This representation allows the model to capture both the individual properties of each sample and their interaction patterns.

PReNet usually contains two related output branches. The first branch assigns an anomaly score to an individual sample:

$$s(x) = f_{\text{score}}(h). \tag{12}$$

The second branch predicts the relation between a pair of samples:

$$r_{ij} = f_{\text{rel}}(\phi(h_i, h_j)). \tag{13}$$

The individual anomaly score is used for final ranking, while the relation branch provides additional pairwise supervision during training.

A central component of PReNet is pairwise ranking supervision. Let $x_a$ denote a labeled anomaly and $x_u$ denote an unlabeled sample. Their anomaly scores are $s_a = s(x_a)$ and $s_u =$

$s(x_u)$. A margin-based ranking loss can be written as

$$\mathcal{L}_{\text{rank}} = \max(0, m - (s_a - s_u)), \tag{14}$$

where $m > 0$ is a margin. This loss encourages labeled anomalies to receive higher scores than unlabeled samples by at least margin $m$. Compared with reconstruction-based objectives, this learning signal is more directly aligned with anomaly ranking, because it explicitly optimizes the ordering between suspicious and background samples.

In addition to ranking supervision, PReNet uses relation learning to distinguish different pair types, such as anomaly–unlabeled pairs and unlabeled–unlabeled pairs. This encourages the latent representation to capture relational patterns between abnormal and background samples, instead of relying only on individual sample scores. As a result, pairwise relation learning can improve generalization when only a small number of labeled anomalies are available.

For the piracy detection problem studied in this thesis, the pairwise learning idea is particularly relevant. The final objective is to rank customers by suspicion, not only to reconstruct customer behavior or classify each customer independently. PReNet therefore provides a useful weakly supervised baseline because its training objective is closer to the operational goal of prioritizing suspicious customers for investigation.

### 2.3.3 Recent extensions

Recent work has extended weakly supervised anomaly detection beyond basic score supervision and pairwise learning. One important direction is the incorporation of external knowledge. KDAlign introduces expert-rule knowledge into weakly supervised anomaly detection by aligning knowledge-based signals and data-driven representations through optimal transport [34]. This type of approach is relevant for practical security and anti-abuse applications, where labeled anomalies may be scarce but domain knowledge is often available.

Another direction is transfer learning across related domains. Recent work has combined adversarial transfer learning with weakly supervised anomaly detection in order to improve anomaly separation when labeled anomalies are limited in the target domain [35]. This is useful when anomaly patterns may differ across datasets or operational environments, but some transferable structure still exists.

Other studies have explored richer forms of supervision. ROSAS proposes a semi-supervised anomaly detection framework that is designed to be robust to contamination in the unlabeled set [36]. Instead of relying only on binary labels, it constructs continuous supervisory signals by diffusing the abnormality of labeled anomalies. This idea is relevant to weakly supervised settings because unlabeled data may contain hidden anomalies and should not always be treated as purely normal.

Recent work has also investigated how to shape the latent representation more explicitly. WSAD-DT uses two centroids, one for normal samples and one for anomalies, together with a dual-tailed kernel design [37]. The aim is to compact samples within the same class while preserving stronger separation between normal and anomalous regions.

Overall, these extensions show that weakly supervised anomaly detection is moving toward richer supervision, stronger representation learning, and better use of auxiliary knowledge.

## 2.4 Summary and research gap

The literature reviewed above shows a clear development from classical anomaly detection to deep representation-based anomaly detection, and further to weakly supervised methods that can exploit limited anomaly labels. Classical anomaly detection provides the general formulation of identifying rare deviations from normal patterns [20], while deep anomaly detection improves this setting by learning representations directly from data [29]. Reconstruction-based methods, especially autoencoders, are useful when labels are scarce, but their anomaly scores are usually based on reconstruction error, which is only an indirect indicator of abnormality.

Weakly supervised anomaly detection addresses this limitation by using a small number of labeled anomalies to guide score learning or representation learning [16]. DevNet introduces deviation-based score supervision, FeaWAD combines this idea with reconstruction-aware feature encoding, and PReNet uses pairwise relation learning to improve anomaly ranking [17, 18, 19]. These methods are particularly relevant to the setting of this thesis, where only a limited number of confirmed piracy cases are available and the remaining customer population is unlabeled rather than verified normal.

However, several gaps remain. First, much existing work is evaluated mainly on general anomaly detection benchmarks, while production-scale business datasets introduce additional challenges such as noisy operational records, heterogeneous feature types, strong class imbalance, and partially contaminated unlabeled data. Second, reconstruction-based and weakly supervised anomaly detection methods have been less studied in the specific context of broadcast piracy detection from customer subscription behavior. Third, while FeaWAD and PReNet represent two useful but different learning principles, namely reconstruction-aware deviation learning and pairwise relation learning, it remains unclear how these ideas compare and whether they can be combined effectively for suspicious-customer ranking.

These gaps motivate the present thesis. This work studies broadcast piracy detection as a weakly supervised anomaly ranking problem on large-scale customer-level behavioral data. It evaluates a reconstruction-based autoencoder baseline, compares it with FeaWAD and PReNet, and further proposes a new model that combines reconstruction-aware feature encoding with pairwise ranking supervision. The goal is not only to improve detection performance, but also to understand how anomaly-ranking models can support practical anti-piracy investigation under limited inspection capacity.

# 3 Data

This project uses large-scale operational data from a subscription-based broadcast TV service. The raw data are stored across multiple tabular business records, including customer information, financial-account information, subscription events, migrations, subscriptions, and

package-channel mappings. The goal of the data processing pipeline is to transform these heterogeneous raw records into a customer-level dataset suitable for anomaly detection. In the final dataset, each row represents one customer, and each customer is associated either with a confirmed piracy label or with an unlabeled status.

## 3.1 Data sources

The original data are distributed across several CSV files, each describing a different part of the business process. Table 1 summarizes the main raw tables used in this thesis.

Table 1: Main raw tables used in the data pipeline.

| **Table** | **Main role in the pipeline** |
|---|---|
| `Customer.csv` | Customer master data at `CustomerNo` level, including customer-level attributes such as customer class. |
| `FinancialAccount.csv` | Financial-account records used to derive account-type indicators and the piracy label from `FinancialAccountStatus`. |
| `SubscriptionEvents.csv` | Subscription-event history at agreement-detail and subscription level, including time boundaries, product information, and event types. |
| `Migrations.csv` | Migration records between agreement details, used to rectify subscription histories across agreement changes. |
| `Subscriptions.csv` | Subscription and agreement records, including smartcard identifiers used to derive shared-card indicators. |
| `ChannelsByPackage.csv` | Package-to-channel mapping used to derive product-level sports exposure features. |

The final modeling unit is the customer. This choice follows the operational goal of the thesis: the model should rank customers by suspicion level so that investigators can prioritize accounts for further inspection. Therefore, event-level, agreement-level, account-level, and product-level records are all linked and aggregated into a single row per `CustomerNo`.

This transformation is non-trivial because the raw data are not directly suitable for model training. Subscription behavior is recorded across time-bounded event segments, migrations may change the agreement-detail identity of a subscription history, and product-level information must be enriched before it can be used as customer-level behavioral evidence. The data pipeline therefore plays an important role in converting operational records into structured behavioral features.

## 3.2 Data processing pipeline

Figure 1 gives an overview of the data processing and feature engineering pipeline. The pipeline begins with raw CSV files and ends with a customer-level dataset used for weakly supervised anomaly ranking.

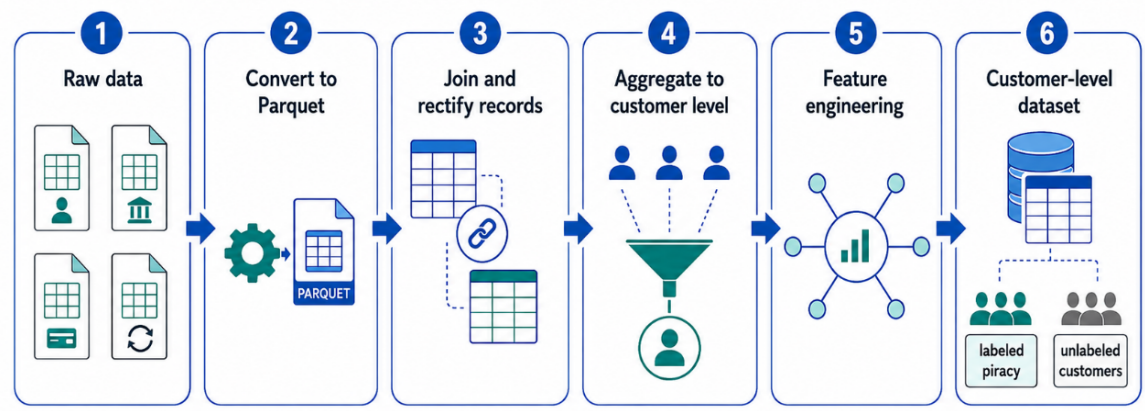


Figure 1: Overview of the data processing and feature engineering pipeline. Raw operational CSV files are first converted to Parquet for scalable processing. Subscription, migration, account, and product records are then joined and rectified. Agreement-level and segment-level histories are aggregated into one row per customer. Feature engineering produces customer-level behavioral features covering subscription activity, event-type distributions, migration behavior, concurrency, temporal changes, sports exposure, and customer/account characteristics. The resulting customer-level dataset contains labeled piracy customers and unlabeled customers.

First, the raw CSV files are converted into Parquet format [38]. This step improves storage efficiency and enables faster scanning during feature construction. Since the raw operational records are large, the processing pipeline uses lazy computation to avoid loading all intermediate data into memory at once.

Second, the subscription-event records are converted into time segments. Each event contains time-boundary fields, such as `StartDate` and `EndDate`. Segment duration is then computed using an end-exclusive convention, with a minimum duration of one day. This converts raw event records into a temporal representation of customer subscription behavior.

Third, migration records are used to rectify subscription histories. In the raw data, a customer may move from one agreement detail to another through a migration. If this is ignored, the same behavioral history may be split across different agreement identifiers, which would make customer-level aggregation incomplete. Therefore, migration records are applied in chronological order to reconnect related agreement-detail histories.

Fourth, the rectified agreement-level and segment-level records are aggregated to customer level. All subscription behavior belonging to the same `CustomerNo` is summarized into a single customer-level representation. This step changes the modeling unit from events, segments, and agreements to customers.

Fifth, feature engineering is applied to the aggregated customer histories. The goal is to convert raw operational behavior into numerical signals that can be used by anomaly detection models. The engineered features cover several behavioral dimensions: subscription and segment complexity, active versus inactive time, event-type distributions, migration frequency, overlapping active agreements, recent-versus-historical behavioral changes, sports-content exposure, and customer/account characteristics.

Finally, the engineered features are combined with the label information to form the final modeling dataset. Each row corresponds to one customer. This final customer-level dataset is used as input to the anomaly detection models evaluated in later sections.

## 3.3 Label definition

All customers with confirmed piracy labels are included in the dataset. In total, the data contain 8,703 labeled piracy customers. These labels are derived from the financial-account table: a customer is treated as a labeled piracy case when `FinancialAccountStatus` indicates `Piracy`.

All other customers are treated as unlabeled. Importantly, unlabeled does not mean verified normal. Some customers in the unlabeled population may still be suspicious but not yet confirmed as piracy cases. This is why the task is formulated as weakly supervised anomaly detection rather than standard binary classification.

## 3.4 Feature engineering

The feature engineering stage converts rectified subscription histories and account records into numerical customer-level features. The initial feature construction produced 158 engineered features. These features are designed to capture different aspects of customer behavior, including subscription complexity, event distributions, migration behavior, concurrency, temporal changes, sports exposure, and account characteristics.

### 3.4.1 Subscription and segment behavior

The first group of features describes the general structure of a customer's subscription history. These include the number of observed segments, the number of agreements, total active days, total inactive days, and the ratio of active time. These features capture how complex and persistent a customer's subscription activity is.

Additional features describe the frequency and length of subscription cycles. For example, the pipeline computes active-cycle intensity normalized by the observation span, as well as typical and long-tail active segment lengths. These features are useful because suspicious customers may show unusual subscription patterns, such as frequent short active periods, repeated reactivation, or irregular switching between products.

### 3.4.2 Event-type distribution

The subscription-event table contains different event types, such as new enable, disconnect, reconnect, and migration-related events. The distribution of these event types is summarized at customer level.

Two general diversity features are used: the number of unique event types and the entropy of the event-type distribution. A higher entropy indicates more diverse event behavior. In addition, the pipeline computes proportions of key event types, such as the proportion of new-enable events, disconnect events, reconnect events, and migration-injected events. These features help capture whether a customer has a stable subscription history or a more irregular event pattern.

### 3.4.3 Migration and concurrency features

Migration-related features describe how often and how broadly a customer's subscription history is affected by agreement migrations. These include the number of migrations, the number of unique source agreement details, the number of unique destination agreement details, and the proportion of segments affected by migration processing.

Concurrency features measure overlapping active subscriptions. The main concurrency feature is the maximum number of simultaneously active agreements for a customer. This is computed by scanning active intervals and counting overlaps over time. High concurrency may indicate complex account usage patterns and is therefore included as a customer-level behavioral signal.

### 3.4.4 Recent versus historical behavior

To capture temporal changes in behavior, the pipeline compares recent and older activity windows. The recent window is defined as the last 180 days, while earlier records form the historical comparison window.

For each customer, several behavior indicators are computed separately for the recent and old windows. These include switching intensity, active-cycle intensity, active-time ratio, event-type proportions, and concurrency. Difference features are then derived by subtracting old-window values from recent-window values. These delta features are intended to capture changes in behavior, such as a recent increase in switching, more frequent disconnects, or a change in active subscription patterns.

This temporal comparison is important because piracy-related behavior may not only be reflected in a customer's overall history, but also in recent changes. A customer whose behavior becomes suddenly more irregular may be more suspicious than a customer with a consistently stable history.

### 3.4.5 Sports exposure features

Sports-related products are important in this task because premium sports content may be more attractive for unauthorized access or redistribution. To capture this signal, product-level sports labels are derived from package-channel mappings. These labels include the number of sports channels, a sports score, an indicator for sports-heavy products, and an indicator for premium sports products.

These product-level labels are then aggregated to customer level using active subscription time. For example, the pipeline computes active-time-weighted average sports score, maximum sports score, and the proportion of active days spent on sports-heavy or premium sports products. Recent and historical versions of these features are also computed, together with their differences.

In addition, switching behavior between sports-related products is captured. Features include the number of switches between sports products, switches into sports products, switches out of sports products, and the average absolute change in sports score at product switches. These features are designed to represent both exposure to valuable content and dynamic changes in product usage.

### 3.4.6 Customer and account characteristics

The final feature group describes customer-level and account-level attributes. Customer characteristics include fields such as customer type and customer class. Account-level features include the number of distinct financial-account types associated with the customer.

A shared-card indicator is also included. This feature is derived by linking smart-card serial numbers across the global subscription data. If the same smart card is associated with multiple customers or agreements, the corresponding customer can be marked as having a shared-card signal. This feature is relevant because card sharing is one possible form of unauthorized access behavior.

## 3.5 Preprocessing for modeling

The feature engineering stage initially produces 158 customer-level candidate features. Before model training, the feature set is further cleaned and transformed. Identifier fields, join artifacts, potential leakage-related columns, and variables unsuitable for model input are removed. The remaining features are converted into numerical format where possible. Invalid values and infinite values are treated as missing values.

Missing values are imputed using feature-wise medians computed from the training data. For heavy-tailed non-negative variables, optional `log1p` transformations are applied to reduce the effect of extreme values. Categorical variables, such as customer type or customer class, are encoded using one-hot encoding. Finally, numerical features are normalized using robust scaling based on the training-set median and interquartile range.

After this preprocessing stage, the final model input contains 108 features per customer. This preprocessing strategy is designed to make the customer-level feature matrix suitable for neural anomaly detection models while reducing sensitivity to outliers, missing values, and skewed feature distributions.

### 3.6 Dataset characteristics

The resulting dataset has several characteristics that directly influence the modeling strategy. First, it is large-scale: the original operational data contain millions of raw records before aggregation. This means that the data preparation stage must be scalable and reproducible.

Second, the final dataset is high-dimensional and behavior-oriented. Each customer is represented by a set of engineered features that summarize subscription activity, event patterns, migration behavior, concurrency, temporal changes, sports exposure, and account characteristics. Many of these features are related, so some degree of redundancy is expected.

Third, the dataset is highly imbalanced. Only 8,703 customers are confirmed piracy cases, while the remaining population is unlabeled. This makes standard supervised classification less suitable, especially because the negative class is not fully reliable.

Finally, the unlabeled set may be contaminated by undiscovered piracy cases. Therefore, the task is not a clean normal-versus-anomaly classification problem. Instead, it is more naturally treated as a weakly supervised anomaly ranking problem, where the objective is to assign higher suspicion scores to customers whose behavior resembles confirmed piracy cases or deviates strongly from typical customer behavior.

## 4 Methods

This section describes how the anomaly detection methods introduced in Section 2 are instantiated for the broadcast piracy detection task. The focus here is on explaining how each model is implemented, trained, and used to rank customer-level feature vectors. The baseline autoencoder, FeaWAD, and PReNet are implemented as comparative models, while the unified FeaWAD–PReNet model is proposed as the main modeling contribution of this thesis.

### 4.1 Problem formulation

After preprocessing, each customer is represented by a feature vector

$$x_i \in \mathbb{R}^d,$$

where $d = 108$ is the number of model-ready input features. The dataset contains a small set of labeled piracy customers and a much larger set of unlabeled customers. Let $\mathcal{P}$ denote

the labeled piracy set and $\mathcal{U}$ denote the unlabeled set. Since the unlabeled set may contain undiscovered piracy cases, the task is not formulated as standard binary classification.

Instead, following the anomaly-ranking view introduced in Section 2, the objective is to learn a scoring function

$$s : \mathbb{R}^d \to \mathbb{R},$$

where a larger value of $s(x_i)$ indicates a higher suspicion level. Customers are then ranked in descending order of $s(x_i)$, so that confirmed piracy customers are expected to appear near the top of the investigation list.

## 4.2 Baseline autoencoder

The baseline model follows the reconstruction-based anomaly detection approach reviewed in Section 2.2. It is used as an unsupervised reference model to examine whether reconstruction error alone can provide useful suspicious-customer rankings.

Given a preprocessed customer feature vector $x \in \mathbb{R}^d$, the autoencoder maps the input to a latent representation and reconstructs it following the general encoder-decoder formulation in Equation (1) and (2). In the implemented model, both the encoder and decoder are multilayer perceptrons. The encoder compresses the customer-level feature vector into a lower-dimensional latent representation, and the decoder reconstructs the original feature vector from this representation.

Figure 2 illustrates the architecture of the baseline autoencoder used in this thesis. The input consists of the 108 model-ready customer-level features described in Section 3. The output is a reconstructed feature vector $\hat{x}$ with the same dimension as the input.

The model is trained by minimizing the mean squared reconstruction error:

$$\mathcal{L}_{\mathrm{AE}} = \frac{1}{d}\|x - \hat{x}\|_2^2. \tag{15}$$

At inference time, the anomaly score for a customer is defined as the per-feature reconstruction error:

$$s_{\mathrm{AE}}(x) = \frac{1}{d}\sum_{j=1}^{d}(x_j - \hat{x}_j)^2. \tag{16}$$

Customers with larger reconstruction errors are assigned higher suspicion scores. Since the baseline model does not use piracy labels during training, it tests how much piracy-related signal can be captured purely from deviations in customer behavior. This provides a reference point for evaluating whether weakly supervised methods can make better use of the available labeled piracy cases.

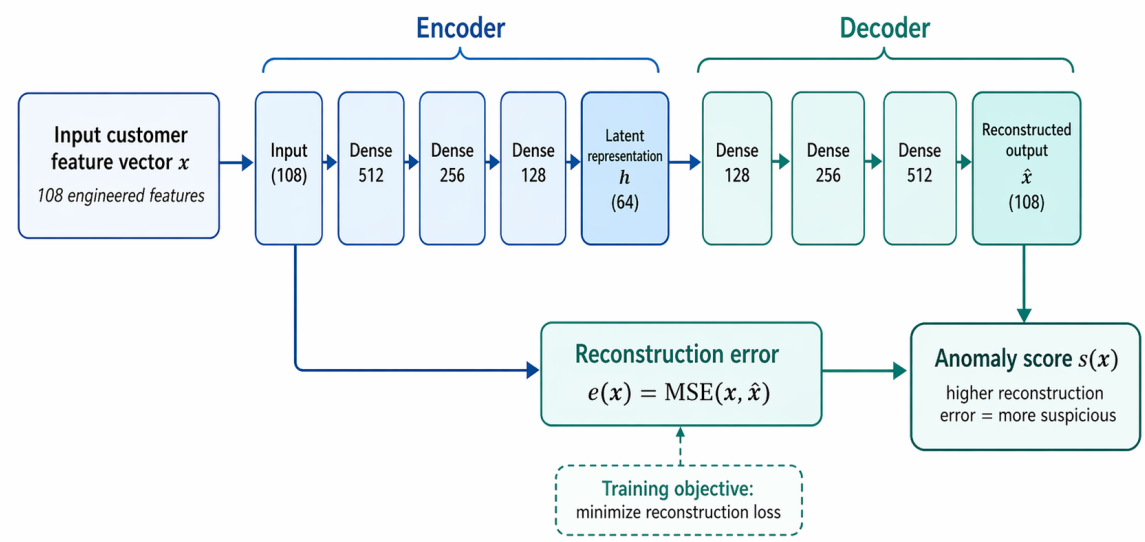


Figure 2: Architecture of the baseline autoencoder. The model takes the 108-dimensional customer-level feature vector as input, compresses it into a latent representation through the encoder, and reconstructs it through the decoder. The reconstruction error between the original input $x$ and the reconstruction $\hat{x}$ is used as the customer-level anomaly score.

## 4.3 FeaWAD

The FeaWAD-style model follows the deviation-based weakly supervised anomaly detection approach reviewed in Section 2.3.1. In this thesis, FeaWAD is adapted to customer-level piracy ranking by using confirmed piracy customers as labeled anomalies and the remaining customers as unlabeled reference samples. Compared with the baseline autoencoder, FeaWAD does not rely only on reconstruction error. Instead, it learns an anomaly score from a reconstruction-aware feature representation.

### 4.3.1 Network design

Given a preprocessed customer feature vector $x \in \mathbb{R}^d$, where $d = 108$, the model first passes $x$ through an autoencoder backbone. The encoder produces a latent representation $h$, and the decoder reconstructs the input as $\hat{x}$. Following the FeaWAD representation introduced in Equation (8), the model constructs a reconstruction-aware feature vector

$$z = [h, r, e],$$

where $r = x - \hat{x}$ is the feature-wise reconstruction residual and $e = \|x - \hat{x}\|_2$ is the reconstruction magnitude.

The vector $z$ is then passed to a scoring network:

$$s_{\mathrm{FeaWAD}}(x) = f_{\mathrm{score}}(z), \tag{17}$$

where $s_{\text{FeaWAD}}(x)$ is the customer-level anomaly score. This score is used to rank customers in descending order of suspicion.

Figure 3 illustrates the implemented FeaWAD architecture. The autoencoder branch provides the latent representation and reconstruction residuals, while the scoring branch learns how these signals should be combined to produce a piracy-risk score.

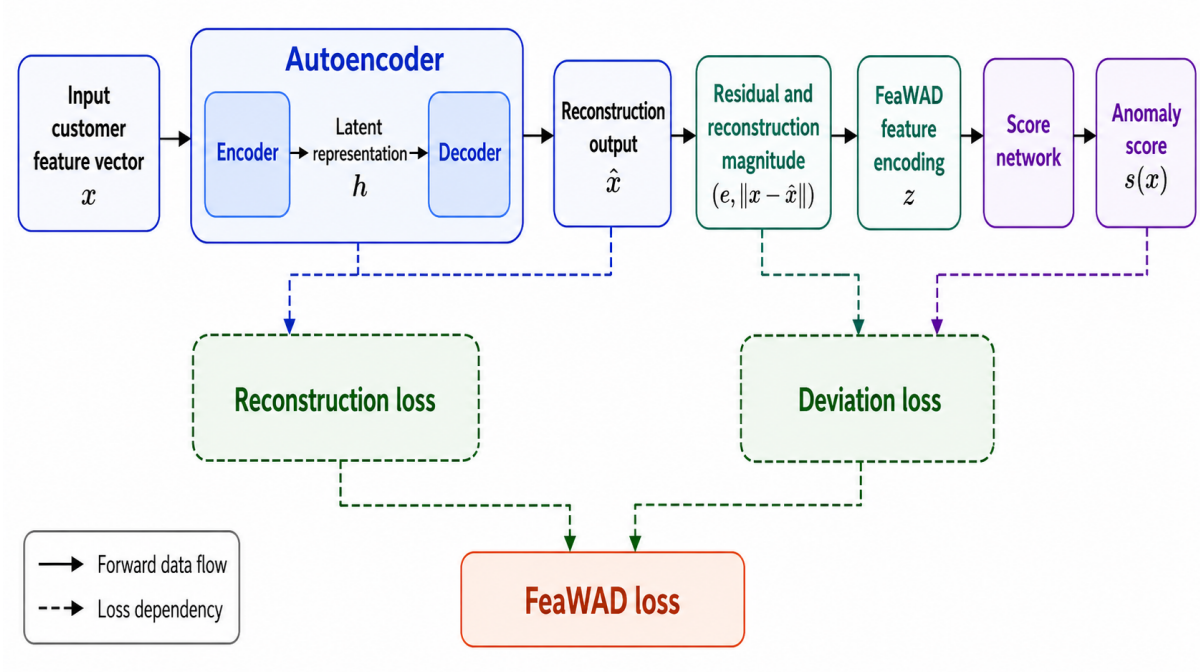


Figure 3: Architecture and training objective of the implemented FeaWAD-style model. The customer feature vector $x$ is first processed by an autoencoder to obtain the latent representation $h$ and reconstruction $\hat{x}$. The residual vector $r = x - \hat{x}$ and reconstruction magnitude $e = \|x - \hat{x}\|_2$ are combined with $h$ to form the reconstruction-aware representation $z = [h, r, e]$, which is then mapped by a scoring network to the anomaly score $s_{\text{FeaWAD}}(x)$. During training, the autoencoder branch is optimized with reconstruction loss, while the scoring branch is optimized with deviation loss. The final FeaWAD objective combines these two terms. Solid arrows indicate forward data flow, and dashed arrows indicate loss dependencies.

### 4.3.2 Loss function

The FeaWAD-style model is trained with two complementary objectives: reconstruction learning and deviation-based score supervision. The reconstruction term preserves the autoencoder structure and encourages the model to capture common customer behavior patterns:

$$\mathcal{L}_{\text{rec}} = \frac{1}{d}\|x - \hat{x}\|_2^2. \tag{18}$$

The deviation term follows the DevNet-style loss introduced in Section 2.3.1. In this thesis, the reference score distribution is estimated from the scores of unlabeled customers during

training. At training step $t$, let $U_t$ denote the unlabeled samples in the current batch. The batch mean and standard deviation of the unlabeled scores are computed as

$$\hat{\mu}_t = \frac{1}{|U_t|} \sum_{x_i \in U_t} s_{\text{FeaWAD}}(x_i), \tag{19}$$

and

$$\hat{\sigma}_t = \sqrt{\frac{1}{|U_t|} \sum_{x_i \in U_t} \left(s_{\text{FeaWAD}}(x_i) - \hat{\mu}_t\right)^2}. \tag{20}$$

To make the reference distribution more stable across batches, the running mean and standard deviation are updated using an exponential moving average:

$$\mu_t = \beta \mu_{t-1} + (1-\beta)\hat{\mu}_t, \tag{21}$$

$$\sigma_t = \beta \sigma_{t-1} + (1-\beta)\max(\hat{\sigma}_t, \sigma_{\min}), \tag{22}$$

where $\beta$ is the smoothing coefficient and $\sigma_{\min}$ is used to avoid numerical instability when the score variance is very small.

Each sample score is normalized relative to this running reference distribution:

$$d_i = \frac{s_{\text{FeaWAD}}(x_i) - \mu_t}{\max(\sigma_t, \sigma_{\min})}. \tag{23}$$

This normalized quantity $d_i$ measures how far the sample score lies from the current reference score distribution. A value near zero means that the sample score is close to the typical unlabeled score level, while a larger positive value indicates that the sample is more strongly separated from the reference population.

Given label $y_i \in \{0, 1\}$, where $y_i = 1$ indicates a labeled piracy customer and $y_i = 0$ indicates an unlabeled customer, the deviation loss is defined as

$$\mathcal{L}_{\text{dev}} = \frac{1}{N} \sum_{i=1}^{N} \left[(1-y_i)|d_i| + y_i \max(0, a - d_i)\right], \tag{24}$$

where $a$ is the deviation margin. This loss encourages unlabeled customers to remain close to the reference score distribution, while pushing labeled piracy customers above the reference distribution by at least the margin $a$. The margin parameter $a$ is important because it enforces *separation*, not merely a slight score increase. Without such a margin, labeled piracy samples might only be pushed to be marginally larger than the reference level, which may be insufficient for reliable ranking under heavy class imbalance. By requiring $d_i \geq a$, the model is encouraged to assign clearly higher scores to labeled piracy samples. In this sense, $a$ controls how strongly the model separates labeled piracy behavior from the reference unlabeled population.

The final FeaWAD objective is

$$\mathcal{L}_{\text{FeaWAD}} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{dev}}. \tag{25}$$

The full FeaWAD objective combines reconstruction-based representation learning with supervised score separation. The reconstruction term helps preserve the dominant behavioral structure of the data, while the deviation term explicitly encourages labeled piracy customers to occupy the high-score region.

### 4.3.3 Inference

At inference time, only the scoring branch is needed. For each customer $x$, the model computes $s_{\text{FeaWAD}}(x)$, and customers are ranked in descending order of this score. A larger score indicates that the customer is more separated from the unlabeled reference distribution and is therefore considered more suspicious.

## 4.4 PReNet

The PReNet-style model follows the pairwise relation learning approach reviewed in Section 2.3.2. In this thesis, it is adapted to customer-level piracy detection by constructing pairs between labeled piracy customers and unlabeled customers. Unlike the baseline autoencoder, which relies on reconstruction error, PReNet directly learns relative ordering: labeled piracy customers should receive higher anomaly scores than unlabeled customers.

### 4.4.1 Network design

Given a preprocessed customer feature vector $x \in \mathbb{R}^d$, where $d = 108$, the model first maps the input into a latent representation:

$$h = f_{\text{enc}}(x). \tag{26}$$

Based on this shared representation, the implemented PReNet model has two output branches. The first branch is a single-sample scoring branch:

$$s_{\text{PReNet}}(x) = f_{\text{score}}(h), \tag{27}$$

where $s_{\text{PReNet}}(x)$ is the customer-level anomaly score used for final ranking.

The second branch is a pairwise relation branch. For a pair of customers $(x_i, x_j)$, let

$$h_i = f_{\text{enc}}(x_i), \qquad h_j = f_{\text{enc}}(x_j).$$

Following the pairwise representation introduced in Equation (11), the pair feature vector is constructed as

$$\phi(h_i, h_j) = [h_i,\ h_j,\ |h_i - h_j|,\ h_i \odot h_j], \tag{28}$$

where $\odot$ denotes element-wise multiplication. The relation branch then outputs a relation logit:

$$r_{ij} = f_{\text{rel}}(\phi(h_i, h_j)). \tag{29}$$

The single-sample scoring branch determines the final suspicious-customer ranking, while the pairwise relation branch provides additional supervision during training. The overall architecture of the implemented PReNet-style model is shown in Figure 4.

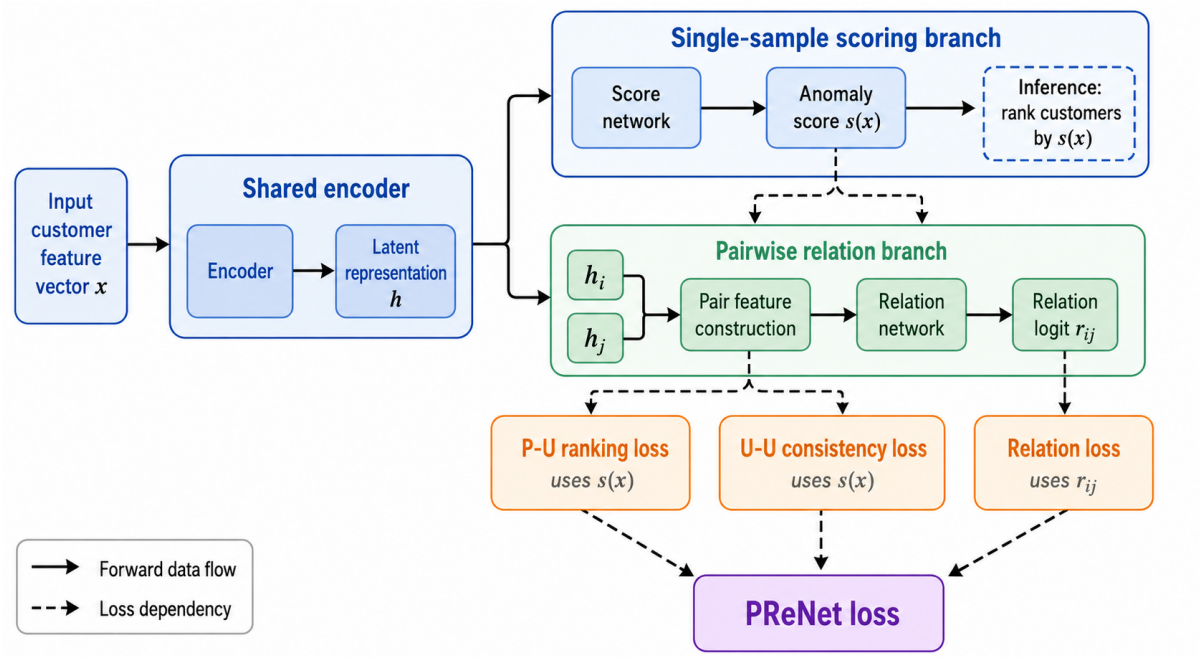


Figure 4: Architecture and training objective of the implemented PReNet-style model. A shared encoder maps each customer feature vector to a latent representation $h$. The single-sample scoring branch outputs the anomaly score $s_{\mathrm{PReNet}}(x)$, which is used to rank customers at inference time. The pairwise relation branch constructs pair features from two latent representations and outputs the relation logit $r_{ij}$. During training, $s_{\mathrm{PReNet}}(x)$ is used by the piracy–unlabeled ranking loss and the unlabeled–unlabeled consistency loss, while $r_{ij}$ is used by the relation loss. The three loss terms are combined into the final PReNet objective. Solid arrows indicate forward data flow, and dashed arrows indicate loss dependencies.

### 4.4.2 Loss function

The implemented PReNet-style model is trained using three loss components:

$$\mathcal{L}_{\mathrm{PReNet}} = \mathcal{L}_{\mathrm{rank}} + \lambda_{\mathrm{rel}}\mathcal{L}_{\mathrm{rel}} + \lambda_{\mathrm{uu}}\mathcal{L}_{\mathrm{uu}}, \tag{30}$$

where $\mathcal{L}_{\mathrm{rank}}$ is the main pairwise ranking loss, $\mathcal{L}_{\mathrm{rel}}$ is the relation-classification loss, and $\mathcal{L}_{\mathrm{uu}}$ is the unlabeled–unlabeled score consistency loss. The coefficients $\lambda_{\mathrm{rel}}$ and $\lambda_{\mathrm{uu}}$ control the contribution of the auxiliary terms.

**Piracy–unlabeled ranking loss.** Let $x_p$ denote a labeled piracy customer and $x_u$ denote an unlabeled customer. Their anomaly scores are

$$s_p = s_{\mathrm{PReNet}}(x_p), \qquad s_u = s_{\mathrm{PReNet}}(x_u).$$

The ranking loss is defined as

$$\mathcal{L}_{\mathrm{rank}} = \frac{1}{N_{pu}} \sum_{(p,u)} \max\big(0, m - (s_p - s_u)\big), \tag{31}$$

where $m > 0$ is the ranking margin and $N_{pu}$ is the number of piracy–unlabeled pairs. This loss encourages labeled piracy customers to receive higher scores than unlabeled customers by at least margin $m$.

**Relation loss.** The relation branch is trained to distinguish different pair types. In this implementation, piracy–unlabeled pairs and unlabeled–unlabeled pairs are used. For piracy–unlabeled pairs, the target is set to $1$:

$$\mathcal{L}_{\mathrm{rel}}^{PU} = \frac{1}{N_{pu}} \sum_{(p,u)} \mathrm{BCEWithLogits}(r_{pu}, 1). \tag{32}$$

For unlabeled–unlabeled pairs, the target is set to $0$:

$$\mathcal{L}_{\mathrm{rel}}^{UU} = \frac{1}{N_{uu}} \sum_{(u_1,u_2)} \mathrm{BCEWithLogits}(r_{u_1u_2}, 0). \tag{33}$$

The full relation loss is

$$\mathcal{L}_{\mathrm{rel}} = \mathcal{L}_{\mathrm{rel}}^{PU} + \mathcal{L}_{\mathrm{rel}}^{UU}. \tag{34}$$

This loss encourages the relation branch to assign higher logits to piracy–unlabeled pairs and lower logits to unlabeled–unlabeled pairs. In this way, the model learns not only individual customer scores, but also pairwise patterns that distinguish labeled piracy behavior from background customer behavior.

**Unlabeled–unlabeled score consistency loss.** For unlabeled–unlabeled pairs, an additional consistency loss is used:

$$\mathcal{L}_{\mathrm{uu}} = \frac{1}{N_{uu}} \sum_{(u_1,u_2)} \left( s_{\mathrm{PReNet}}(u_1) - s_{\mathrm{PReNet}}(u_2) \right)^2 . \tag{35}$$

This term encourages unlabeled customers to occupy a relatively compact score region. Since the unlabeled population is expected to be dominated by non-piracy customers, this regularization helps stabilize the scoring function and reduces unnecessary score dispersion among unlabeled samples.

### 4.4.3 Interpretation

The implemented PReNet-style model combines individual scoring and pairwise supervision. The ranking loss directly optimizes the ordering between labeled piracy customers and unlabeled customers, which is closely aligned with the goal of suspicious-customer ranking. The relation branch provides an additional training signal by encouraging the encoder to learn pairwise differences between piracy–unlabeled and unlabeled–unlabeled pairs. The U-U consistency loss further stabilizes the score distribution of the unlabeled population.

Compared with FeaWAD, PReNet does not rely on reconstruction-aware residual features. Instead, its main strength comes from pairwise ordering and relation learning. This makes it a useful comparison model for evaluating whether direct ranking supervision is more effective than deviation-based score learning in this piracy detection task.

### 4.4.4 Inference

At inference time, only the single-sample scoring branch is used. For each customer $x$, the model computes $s_{\mathrm{PReNet}}(x)$, and customers are ranked in descending order of this score. The pairwise relation branch is used only during training and is not required for scoring customers during evaluation.

## 4.5 Unified FeaWAD–PReNet model

The unified FeaWAD–PReNet model is proposed to combine the two weakly supervised learning principles used by FeaWAD and PReNet. FeaWAD provides reconstruction-aware feature encoding through the representation $z = [h, r, e]$, while PReNet provides pairwise ranking supervision between labeled piracy customers and unlabeled customers. The goal of the unified model is to learn a shared representation that benefits from both individual reconstruction-based deviation signals and pairwise ordering signals.

### 4.5.1 Network design

Given a preprocessed customer feature vector $x \in \mathbb{R}^d$, where $d = 108$, the model first passes $x$ through a shared autoencoder backbone. The encoder maps the input to a latent representation

$$h = f_{\text{enc}}(x), \tag{36}$$

and the decoder reconstructs the input as

$$\hat{x} = f_{\text{dec}}(h). \tag{37}$$

Following the FeaWAD-style construction, the reconstruction residual and reconstruction magnitude are computed as

$$r = x - \hat{x}, \qquad e = \|x - \hat{x}\|_2. \tag{38}$$

These components are concatenated into a reconstruction-aware representation:

$$z = [h,\ r,\ e]. \tag{39}$$

This representation is shared by two branches. The first branch is a single-sample scoring branch:

$$s_{\text{Hybrid}}(x) = f_{\text{score}}(z), \tag{40}$$

where $s_{\text{Hybrid}}(x)$ is the customer-level anomaly score used for final ranking.

The second branch is a pairwise relation branch. For two customers $x_i$ and $x_j$, the model first computes their reconstruction-aware representations $z_i$ and $z_j$. A pairwise feature vector is then constructed as

$$\phi(z_i, z_j) = [z_i,\ z_j,\ |z_i - z_j|,\ z_i \odot z_j], \tag{41}$$

where $\odot$ denotes element-wise multiplication. The relation branch maps this pair representation to a relation logit:

$$r_{ij} = f_{\text{rel}}(\phi(z_i, z_j)). \tag{42}$$

The key difference from PReNet is that the pairwise branch operates on the reconstruction-aware representation $z$, rather than only on the latent representation $h$. Therefore, the relation branch can use not only latent behavioral information, but also feature-wise reconstruction residuals and overall reconstruction magnitude.

The overall architecture of the proposed unified FeaWAD–PReNet model is shown in Figure 5.

### 4.5.2 Loss function

The unified model is trained with a weighted combination of the FeaWAD-style loss and the PReNet-style loss:

$$\mathcal{L}_{\text{Hybrid}} = (1 - \lambda_{\text{rank}})\mathcal{L}_{\text{FeaWAD}} + \lambda_{\text{rank}}\mathcal{L}_{\text{PReNet}}, \tag{43}$$

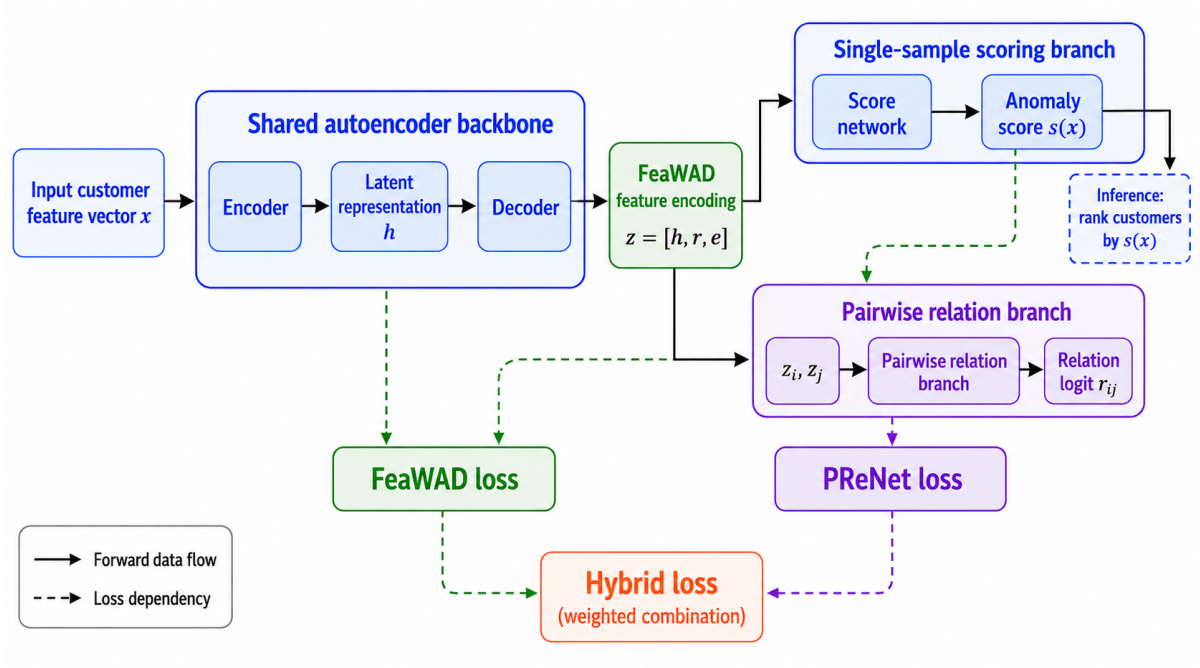


Figure 5: Architecture and training objective of the unified FeaWAD–PReNet model. The customer feature vector $x$ is first processed by an autoencoder to obtain the latent representation $h$ and reconstruction $\hat{x}$. The residual vector $r = x - \hat{x}$ and reconstruction magnitude $e = \|x - \hat{x}\|_2$ are combined with $h$ to form the reconstruction-aware representation $z = [h, r, e]$. This shared representation is used by both the single-sample scoring branch and the pairwise relation branch. The model is trained with a hybrid objective that combines FeaWAD-style reconstruction and deviation losses with PReNet-style ranking, relation, and consistency losses. Solid arrows indicate forward data flow, and dashed arrows indicate loss dependencies.

where $\lambda_{\text{rank}} \in [0, 1]$ controls the balance between reconstruction-aware deviation learning and pairwise ranking learning.

The FeaWAD component consists of reconstruction loss and deviation loss:

$$\mathcal{L}_{\text{FeaWAD}} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{dev}}, \tag{44}$$

The PReNet component follows the same structure as in Section 4.4:

$$\mathcal{L}_{\text{PReNet}} = \mathcal{L}_{\text{rank}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{uu}}\mathcal{L}_{\text{uu}}. \tag{45}$$

Combining these components, the full objective can be written as

$$\mathcal{L}_{\text{Hybrid}} = (1 - \lambda_{\text{rank}}) \left(\lambda_{\text{recon}}\mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{dev}}\right) + \lambda_{\text{rank}} \left(\mathcal{L}_{\text{rank}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{uu}}\mathcal{L}_{\text{uu}}\right). \tag{46}$$

#### 4.5.3 Inference

At inference time, only the single-sample scoring branch is used. For each customer $x$, the model computes $s_{\text{Hybrid}}(x)$, and customers are ranked in descending order of this score. The autoencoder and scoring branch are required to compute $z$ and $s_{\text{Hybrid}}(x)$, while the pairwise relation branch is used only during training.

# 5 Experimental Setup

## 5.1 Dataset split

The experiments use all labeled piracy customers together with a 30% deterministic sample of the unlabeled population. The labeled and unlabeled subsets are split separately into 70% training, 10% validation, and 20% test, after which the corresponding subsets are merged and shuffled. In the main setup, this produces 3,347,624 / 478,232 / 956,464 unlabeled customers and 6,092 / 870 / 1,741 labeled piracy customers for train / validation / test, respectively.

Table 2: Dataset split statistics.

| **Class** | **Train** | **Validation** | **Test** |
|---|---|---|---|
| Unlabeled | 3,347,624 | 478,232 | 956,464 |
| Labeled piracy | 6,092 | 870 | 1,741 |

## 5.2 Evaluation protocol

The task is evaluated as a ranking problem rather than a standard classification problem. For each model, all customers in the evaluation pool are assigned an anomaly score and ranked

in descending order, so that customers with larger scores are considered more suspicious. The evaluation pool is a mixed set consisting of labeled piracy customers and unlabeled customers:

$$\mathcal{D}_{\text{eval}} = P_{\text{eval}} \cup U_{\text{eval}},$$

where $P_{\text{eval}}$ denotes the set of labeled piracy customers in the evaluation split and $U_{\text{eval}}$ denotes the set of unlabeled customers in the same split.

For a given inspection budget $K$, let Top-$K$ denote the set of the top $K$ ranked customers, and let

$$n_K = |\text{Top-}K \cap P_{\text{eval}}|$$

be the number of labeled piracy customers found in the top-$K$ list. Based on this quantity, three ranking metrics are used.

**Recall@$K$.** Recall@$K$ measures the proportion of labeled piracy customers retrieved within the top-$K$ ranked accounts:

$$\text{Recall@}K = \frac{n_K}{|P_{\text{eval}}|}.$$

This metric reflects how much of the known piracy population is covered when investigators inspect only the first $K$ accounts. In other words, it measures the fraction of piracy cases recovered under a limited investigation budget.

**HitRate@$K$.** HitRate@$K$ measures the piracy density within the top-$K$ list:

$$\text{HitRate@}K = \frac{n_K}{K}.$$

This metric indicates the proportion of investigated accounts that are actually labeled piracy customers. Compared with Recall@$K$, it focuses on the precision of the top-ranked portion rather than on overall piracy coverage.

**Lift@$K$.** Because piracy is extremely rare in the full customer population, Recall@$K$ and HitRate@$K$ can appear numerically small even when a model is useful. To address this issue, Lift@$K$ normalizes the top-$K$ hit rate by the overall piracy prevalence in the evaluation pool:

$$\text{Lift@}K = \frac{\text{HitRate@}K}{\pi_{\text{eval}}}, \qquad \pi_{\text{eval}} = \frac{|P_{\text{eval}}|}{|P_{\text{eval}}| + |U_{\text{eval}}|}.$$

Here, $\pi_{\text{eval}}$ is the overall piracy ratio in the mixed evaluation set.

Lift@$K$ answers the practical question: *if investigators only inspect the top-$K$ accounts, how much more piracy will they observe compared with random selection?* This makes Lift@$K$ especially relevant for the operational use case, where investigation capacity is limited and the goal is to prioritize the most promising accounts.

**Example.** Assuming the evaluation pool contains

$$|P_{\text{eval}}| = 1741, \qquad |U_{\text{eval}}| = 956{,}464.$$

The overall piracy ratio is therefore

$$\pi_{\text{eval}} = \frac{1741}{1741 + 956464} \approx 0.00182.$$

Suppose a model returns a ranked list and finds $n_K = 90$ labeled piracy customers in the top $K = 10{,}000$ accounts. Then

$$\text{HitRate@10,000} = \frac{90}{10{,}000} = 0.009,$$

and

$$\text{Lift@10,000} = \frac{0.009}{0.00182} \approx 4.95.$$

This means that the top-10,000 list is about 4.95 times more piracy-dense than random selection from the full evaluation pool.

**Interpretation of Lift@$K$.** The interpretation of Lift@$K$ is straightforward:

Table 3: Interpretation of Lift@$K$ Metrics

| Metric Value | Relationship | Interpretation |
|---|---|---|
| Lift@$K > 1$ | $>$ Random | Piracy is enriched in the top-$K$ list; the model identifies targets better than chance. |
| Lift@$K = 10$ | $= 10 \times$ Random | The top-$K$ accounts contain ten times more piracy than expected under random sampling. |
| Lift@$K = 1$ | $=$ Random | The model is no better than random ranking (neutral predictive power). |
| Lift@$K < 1$ | $<$ Random | The ranking is worse than random selection (the model is actively avoiding targets). |

## 5.3 Batch strategy and warmup

The weakly supervised models use different batch construction strategies. FeaWAD is trained on mixed instance batches containing both piracy and unlabeled customers, with piracy oversampled to ensure that labeled positives contribute to each update. PReNet is trained on pair batches, where the dominant supervision comes from piracy–unlabeled pairs and a smaller auxiliary portion comes from unlabeled–unlabeled pairs. The unified hybrid model combines both strategies within each training iteration: one mixed instance batch is used for the FeaWAD component, and one pair batch is used for the PReNet component.

Warmup is used only for models with an autoencoder reconstruction branch. In FeaWAD, the first five epochs optimize reconstruction loss only, after which the deviation loss is activated. The hybrid model follows the same idea: during the warmup stage, training uses only the reconstruction branch before switching to the blended objective.

## 5.4 Baseline autoencoder setup

The baseline tabular autoencoder was tuned using a validation-based hyperparameter search. The training objective was mean squared reconstruction error. Optimization used AdamW with learning rate $10^{-3}$, weight decay $10^{-5}$, and gradient clipping with maximum norm 5.0. Training was run for up to 200 epochs, with early stopping (patience 15) and learning-rate reduction on plateau (patience 7, decay factor 0.5).

## 5.5 Weakly supervised model settings

The main training settings are summarized in Table 4. The notation follows the model definitions in Section 4: $a$, $\beta$, and $\sigma_{\min}$ are the FeaWAD deviation-loss parameters; $m$, $\lambda_{\text{rel}}$, and $\lambda_{\text{uu}}$ are the PReNet loss parameters; and $\lambda_{\text{rank}}$ controls the hybrid objective.

Table 4: Model-specific training settings used in the main experiments.

| Model | Training setup | Objective-specific settings |
|---|---|---|
| **FeaWAD** | Representation dimension: 32; batch size: 4096; epochs: 30; learning rate: $10^{-3}$; weight decay: 0. Each batch contains 10% labeled piracy customers and 90% unlabeled customers. | Uses $\mathcal{L}_{\text{FeaWAD}} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{dev}}$. The reconstruction term is computed on unlabeled samples only. The deviation loss uses $a = 5.0$, $\beta = 0.99$, and $\sigma_{\min} = 10^{-2}$. A 5-epoch autoencoder warmup is applied before activating the full objective. |
| **PReNet** | Representation dimension: 64; pair batch size: 2048; epochs: 5; learning rate: $10^{-3}$; weight decay: $10^{-5}$. Training pairs contain 80% piracy–unlabeled pairs and 20% unlabeled–unlabeled pairs. | Uses $\mathcal{L}_{\text{PReNet}} = \mathcal{L}_{\text{rank}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{uu}}\mathcal{L}_{\text{uu}}$. The ranking margin is $m = 1.0$. The auxiliary weights are $\lambda_{\text{rel}} = 0.25$ and $\lambda_{\text{uu}} = 0.5$. |
| **Hybrid** | Representation dimension: 32; batch size: 2048; epochs: 10; learning rate: $10^{-3}$; weight decay: $10^{-5}$. A 5-epoch warmup is applied before joint training. | Uses a weighted combination of FeaWAD and PReNet objectives. The FeaWAD branch uses $a = 5.0$, $\beta = 0.99$, and $\sigma_{\min} = 10^{-2}$. The PReNet branch uses an 80% / 20% piracy–unlabeled / unlabeled–unlabeled pair mix, $m = 1.0$, $\lambda_{\text{rel}} = 0.5$, and $\lambda_{\text{uu}} = 0.1$. The objective balance is controlled by $\lambda_{\text{rank}} = 0.42$. |

*Note.* Piracy–unlabeled pairs use one labeled piracy customer and one unlabeled customer. Unlabeled–unlabeled pairs use two unlabeled customers.

# 6 Results

This section reports the test-set results of the four models: the baseline autoencoder, FeaWAD, PReNet, and the unified FeaWAD–PReNet hybrid model. All results are obtained on the same mixed evaluation pool, consisting of 1,741 labeled piracy customers and 956,464 unlabeled customers.

## 6.1 Baseline autoencoder experiments

The baseline autoencoder was first used to examine how far a purely reconstruction-based model can detect piracy-related behavior from customer-level subscription features.

### 6.1.1 Hyperparameter optimization

Since the performance of the autoencoder depends on architectural choices, hyperparameter tuning was conducted first on the validation set. Seven candidate architectures listed in Table 5 were evaluated by varying hidden dimensions, latent dimension, activation, and dropout.

Table 5: Hyperparameter search space for the baseline autoencoder.

| ID | Hidden dims | Latent dim | Activation | Dropout |
|---|---|---|---|---|
| 1 | [512, 256, 128, 32] | 64 | ReLU | 0.01 |
| 2 | [512, 256, 128] | 64 | ReLU | 0.01 |
| 3 | [512, 256, 64] | 32 | ReLU | 0.01 |
| 4 | [256, 128, 64] | 32 | ReLU | 0.01 |
| 5 | [256, 128] | 32 | GELU | 0.02 |
| 6 | [512, 256, 128, 64] | 32 | LeakyReLU | 0.01 |
| 7 | [256, 128, 64] | 16 | ReLU | 0.00 |

Figure 6 shows Recall@$K$ for the seven candidate autoencoder configurations, and Figure 7 shows the corresponding Lift@$K$ curves. The results indicate that the architecture mainly affects performance at relatively small and medium inspection budgets. The differences between configurations are most visible in Figure 7 from around top-500 to top-10,000, while the curves become more similar at larger $K$. Based on this comparison, the final baseline used the configuration selected by validation performance, namely hidden dimensions $[512, 256, 128]$, latent dimension $64$, ReLU activation, and dropout $0.01$.

### 6.1.2 Feature sweep

A feature-sweep experiment was conducted to investigate how many input features are needed to obtain satisfactory performance. For this purpose, the engineered features were ranked, and progressively smaller top-$M$ feature subsets were used to retrain the baseline autoencoder,

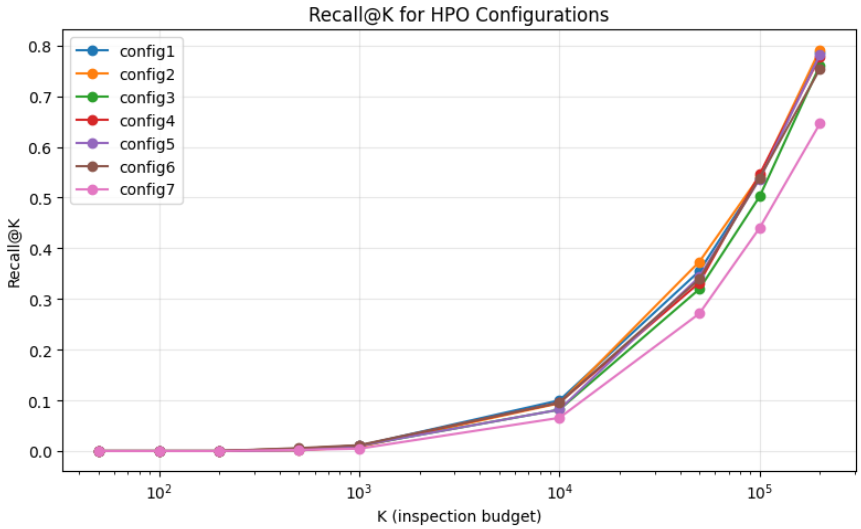


Figure 6: Recall@$K$ for the baseline autoencoder under different hyperparameter configurations.

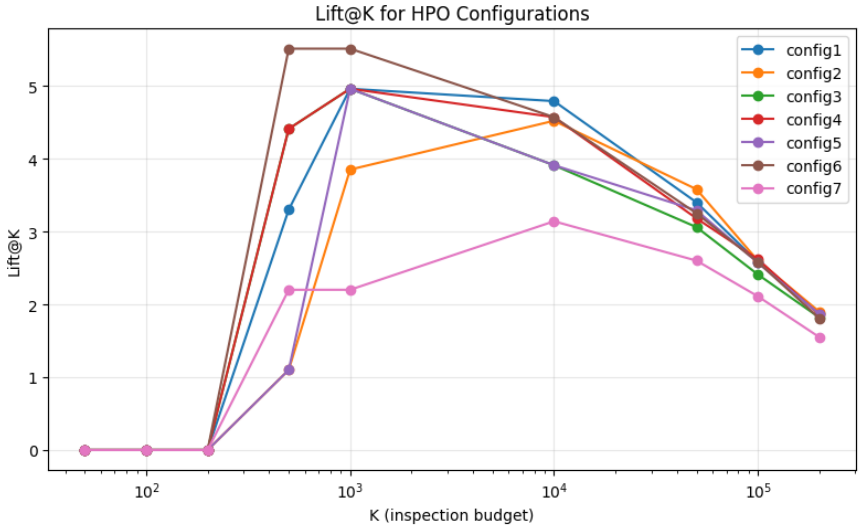


Figure 7: Lift@$K$ for the baseline autoencoder under different hyperparameter configurations.

where $M$ denotes the number of selected input features. Figure 8 and Figure 9 compare Recall@$K$ and Lift@$K$ across different feature subset sizes. Here, $K$ denotes the inspection budget in the ranked customer list, while $M$ denotes the number of retained features.

The results show that reducing the number of features does not necessarily lead to a large performance drop. Several reduced feature sets achieve performance comparable to the full 108-feature configuration, especially at medium and large inspection budgets. This suggests that much of the useful piracy-related signal is concentrated in a smaller subset of features, and that the baseline autoencoder is relatively robust to redundant inputs.

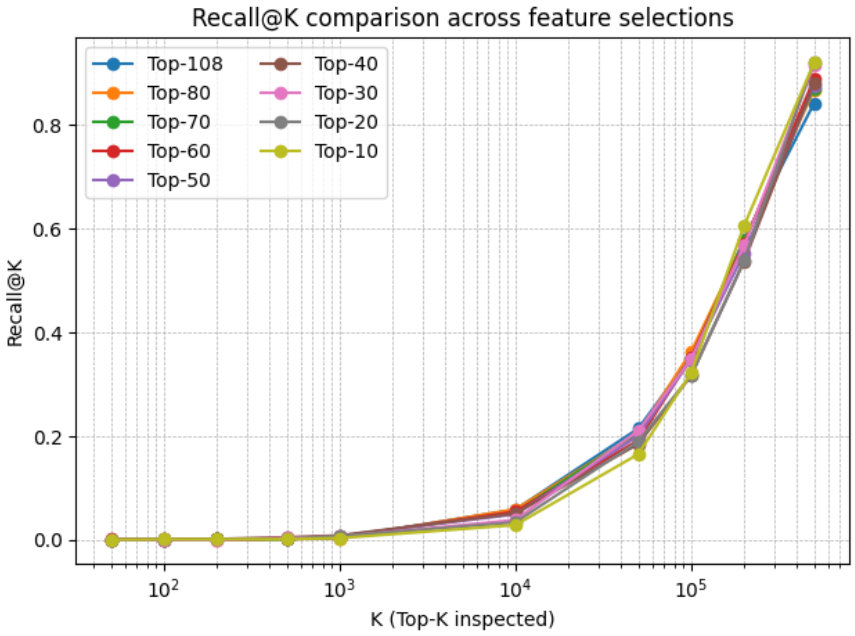


Figure 8: Recall@$K$ of the baseline autoencoder across different feature subset sizes.

### 6.1.3 Separation score statistics

Finally, the selected baseline model was evaluated on the test set. Table 6 summarizes the global score-separation performance. The model achieves an AUC of 0.747, indicating that reconstruction error captures some useful discrimination between labeled piracy and unlabeled customers. In particular, the median reconstruction score of piracy customers is about 2.78 times that of unlabeled customers. However, this global separation does not translate into strong early ranking performance: the baseline retrieves only 1 piracy customer in the top 50 accounts and 4 piracy customers in the top 1,000 accounts. Taken together, these results show that the baseline autoencoder captures some anomaly signal from customer subscription behavior, but its practical early-ranking capability remains limited.

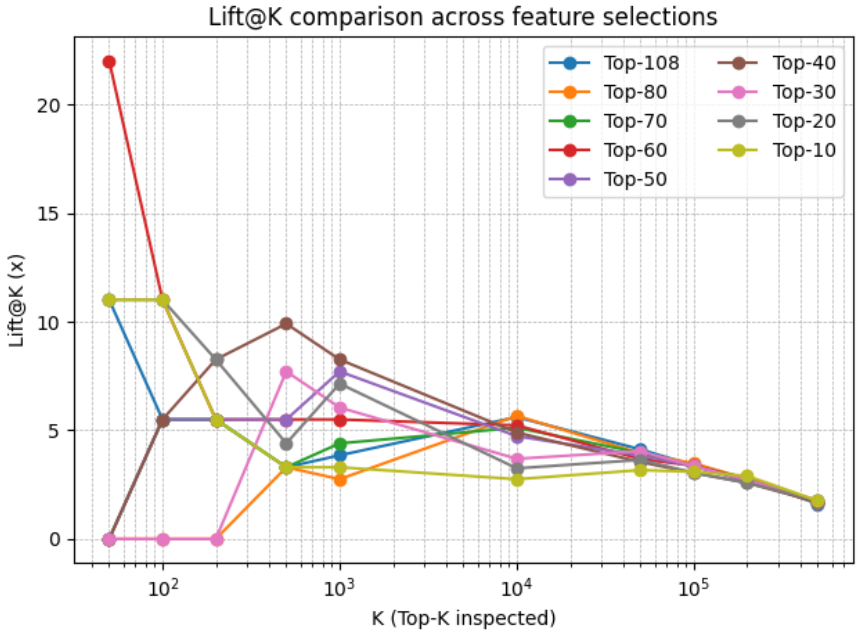


Figure 9: Lift@$K$ of the baseline autoencoder across different feature subset sizes.

Table 6: Global score-separation statistics for the baseline autoencoder on the test set.

| **Metric** | **Value** |
|---|---|
| AUC | 0.746962 |
| Median score (unlabeled) | 0.004157 |
| Median score (piracy) | 0.011575 |
| Median ratio (piracy / unlabeled) | 2.784 |
| TPR @ non-p99 (1% FPR) | 0.0454 |
| TPR @ non-p99.5 (0.5% FPR) | 0.0281 |

## 6.2 Effect of the blending coefficient $\lambda_{\text{rank}}$

To examine how the balance between the FeaWAD-style objective and the PReNet-style objective on our hybrid model affects performance, an ablation experiment was conducted by varying the blending coefficient $\lambda_{\text{rank}}$ from 0 to 1. Recall that the hybrid objective is defined as

$$\mathcal{L}_{\text{hybrid}} = (1 - \lambda_{\text{rank}})\,\mathcal{L}_{\text{FeaWAD}} + \lambda_{\text{rank}}\,\mathcal{L}_{\text{PReNet}}.$$

Thus, $\lambda_{\text{rank}} = 0$ corresponds to a purely FeaWAD-style objective on the unified architecture, while $\lambda_{\text{rank}} = 1$ corresponds to a purely PReNet-style objective on the same shared model.

Table 7 reports the number of labeled piracy customers found in the top-$K$ ranked accounts for several representative inspection budgets. Figure 10 provides the corresponding visual comparison.

Table 7: Effect of the blending coefficient $\lambda_{\text{rank}}$ on hybrid-model performance. The table reports the number of labeled piracy customers found in the top-$K$ ranked accounts on the test set, with Recall@$K$ shown in parentheses.

| $\lambda_{\text{rank}}$ | **Top-50** | **Top-1,000** | **Top-10,000** | **Top-100,000** | **Top-500,000** |
|---|---|---|---|---|---|
| 0 | 7 (0.0040) | 140 (0.0804) | 707 (0.4061) | 1409 (0.8093) | 1734 (0.9960) |
| 0.1 | 7 (0.0040) | 116 (0.0666) | 673 (0.3866) | 1409 (0.8093) | 1658 (0.9523) |
| 0.2 | 18 (0.0103) | 188 (0.1080) | 717 (0.4118) | 1399 (0.8036) | 1732 (0.9948) |
| 0.42 | 22 (0.0126) | 187 (0.1074) | 756 (0.4342) | 1417 (0.8139) | 1620 (0.9305) |
| 0.5 | 8 (0.0046) | 187 (0.1074) | 739 (0.4245) | 1368 (0.7858) | 1675 (0.9621) |
| 0.7 | 14 (0.0080) | 186 (0.1068) | 765 (0.4394) | 1468 (0.8432) | 1572 (0.9029) |
| 0.9 | 8 (0.0046) | 149 (0.0856) | 747 (0.4291) | 1492 (0.8570) | 1622 (0.9316) |
| 1.0 | 16 (0.0092) | 182 (0.1045) | 740 (0.4250) | 1517 (0.8713) | 1684 (0.9673) |

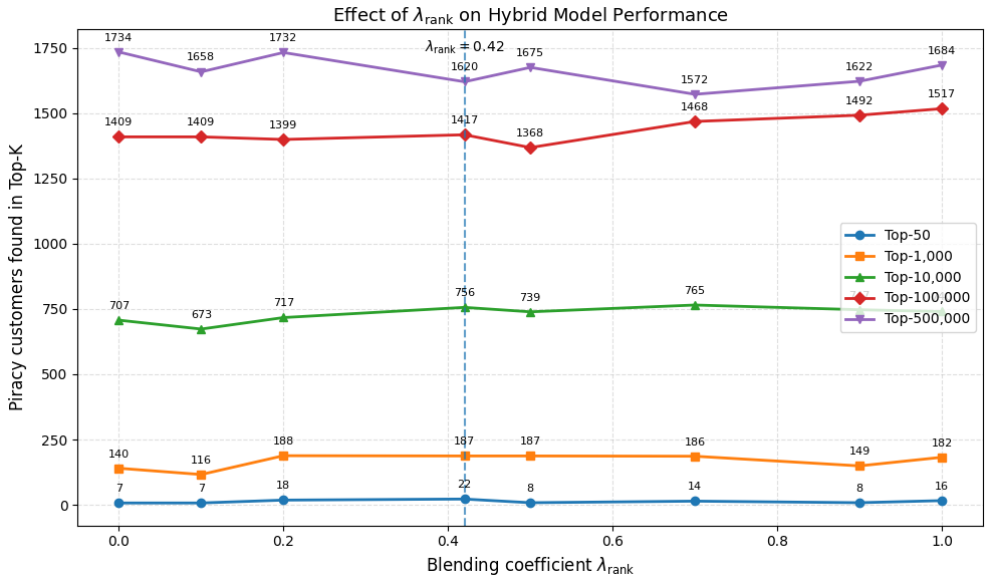


Figure 10: Effect of the blending coefficient $\lambda_{\text{rank}}$ on hybrid-model performance at different inspection budgets.

Several observations can be made from these results. First, the model performance changes substantially as $\lambda_{\text{rank}}$ varies, which confirms that the balance between the two objectives is important. Second, neither extreme is uniformly optimal. When $\lambda_{\text{rank}} = 0$, the model behaves like a pure FeaWAD-style objective and performs strongly at very large budgets, but its very early ranking performance is weaker than the best mixed settings. When $\lambda_{\text{rank}} = 1$, the model becomes purely PReNet-style and improves some larger-budget retrieval results, but it does not achieve the strongest overall balance across early and medium budgets.

Third, intermediate values of $\lambda_{\text{rank}}$ generally provide better trade-offs. In particular, $\lambda_{\text{rank}} = 0.42$ gives the strongest top-50 result and strong performance at top-1,000, while $\lambda_{\text{rank}} = 0.7$ gives the best top-10,000 result among the tested settings. This indicates that reconstruction-aware learning and pairwise ranking supervision are complementary: giving non-zero weight to both objectives improves the concentration of piracy customers near the top of the ranking.

Finally, the results suggest that the best value of $\lambda_{\text{rank}}$ depends on the operational objective. Smaller or intermediate values favor stronger early prioritization, while larger values move the model closer to the behavior of the pairwise ranking objective and may improve performance at larger inspection budgets. Overall, the ablation study supports the usefulness of the hybrid design and shows that the blending coefficient is a meaningful control parameter rather than a purely technical detail.

## 6.3 Top-$K$ comparison across models

Table 8 reports the number of labeled piracy customers retrieved in the top-$K$ ranked accounts for all four models, together with the corresponding Recall@$K$ values. To provide a more direct visual comparison, Figure 11 plots Recall@$K$ as a function of the inspection budget $K$.

Table 8: Number of labeled piracy customers found in the top-$K$ ranked accounts on the test set. Recall@$K$ is shown in parentheses.

| $K$ | **Baseline** | **FeaWAD** | **PReNet** | **Hybrid** |
|---|---|---|---|---|
| 50 | 1 (0.0006) | 11 (0.0063) | 17 (0.0098) | 22 (0.0126) |
| 100 | 2 (0.0011) | 16 (0.0092) | 30 (0.0172) | 32 (0.0184) |
| 200 | 3 (0.0017) | 26 (0.0149) | 43 (0.0247) | 51 (0.0293) |
| 500 | 3 (0.0017) | 50 (0.0287) | 72 (0.0414) | 111 (0.0638) |
| 1,000 | 4 (0.0023) | 88 (0.0505) | 111 (0.0638) | 187 (0.1074) |
| 10,000 | 80 (0.0460) | 494 (0.2837) | 488 (0.2803) | 756 (0.4342) |
| 50,000 | 372 (0.2137) | 1027 (0.5899) | 1037 (0.5956) | 1255 (0.7209) |
| 100,000 | 649 (0.3728) | 1160 (0.6663) | 1278 (0.7341) | 1417 (0.8139) |
| 200,000 | 1000 (0.5744) | 1277 (0.7335) | 1491 (0.8564) | 1541 (0.8851) |
| 500,000 | 1463 (0.8403) | 1475 (0.8472) | 1681 (0.9655) | 1620 (0.9305) |

As shown in Table 8 and Figure 11, the weakly supervised models substantially outperform the baseline at all practically relevant inspection budgets. FeaWAD already provides a large improvement over the reconstruction-only baseline. For example, at $K = 1{,}000$, FeaWAD retrieves 88 piracy customers, compared with only 4 for the baseline.

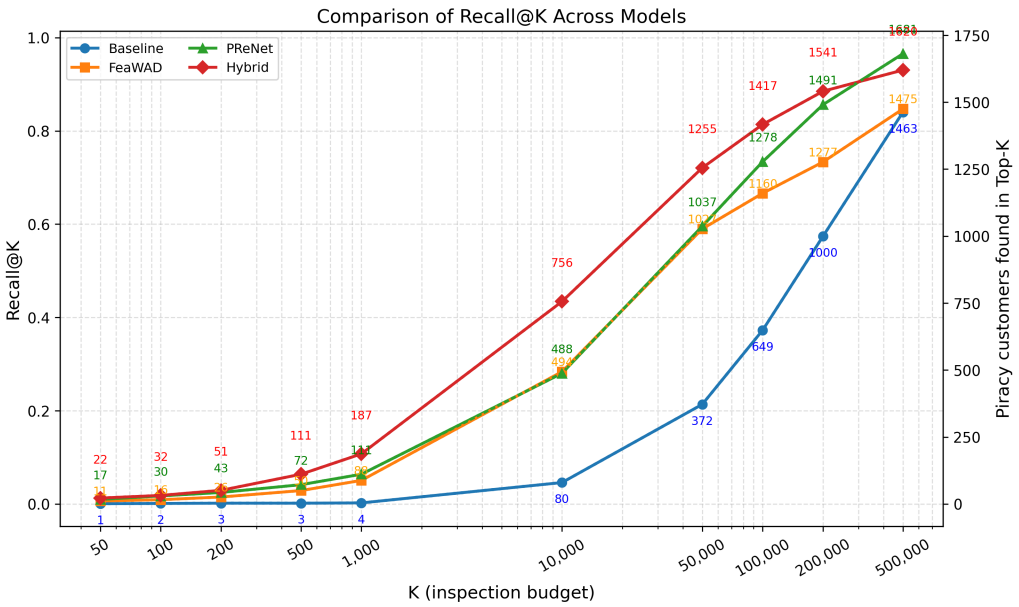


Figure 11: Recall@$K$ comparison across the four models on the test set.

PReNet improves further at most budgets, especially in the early part of the ranking. At $K = 100$, it finds 30 piracy customers, compared with 16 for FeaWAD. The unified hybrid model achieves the strongest early- and mid-budget performance. It retrieves 22 piracy customers in the top 50, 187 in the top 1,000, and 756 in the top 10,000. These results are higher than those of both FeaWAD and PReNet at the same budgets.

At very large budgets, however, PReNet becomes slightly stronger. At $K = 500{,}000$, it retrieves 1,681 piracy customers, compared with 1,620 for the hybrid model. Overall, the figure makes the same pattern more explicit: the baseline remains clearly below the weakly supervised methods, the hybrid model performs best over most small and medium budgets, and PReNet becomes the strongest model at the largest tested budget.

## 6.4 Lift@$K$ comparison across models

Table 9 reports Lift@$K$ for all four models, and Figure 12 provides the corresponding visual comparison. Since the overall piracy rate in the evaluation pool is only about 0.18%, Lift@$K$ offers a more operationally meaningful view of ranking quality by showing how much more piracy-dense the top-$K$ list is compared with random selection.

As shown in Table 9 and Figure 12, the Lift@$K$ results confirm the same overall pattern as the Recall@$K$ comparison. The baseline performs better than random ranking, but its enrichment remains modest except at the smallest inspection budgets. In contrast, FeaWAD and PReNet both achieve strong enrichment, especially at small and medium $K$, indicating that weak supervision substantially improves the concentration of piracy customers near the top of the ranking.

The hybrid model achieves the highest Lift@$K$ from $K = 50$ up to $K = 200{,}000$. In particular,

Table 9: Lift@$K$ on the test set.

| $K$ | **Baseline** | **FeaWAD** | **PReNet** | **Hybrid** |
|---|---|---|---|---|
| 50 | 11.01 | 121.08 | 187.13 | 242.17 |
| 100 | 11.01 | 88.06 | 165.11 | 176.12 |
| 200 | 8.26 | 71.55 | 118.33 | 140.35 |
| 500 | 3.30 | 55.04 | 79.25 | 122.18 |
| 1,000 | 2.20 | 48.43 | 61.09 | 102.92 |
| 10,000 | 4.40 | 27.19 | 26.86 | 41.61 |
| 50,000 | 4.09 | 11.30 | 11.41 | 13.81 |
| 100,000 | 3.57 | 6.38 | 7.03 | 7.80 |
| 200,000 | 2.75 | 3.51 | 4.10 | 4.24 |
| 500,000 | 1.61 | 1.62 | 1.85 | 1.78 |

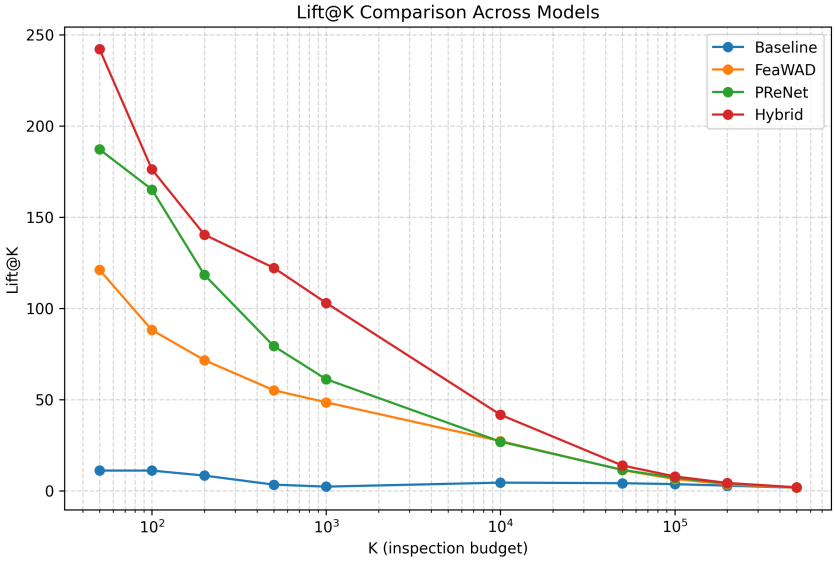


Figure 12: Lift@$K$ comparison across the four models on the test set.

it reaches 242.17× at $K = 50$, 102.92× at $K = 1{,}000$, and 41.61× at $K = 10{,}000$. These results show that the top-ranked accounts produced by the hybrid model are much more piracy-dense than would be expected under random selection.

# 7 Discussion

## 7.1 Main findings and interpretation

The experimental results reveal a consistent progression in performance across the evaluated models, highlighting the importance of incorporating weak supervision into anomaly detection for this task. While the baseline autoencoder is able to capture a certain degree of separation between labeled piracy and unlabeled customers, its performance in early ranking remains limited. This suggests that reconstruction error alone, although informative, is not sufficiently aligned with the operational objective of prioritizing high-risk accounts under extreme class imbalance.

In contrast, both FeaWAD and PReNet significantly improve ranking quality, but through different mechanisms. FeaWAD enhances representation learning by incorporating reconstruction residual information and explicitly optimizing score deviation relative to a reference distribution. This leads to better global separation between suspicious and typical behavior. However, its objective remains indirectly related to the final ranking goal.

PReNet, on the other hand, directly optimizes pairwise ordering between labeled piracy and unlabeled customers. This alignment between training objective and evaluation metric explains its strong performance, particularly in early and large-budget ranking scenarios. By learning from pairwise relations, PReNet is able to exploit limited labeled data more effectively and generalize beyond observed piracy patterns.

The hybrid model combines these two learning principles within a shared architecture, and the results suggest that they are complementary rather than redundant. Reconstruction-aware feature encoding provides a structured representation of normal behavior and deviations, while pairwise ranking supervision reshapes this representation to emphasize separation in the high-risk region of the score distribution. This synergy is most visible in early and medium inspection budgets, where the hybrid model consistently achieves the highest Recall@$K$ and Lift@$K$.

## 7.2 Answering the research questions

The research questions introduced in Section 1 can now be answered based on the data processing pipeline, model design, and experimental results.

**RQ1: How can large-scale raw tabular business data be integrated, processed, and transformed into customer-level behavioral features suitable for anomaly**

**detection?** The raw operational data can be transformed into model-ready customer-level features through a multi-stage processing pipeline. In this thesis, raw customer, account, subscription-event, migration, subscription, and product-channel records were first converted into a scalable storage format and then linked through common business identifiers. Subscription events were converted into time segments, migration records were used to rectify agreement histories, and agreement-level behavior was aggregated to the customer level. The feature engineering process produced candidate features describing subscription activity, event-type distributions, migration behavior, concurrency, recent-versus-historical changes, sports exposure, and account characteristics. After preprocessing and filtering, the final model input contained 108 customer-level features. This shows that large-scale tabular business data can be made suitable for anomaly detection, but only after careful temporal processing, migration rectification, aggregation, and leakage-aware preprocessing.

**RQ2: To what extent can anomaly detection models identify piracy-related behavioral patterns under production-scale data, strong class imbalance, and partial labeling?** The results show that anomaly detection models can identify piracy-related behavioral patterns to a meaningful extent, but their effectiveness depends strongly on the learning objective. The baseline autoencoder captures some separation between labeled piracy customers and unlabeled customers, which indicates that customer subscription behavior contains useful anomaly signals. However, its early-ranking performance remains limited. This suggests that reconstruction error alone is not sufficiently aligned with the practical goal of prioritizing the most suspicious customers under strong class imbalance and partial labeling. In contrast, weakly supervised methods make more effective use of the available piracy labels and achieve substantially better ranking performance.

**RQ3: Can a new weakly supervised anomaly detection model be designed to improve suspicious-customer ranking beyond existing weakly supervised methods in this application setting?** The proposed unified FeaWAD–PReNet model shows that combining reconstruction-aware feature encoding with pairwise ranking supervision can improve suspicious-customer ranking in this application setting. Compared with FeaWAD and PReNet individually, the hybrid model achieves the strongest performance across most small and medium inspection budgets, which are the most relevant budgets for practical investigation. This indicates that the two learning principles are complementary: FeaWAD contributes reconstruction-aware deviation information, while PReNet contributes direct pairwise ranking supervision.

## 7.3 Why the hybrid model improves early ranking

A key observation from the results is that the hybrid model achieves the strongest performance in the top portion of the ranking, which is the most critical region for practical investigation. This can be interpreted as a consequence of combining global and local learning signals.

The FeaWAD component encourages the model to learn a smooth and structured anomaly score distribution, where most unlabeled samples remain close to a reference center and de-

viations are captured through reconstruction-aware features. This provides a stable global structure.

The PReNet component, in contrast, introduces local pairwise constraints that explicitly push labeled piracy samples above unlabeled samples. These constraints are particularly effective in shaping the upper tail of the score distribution, where high-risk customers are located.

When combined, the model benefits from both effects: the global structure provided by FeaWAD prevents overfitting to noisy pairwise signals, while the pairwise supervision provided by PReNet sharpens the separation at the top of the ranking. This explains why the hybrid model concentrates more piracy cases in the highest-ranked positions.

## 7.4 Implications for real-world anti-piracy workflows

From an operational perspective, the results demonstrate that weakly supervised anomaly ranking can significantly improve the efficiency of anti-piracy investigation. Given the extremely low base rate of piracy in the full population, random inspection is highly inefficient. The observed Lift@$K$ values indicate that the proposed models can increase piracy concentration in the top-ranked accounts by two orders of magnitude at small inspection budgets.

This has direct practical implications. For example, when investigators can only review a limited number of accounts per day, a high-quality ranking model can substantially increase the number of confirmed piracy cases discovered within the same resource constraints. In this sense, the model does not replace human investigation, but acts as a prioritization tool that improves the allocation of investigative effort.

In addition, the results suggest that even a small number of labeled piracy cases can provide substantial value when used through weakly supervised learning. This is particularly relevant in real-world settings, where obtaining reliable labels is often expensive and time-consuming.

## 7.5 Trade-offs between models

Although the hybrid model performs best over most practically relevant budgets, the results also reveal a trade-off. At very large inspection budgets, PReNet slightly outperforms the hybrid model. This suggests that the hybrid objective places more emphasis on separating the highest-risk customers, potentially at the expense of slightly reduced coverage in the lower-risk region.

This trade-off implies that the choice of model should depend on the operational objective. If the goal is to maximize the effectiveness of a small investigation team focusing on the highest-risk accounts, the hybrid model is preferable. If the goal is to achieve broader coverage across a large portion of the population, a pure pairwise ranking approach such as PReNet may be more suitable.

## 7.6 Feature representation and data insights

The feature-sweep experiments indicate that piracy-related signals are concentrated in a subset of engineered features, while the remaining features contribute less significantly or provide redundant information. This suggests that certain behavioral patterns, such as subscription dynamics, account sharing indicators, or specific product usage patterns, are more strongly associated with piracy behavior.

At the same time, the relatively stable performance across different feature subsets indicates that the models are robust to feature redundancy. This is particularly important in large-scale operational systems, where feature engineering pipelines may evolve over time and include partially overlapping signals.

## 7.7 Limitations

Despite the promising results, several limitations remain.

First, piracy is not directly observed in the available data. It must be inferred from behavioral proxies derived from subscription history, migrations, sports exposure, concurrency, and account-level variables. This makes the quality of the final detection system highly dependent on the available operational data and the effectiveness of the engineered features.

Second, the current study uses customer-level aggregated features. This representation is suitable for account ranking, but it compresses temporal information. Some short-term or sequential piracy patterns may be weakened when raw event histories are summarized into static customer-level variables.

Third, the empirical study is based on a fixed split and a specific unlabeled sampling setup. Although this is sufficient for comparative evaluation, the reported results may still depend to some extent on the chosen split, sampling ratio, and hyperparameter configuration.

Finally, the hybrid model introduces additional design choices, such as the blending coefficient between the FeaWAD and PReNet objectives. While the current results are promising, further experiments would be needed to understand how stable this design is across different datasets, parameter settings, and operating conditions.

# 8 Conclusion

This work investigated broadcast piracy detection from customer subscription behavior as a weakly supervised anomaly ranking problem. Rather than formulating the task as standard binary classification, the focus was placed on ranking customers by suspicion score so that limited investigative resources can be directed to the most promising accounts. To support this objective, multiple operational data sources were transformed into a customer-level dataset, and several anomaly-oriented models were evaluated under the same ranking setting.

The experimental results show that customer subscription behavior contains useful signal for piracy screening. The baseline autoencoder was able to separate labeled piracy customers from the unlabeled population to some extent, but its early-ranking performance was limited. This indicates that reconstruction error alone is not sufficient for practical prioritization in a highly imbalanced and partially labeled setting.

Weakly supervised anomaly detection methods proved substantially more effective. Both FeaWAD and PReNet clearly outperformed the reconstruction-based baseline across practically relevant top-$K$ budgets. FeaWAD benefited from reconstruction-aware feature encoding and deviation-based score learning, while PReNet showed the value of pairwise supervision for exploiting scarce labeled piracy samples.

The unified FeaWAD–PReNet hybrid model further improved performance over most practically relevant budgets. Its results suggest that reconstruction-aware encoding and pairwise ranking supervision can be combined effectively within a shared architecture. Overall, the findings indicate that weakly supervised anomaly ranking is a promising approach for large-scale anti-piracy screening, and that the proposed hybrid design provides a useful direction for further improvement.

Several directions remain for future work. One natural next step is to evaluate the models under additional data splits and sampling settings to better understand their robustness. Another direction is to incorporate richer temporal modeling, since the current customer-level representation compresses sequential event information into aggregated features. It would also be valuable to further explore the design space of the hybrid model, especially the balance between reconstruction-based and pairwise objectives, and to study whether additional domain-specific features or more reliable labels can further improve ranking quality in real operational settings.

# References


[1] Anggo Doyoharjo and FX Hastowo Broto Laksito. "Illegal Streaming Sites: Legal Analysis and Challenges in the Digital Era". In: *Jembatan Hukum: Kajian ilmu Hukum, Sosial dan Administrasi Negara* 2.3 (2025), pp. 216–225.

[2] Uchenna Oluchukwu Okechukwu. "Effective Approaches to Tackle Digital Piracy in the Media and Entertainment Industry". In: *NIGERIAN JOURNAL OF ARTS AND HUMANITIES (NJAH)* 4.3 (2024).

[3] Albert Hasoloan Limbong. "ESPN's Broadcast Rights Protection in Intellectual Property Law Over World Cup Broadcast Piracy". In: *Journal of Law, Politic and Humanities* 5.4 (2025), pp. 3106–3116.

[4] Nakia Stokes. "Strategies to Reduce the Impact of Digital Piracy in the Media Industry". PhD thesis. Walden University, 2024.

[5] Zachary Nolan, Jonathan W Williams, and Haoran Zhang. *The Impact of Video Piracy on Content Producers and Distributors*. 2022.

[6] Milosh Stolikj, Dmitri Jarnikov, and Andrew Wajs. "Artificial intelligence for detecting media piracy". In: *SMPTE Motion Imaging Journal* 127.6 (2018), pp. 22–27.

[7] TP Rani et al. "Piracy Estimation and Detection in Movies and OTT Series using Similarity Functions and Deep Learning Techniques". In: *2024 International Conference on Emerging Research in Computational Science (ICERCS)*. IEEE. 2024, pp. 1–6.

[8] B Sanjana et al. "Real-Time Piracy Detection Based on Thermogram Analysis and Machine Learning Techniques". In: *International Conference on Computational Intelligence in Communications and Business Analytics*. Springer. 2022, pp. 336–350.

[9] Igor Slabykh. "The new approaches to digital anti-piracy in the entertainment industry". In: *UIC Rev. Intell. Prop. L.* 19 (2019), p. 75.

[10] Bruno Manuel Degardin. "Weakly and partially supervised learning frameworks for anomaly detection". MA thesis. Universidade da Beira Interior (Portugal), 2020.

[11] Guansong Pang et al. "Learning representations of ultrahigh-dimensional data for random distance-based outlier detection". In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 2041–2050.

[12] Boyang Liu et al. "Rca: A deep collaborative autoencoder approach for anomaly detection". In: *IJCAI: proceedings of the conference*. Vol. 2021. 2021, p. 1505.

[13] Tom Shenkar and Lior Wolf. "Anomaly detection for tabular data with internal contrastive learning". In: *International conference on learning representations*. 2022.

[14] Kamal Berahmand et al. "Autoencoders and their applications in machine learning: a survey". In: *Artificial intelligence review* 57.2 (2024), p. 28.

[15] Hasan Torabi, Seyedeh Leili Mirtaheri, and Sergio Greco. "Practical autoencoder based anomaly detection by using vector reconstruction error". In: *Cybersecurity* 6.1 (2023), p. 1.

[16] Minqi Jiang et al. "Weakly supervised anomaly detection: A survey". In: *arXiv preprint arXiv:2302.04549* (2023).

[17] Guansong Pang, Chunhua Shen, and Anton Van Den Hengel. "Deep anomaly detection with deviation networks". In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2019, pp. 353–362.

[18] Yingjie Zhou et al. "Feature encoding with autoencoders for weakly supervised anomaly detection". In: *IEEE Transactions on Neural Networks and Learning Systems* 33.6 (2021), pp. 2454–2465.

[19] Guansong Pang et al. "Deep weakly-supervised anomaly detection". In: *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*. 2023, pp. 1795–1807.

[20] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey". In: *ACM computing surveys (CSUR)* 41.3 (2009), pp. 1–58.

[21] Eleazar Eskin. "Anomaly detection over noisy data using learned probability distributions". In: (2000).

[22] Deepak Agarwal. "Detecting anomalies in cross-classified streams: a bayesian approach". In: *Knowledge and information systems* 11.1 (2007), pp. 29–44.

[23] Mingxi Wu and Christopher Jermaine. "Outlier detection by sampling with accuracy guarantees". In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2006, pp. 767–772.

[24] Eleazar Eskin et al. "A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data". In: *Applications of data mining in computer security*. Springer, 2002, pp. 77–101.

[25] Dragoljub Pokrajac, Aleksandar Lazarevic, and Longin Jan Latecki. "Incremental local outlier detection for data streams". In: *2007 IEEE symposium on computational intelligence and data mining*. IEEE. 2007, pp. 504–515.

[26] Anny Lai-mei Chiu and Ada Wai-chee Fu. "Enhancements on local outlier detection". In: *Seventh International Database Engineering and Applications Symposium, 2003. Proceedings.* IEEE. 2003, pp. 298–307.

[27] Volker Roth. "Outlier detection with one-class kernel fisher discriminants". In: *Advances in Neural Information Processing Systems* 17 (2004).

[28] Volker Roth. "Kernel fisher discriminants for outlier detection". In: *Neural computation* 18.4 (2006), pp. 942–960.

[29] Guansong Pang et al. "Deep Learning for Anomaly Detection: A Review". In: *ACM Computing Surveys* 54.2 (2021), pp. 1–38. DOI: 10.1145/3439950.

[30] Abhaya Abhaya and Bidyut Kr Patra. "An efficient method for autoencoder based outlier detection". In: *Expert Systems with Applications* 213 (2023), p. 118904.

[31] CMS ECAL Collaboration. "Autoencoder-Based Anomaly Detection System for Online Data Quality Monitoring of the CMS Electromagnetic Calorimeter". In: *Computing and Software for Big Science* 8.1 (2024), p. 11. DOI: 10.1007/s41781-024-00118-z.

[32] Huitaek Yun et al. “Autoencoder-based anomaly detection of industrial robot arm using stethoscope based internal sound sensor”. In: *Journal of Intelligent Manufacturing* 34.3 (2023), pp. 1427–1444.

[33] Lukas Ruff et al. “Deep Semi-Supervised Anomaly Detection”. In: *International Conference on Learning Representations (ICLR)*. 2020.

[34] Haihong Zhao et al. “Weakly Supervised Anomaly Detection via Knowledge-Data Alignment”. In: *Proceedings of the ACM Web Conference 2024*. 2024, pp. 4083–4094. DOI: 10.1145/3589334.3645429.

[35] Jingkai Chi and Zhizhong Mao. “Weakly-supervised anomaly detection based on adversarial transfer learning”. In: *Engineering Applications of Artificial Intelligence* 155 (2025), p. 110975.

[36] Hongzuo Xu et al. “RoSAS: Deep semi-supervised anomaly detection with contamination-resilient continuous supervision”. In: *Information Processing & Management* 60.5 (2023), p. 103459.

[37] Walid Durani et al. “Weakly Supervised Anomaly Detection via Dual-Tailed Kernel”. In: *Forty-second International Conference on Machine Learning*. 2025.

[38] Deepak Vohra. “Apache Parquet”. In: *Practical Hadoop Ecosystem: A Definitive Guide to Hadoop-Related Frameworks and Tools*. Berkeley, CA: Apress, 2016, pp. 325–335. DOI: 10.1007/978-1-4842-2199-0_8.