Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Optimizing Threshold-based Financial Crime Detection: Smoothing Rank Scale Gradient Descent for Discrete Loss Function-Based Threshold Adjustment

Yen-Wu Ti
Xiamen Ocean Vocational College
Tian-Shyr Dai
cameldai@nycu.edu.tw
National Yang Ming Chiao Tung University
Chin-Yi Hsieh
National Yang Ming Chiao Tung University
Mao-Chia Wu
National Yang Ming Chiao Tung University
Ying-Ping Chen
National Yang Ming Chiao Tung University
Hsin-Yun Chang
MetaEdge Corporation

Research Article

Keywords: Anti-Money Laundering, Threshold Optimization, Financial Crime Detection, Discrete Optimization, Regulatory Compliance, Gradient-Based Methods

Posted Date: August 14th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8997864/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Optimizing Threshold-based Financial Crime Detection: Smoothing Rank Scale Gradient Descent for Discrete Loss Function-Based Threshold Adjustment

Wen-Wu Ti[1], Tian-Shyr Dai[2,3], Chin-Yi Hsieh[4], Mao-Chia Wu[2], Ying-Ping Chen[5] and Hsin Yun Chang[6]

[1]School of Information Engineering, Xiamen Ocean Vocational College, Xiamen , 361100, China.
[2]Department of Information Management and Finance, National Yang Ming Chiao Tung University, Hsinchu, 300093, Taiwan.
[3]Risk and Insurance Research Center, National Chengchi University, Taipei, 116011, Taiwan.
[4]Institute of Computer Science and Engineering, National Yang Ming Chiao Tung University, Hsinchu, 300093, Taiwan.
[5]Department of Computer Science, National Yang Ming Chiao Tung University, Hsinchu, 300093, Taiwan.
[6]Supervision Division I, MetaEdge Corporation, Taipei, 105, Taiwan.

Contributing authors: tiyenwu@gmail.com; cameldai@nycu.edu.tw; eason7588@gmail.com; markwu0821@gmail.com; ypchen@cs.nycu.edu.tw; Cynthia.chang@metaedge.com.tw;

Abstract

Financial institutions employ threshold-based detection for anti-money laundering (AML) and counter-terrorist financing (CTF) to meet regulatory interpretability requirements, creating a discrete optimization challenge for balancing detection accuracy with operational costs. Regulatory agencies define suspicious transaction patterns requiring thresholds on

transaction frequency and amounts. However, static thresholds are susceptible to reverse engineering, necessitating dynamic adjustment to balance recall, Human Resource Savings (HRS), and precision. The resulting discrete loss function exhibits dramatically varying gradients, complicating optimization. Traditional methods require exhaustive enumeration, while simulated annealing suffers from slow convergence and instability. We propose Smoothing Rank Scale Gradient Descent (SRSGD), transforming the discrete loss function into a continuous one through sigmoid-based approximation. Uneven data distribution creates rugged loss landscapes with dramatically varying gradients, making learning rate adjustment difficult and explaining simulated annealing's systematic failures. SRSGD addresses this through two essential components: (1) rank-scale transformation that handles extreme gradient variations caused by uneven data density, enabling effective learning rate tuning, and (2) quantile-based initial threshold selection providing effective starting points. Optimized thresholds are determined in rank space then converted to actual values. SRSGD achieves 4000× computational speedup over discrete gradient methods while delivering superior HRS and maintaining regulatory-compliant recall. This framework has been successfully integrated into AML systems at financial institutions, enhancing financial crime detection while adhering to Financial Supervisory Commission standards.

Keywords: Anti-Money Laundering, Threshold Optimization, Financial Crime Detection, Discrete Optimization, Regulatory Compliance, Gradient-Based Methods

1 Introduction

Money laundering involves concealing assets obtained through illegal activities by channeling them through complex transactions, such as purchasing assets or transferring funds across multiple accounts, to make them appear legitimate and avoid detection. Similarly, terrorist financing focuses on providing funds, often through legal or illicit means, to support terrorist activities. To address these threats, regulatory agencies worldwide have established stringent frameworks requiring financial institutions to implement anti-money laundering and counter-terrorist financing measures. These regulations mandate institutions to identify, monitor, and report suspicious activities, safeguarding the integrity of the global financial system and preventing its exploitation for criminal purposes.

Financial regulatory frameworks worldwide mandate threshold-based detection mechanisms for anti-money laundering to ensure interpretability and regulatory accountability. For instance, in the United States, the Federal Financial Institutions Examination Council provides the comprehensive BSA/AML Examination Manual, while the Financial Crimes Enforcement Network issues various Alerts, Advisories, Notices, Bulletins, and Fact Sheets

to guide financial institutions in effectively implementing AML and CTF measures. Threshold-based frameworks are employed in countries like Canada, Singapore, Australia, and Taiwan, where financial institutions are required to set specific thresholds and monitoring rules to identify suspicious activities. These systems prioritize compliance and interpretability, ensuring alignment with both national regulations and international standards while providing a transparent structure to combat financial crimes effectively.

In Taiwan, the Money Laundering Control Act and the Regulations Governing Anti-Money Laundering of Financial Institutions have been enacted to combat financial crime. Financial institutions are required to adhere to the guidelines outlined in the document, "Type of Suspicious Money Laundering or Terrorism Financing Transactions (in Chinese)," published by the Financial Supervisory Commission. These abstract guidelines serve as a foundation for financial institutions to establish executable rules, such as setting thresholds for identifying abnormal activities (e.g., frequency and amounts of withdrawals). As detailed in Section 2.2, these measures are critical for monitoring customer transactions and detecting suspicious criminal activities.

Figure 1 presents a hypothetical example demonstrating how banks use threshold settings to determine suspicious events. This example illustrates the balance between missing fraudulent transactions and falsely flagging normal transactions. In this scenario, the bank deems all transactions from any bank account (whether associated with a legal entity or a natural person) as suspicious within a specific timeframe if certain transaction characteristics, such as the number of transactions (represented on the x-axis) and accumulated transaction amounts (represented on the y-axis), exceed predefined thresholds.

Consider a setup involving four fraudulent and eight regular instances, depicted as blue and black nodes, respectively. These instances are evaluated by a threshold-based detection system that flags excessive transaction counts and amounts as suspicious. Those flagged are labeled as 'SAM' (Suspicious Activity Monitoring) cases and are subjected to further scrutiny by the bank's legal department. Thresholds are established as intervals between adjacent values on the x and y axes, denoted by lowercase letters in the figure. In this threshold-based system, the placement of a threshold within any given interval does not affect the determination of whether instances are flagged as SAM. This ensures that the identification process remains consistent regardless of the exact threshold location within the same range.

For example, label 'c' denotes a transaction count threshold in the range (10, 15), while label 'u' specifies an accumulated transaction amount threshold in the range (400,000, 500,000) dollars.

Instances whose features surpass these thresholds, highlighted by solid red lines, are marked as suspicious by the threshold-based system and subsequently manually reviewed by compliance officers. As a result, four SAM instances are scrutinized, leading to the identification of two fraudulent instances and

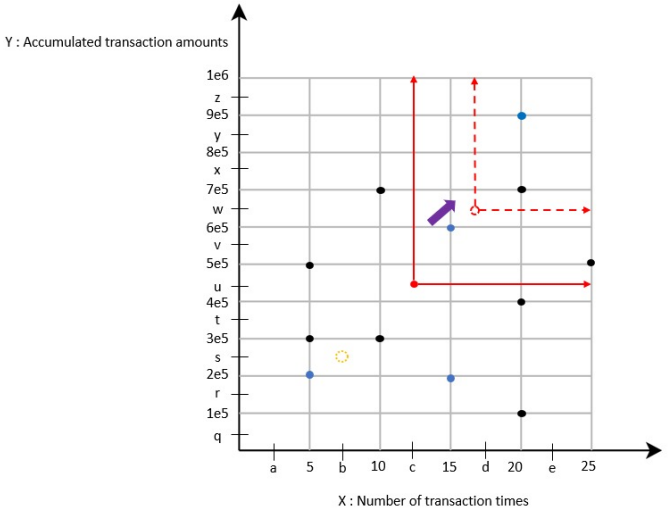


Fig. 1 A Hypothetical threshold-based SAM Detection Example This diagram illustrates the operation of a threshold-based system for detecting suspicious activity monitoring (SAM) cases. Blue nodes represent fraudulent instances, and black nodes denote normal instances. The x-axis indicates the number of transactions, while the y-axis represents accumulated transaction amounts. Lowercase letters along the axes denote different threshold ranges, which establish criteria for flagging transactions as suspicious. Solid and dashed red lines illustrate two distinct threshold combinations used to identify potential SAM cases.

the unwarranted scrutiny of two legitimate instances. However, two fraudulent instances evade detection by the threshold-based system and consequently avoid further scrutiny.

Recall that thresholds must be dynamically adjusted to prevent criminal organizations from detecting and bypassing the threshold-based system. Additionally, adjustments to thresholds must balance recall (capturing fraudulent instances), precision (minimizing disruptions to normal transactions), and operational costs (reducing manual reviews). For example, in Figure 1, raising the thresholds from (c, u) shown as solid red lines to (d, w) shown as dashed red lines could reduce the number of SAM events from four to two, thereby lowering bank review costs and reducing disturbances to normal transactions. However, this adjustment comes with a trade-off: while it reduces false positives from 2 to 1, it also results in missing one fraudulent instance that was previously detected, decreasing the recall rate from 50% (2 out of 4 fraudulent instances detected) to 25% (1 out of 4 detected). To increase recall, thresholds could be lowered, but this would significantly increase review costs. Therefore, dynamically adjusting thresholds based on changes in criminal behavior to minimize labor costs while ensuring an acceptable level of oversight on genuine money laundering cases becomes a critical concern for threshold-based fraud detection systems.

Employing a true brute-force enumeration—exhaustively evaluating all possible threshold configurations across all unique feature values, such as transaction counts a, b, ..., e and accumulated amounts q, r, ..., z as depicted in Figure 1—would result in a combinatorial explosion. For instance, our cooperative banks (Bank T and Bank L) executed this threshold-based system during 2019-2020, resulting in 86,770 SAM events. The number of potential threshold combinations from all unique feature values approaches 10^{20}, making complete enumeration computationally infeasible.

To address this intractability, practitioners commonly adopt a Quantile-Based Grid Search (QGS) approach: randomly selecting i quantiles for each feature and exhaustively searching all combinations of these quantile-based thresholds. While more efficient than complete enumeration, QGS still exhibits exponential complexity $O(N^K)$ where K is the number of features and N is the number of quantile candidates. Our experiments reveal that even with moderate quantile granularity (N = 20-40), QGS requires 1,000-17,000 seconds per optimization run, rendering it impractical for production systems requiring frequent threshold updates.

Consequently, this paper develops an efficient gradient-based alternative that significantly outperforms QGS. We innovatively transform the discrete optimization problem into a continuous formulation and solve it using gradient descent. Specifically, we construct a continuous loss function using multiple sigmoid functions that models the discrete trade-off between recall and manual review costs. However, the highly uneven transaction data distribution creates a rugged, multi-modal loss landscape with numerous local optima, causing optimization methods—including sophisticated alternatives like simulated annealing—to become trapped in suboptimal solutions. This fundamental challenge necessitates two critical components in our Smoothing Rank Scale Gradient Descent (SRSGD) approach: (1) rank-scale transformation that addresses extreme gradient variations caused by uneven data density—where dense regions create very steep gradients while sparse regions produce flat gradients—by equalizing threshold spacing to enable effective learning rate tuning, and (2) quantile-based initial threshold selection that provides well-distributed starting points across the feature space. Ablation studies confirm both components are essential: SRSGD achieves comparable or superior Human Resource Savings (HRS) to QGS while reducing computational time by over 4000×, from hours to seconds, and substantially outperforms both standard gradient descent (SOSGD, which lacks rank-scale transformation) and simulated annealing in efficiency and generalization performance.

The remainder of the paper is organized as follows: Section 2 reviews related literature and introduces typical money laundering scenarios utilized in our subsequent experiments. Section 3 presents our proposed gradient descent method for continuized threshold selection. Section 4 compares the experimental results of our proposed method with those obtained using QGS and simulated annealing approaches. Finally, Section 5 concludes the paper and outlines directions for future research.

2 Preliminaries

2.1 Literature Review

Money laundering encompasses a wide range of financial fraud, including online transaction fraud [1], fund transfer fraud [2, 3], instant payment fraud [4], cell phone fraud [5], credit card fraud [6–9], and cryptocurrency scams [10, 11]. While machine learning models have shown promise in fraud detection research [11–14], financial regulators in countries such as China [12], India [15], Hungary [16], South Africa [17], Pakistan [18], and Taiwan [19, 20] mandate threshold-based detection systems that ensure interpretability and regulatory accountability. This regulatory requirement creates a discrete optimization challenge: efficiently adjusting detection thresholds to balance recall, precision, and operational costs while maintaining compliance.

Moreover, the integration of machine learning techniques with threshold-based frameworks is being actively explored to address the inherent limitations of purely rule-based systems. Recent comprehensive reviews [21] highlight that traditional rule-based systems can generate over 98% false positive alerts, leading to substantial operational costs for financial institutions and motivating the search for more efficient detection approaches. Two primary integration strategies have emerged in the literature. First, deep learning approaches [22, 23] have demonstrated the potential to reduce false positives when used to complement or replace traditional AML systems. However, these methods typically require large labeled datasets and sacrifice the interpretability mandated by regulatory frameworks, limiting their practical deployment in regulated environments. Second, hybrid approaches that combine threshold-based filtering with machine learning refinement have shown promise [20, 24, 25]. In this two-stage framework, threshold-based systems first identify potential fraudulent cases as suspicious activity monitoring (SAM) events, which are then analyzed by machine learning models to distinguish genuine fraud instances (designated SAR=1[1]) from false positives (SAR=0). This hybrid approach preserves regulatory interpretability while leveraging machine learning's pattern recognition capabilities.

Khan et al. [18] integrated Bayesian networks with threshold-based systems to enhance the detection of suspicious transaction patterns. This approach assigns Bayesian scores to account users, reflecting the levels of suspicion associated with their transactions. Gupta et al. [15] introduced a two-stage threshold fine-tuning method aimed at enhancing the detection of fraudulent account users across different scenarios. The initial stage employs K-means clustering to maximize the identification of anomalous customer behaviors. The subsequent stage analyzes the relationships among these behaviors across various money laundering scenarios and adjusts cross-scenario thresholds to minimize the recurrence of identified behaviors. This process involves a brute force examination of 240,000 threshold combinations for specific scenarios.

[1]SAR stands for Suspicious Activity Report, denoting cases with confirmed fraudulent activity that are reported to law enforcement.

Badics et al. [16] economically optimized threshold selection by accounting for the costs associated with missing fraudulent cases and the expenses of manual reviews. They employed SIMD parallelization to expedite the brute force search of all threshold combinations, although the effectiveness of this parallelization is limited by various factors [26]. Such approaches become impractical with very large sets of thresholds, as seen in cases involving up to 10^{10} thresholds.

In contrast, our study proposes transforming the threshold optimization problem into a continuous loss function, which is then tackled using a gradient descent method. This approach provides computationally efficient solutions, as demonstrated in our experiments.

2.2 Scenarios for Suspicious Money Laundering or Terrorism

As previously noted, financial regulatory agencies in various countries have enacted anti-money laundering guidelines that compel banks to devise specific rules for monitoring each suspicious transaction situation. For instance, in Taiwan, the Financial Supervisory Commission provides a comprehensive guideline, Type of Suspicious Money Laundering or Terrorism Financing Transactions, which outlines numerous scenarios. Each scenario describes the features to be monitored and their logical interconnections, such as notably high deposits and withdrawals within the same timeframe. However, it does not define precise thresholds for activating these scenarios based on the observed features.

To determine whether an individual j should be flagged as suspicious under a specific money laundering scenario S, we first extract K transactional features defined by scenario S—such as the number of transactions and the total transaction amounts, as illustrated in Figure 1—and denote them as $(x_1^j, x_2^j, \ldots, x_K^j)$. These features are aggregated from all transaction records associated with the individual across a designated monitoring period, in accordance with the Financial Supervisory Commission's guidelines.

The extracted feature vector is then evaluated using a Boolean expression derived from the logic of scenario S, for example:

$$\left(x_1^j \geq t_1^S\right) \wedge \left(x_2^j \geq t_2^S\right) \vee \cdots,$$

where $t_1^S, t_2^S, \ldots$ denote the adjustable thresholds specified for scenario S, which are optimized to strike a balance between recall and the bank's investigation costs.

If the Boolean expression evaluates to true when applied to the individual's aggregated features, the individual is flagged as suspicious (i.e., SAM = 1); otherwise, they are considered non-suspicious (i.e., SAM = 0).

The scenario outlined in the following is an actual example employed by our collaborating Bank T.

The TWN-A11-01 Scenario Used by Bank T

A representative threshold-based scenario employed by Bank T, referred to hereafter as S', is TWN-A11-01. This scenario is designed to flag suspicious activity based on whether the accumulated amounts and counts of credit and debit transactions exceed predefined thresholds within a specific monitoring period. In accordance with this rule, we define four transactional features x_i for each individual j, where x_i^j denotes the value of feature x_i corresponding to individual j. Specifically:

$$
\begin{aligned}
x_1^j &= \text{Total accumulated credit amount over a period of } d \text{ days,} \\
x_2^j &= \text{Total number of credit transactions during the same period,} \\
x_3^j &= \text{Total accumulated debit amount over the same period,} \\
x_4^j &= \text{Total number of debit transactions during the same period.}
\end{aligned}
$$

For each feature x_i, the objective is to determine an optimal threshold t_i that balances the regulatory recall requirement and the cost of manual investigations. The SAM label for an individual j is then derived by evaluating the following Boolean expression:

$$
\left(\bigwedge_{i=1}^{2} (x_i^j \geq t_i) \right) \vee \left(\bigwedge_{i=3}^{4} (x_i^j \geq t_i) \right). \tag{1}
$$

An account is flagged as suspicious (i.e., SAM = 1) if either both credit-related features exceed their respective thresholds or both debit-related features do so.

2.3 Data Preprocessing and a Real World Scenario Example

The raw transaction data from Bank T, spanning from March 4, 2019, to December 31, 2020, encompasses a total of 179,426,210 transactions[2]. Using the scenario TWN-A11-01 as an example, Bank T implemented a set of very conservative, manually set thresholds that identified 446 SAR=1 instances. Under these settings, the threshold-based system flagged a total of 86,770 instances. Adjusting these thresholds could reduce the number of instances flagged as SAM=1, thereby potentially lowering Bank T's review costs; however, this must be balanced against the risk of missing critical SAR=1 instances. Similarly, for Bank L, data from January 1, 2020, to December 31, 2020, recorded 390,260 instances flagged as SAM=1 by the threshold-based system, with 2,943 of these confirmed as SAR=1.

Additionally, based on Bank L's policy, juridical persons are categorized into four groups: general companies, the financial industry, the construction industry, and government-related institutions. These statistics reflect

[2]Note that each individual could execute multiple transactions during this period.

various scenarios of suspicious money laundering or terrorism financing transactions, defined as the scenario set **S**, as outlined in the Type of Suspicious Money Laundering or Terrorism Financing Transactions guideline, including the scenario described in Equation 1.

In our transaction monitoring process, each individual's transactions are tested against a specific scenario from the set **S**, such as TWN-A11-01, to determine whether a SAM=1 alert should be triggered. We evaluate each transaction, along with its related transactions within a predefined interval prior to its occurrence. If the criteria for a SAM=1 alert are met, the transaction is defined as an instance.

The transaction data presented in Table 1 illustrates how instances are identified based on transaction types and values across consecutive days. This table details each day's credit and debit transactions, under the premise that a SAM=1 label is applied when transactions occur at least once on two consecutive days with cumulative amounts exceeding 2.

On the first day, no alerts are triggered due to the absence of preceding data for comparison. On the second day, although the first credit transaction (amount=2) does not meet the cumulative threshold for SAM=1, resulting in SAM=0, the subsequent credit transaction (amount=1) and debit transaction (amount=2) each meet their respective thresholds, triggering two SAM=1 alerts. All transactions on the third day similarly trigger SAM=1 alerts. On the fourth day, the transaction, when considered with the previous day's activity, fails to meet the required cumulative credit amount threshold, and thus no alert is issued. On the fifth day, both transactions meet the alert criteria, and each is marked as SAM=1. The second transaction of this day is further designated as SAR=1, prompting subsequent monitoring by law enforcement.

We depict the distribution of all SAM=1 instances alerted by the default conservative thresholds set by Bank T in Figure 2. Under these conditions, nearly all fraudulent instances are identified as SAM=1, as illustrated in the figure. All fraudulent instances (SAR=1) and normal instances mistakenly flagged by the threshold-based system (SAR=0) are represented by red and blue dots, respectively. The left sub-figure displays each instance's number of credits and the accumulated credit amounts, while the right sub-figure shows the number of debits and accumulated debit amounts. It is evident from the figures that SAR=1 instances do not necessarily involve high numbers of transactions or large accumulated amounts. Furthermore, the substantial overlap between SAR=1 and SAR=0 instances in the feature space indicates that simple threshold-based rules alone cannot effectively distinguish fraudulent from normal transactions. This inefficiency underscores the importance of refining thresholds to effectively balance recall and monitoring costs.

Table 1 Processing Transaction Data to Issue SAM=1. "-" indicates no data available for a threshold-based decision; "SAM=0" denotes that the threshold-based system did not trigger an alert; "SAM=1" signifies that the threshold-based system has triggered an alert for bank review; "SAM=1*" indicates that the instance has been referred to law enforcement authorities; "Law Enforce" denotes that the individual is under monitoring by law enforcement authorities.

Day	1	1	2	2	2	3	4
Credit/Debit	Credit	Debit	Credit	Credit	Debit	Credit	Debit
Amounts	2	1	2	1	2	1	2
threshold-based	-	-	SAM=0	SAM=1	SAM=1	SAM=1	SAM=0

Day	5	5	6
Credit/Debit	Credit	Credit	Credit
Amounts	2	2	2
threshold-based	SAM=1	SAM=1*	Law Enforce

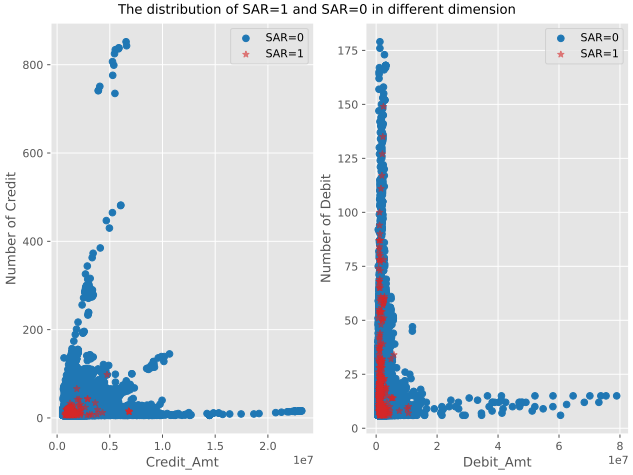


Fig. 2 Distribution of SAM Instances for Scenario TWN-A11-01. Left: credit transactions (number vs. accumulated amounts). Right: debit transactions (number vs. accumulated amounts). Red dots indicate SAR=1 (fraudulent instances); blue dots indicate SAR=0 (false positives).

3 Optimizing Thresholds for threshold-based Detection

This section addresses the threshold optimization problem through three key components. We first formalize the optimization objective by designing a loss function that balances regulatory compliance (minimum recall requirements) with operational efficiency (Human Resource Savings). We then demonstrate why traditional approaches such as Quantile-Based Grid Search (QGS) become computationally intractable as the search space grows. Finally, we introduce our core contribution: the Smoothing Rank Scale Gradient Descent (SRSGD) algorithm, which achieves efficient optimization by transforming the discrete threshold space into a smooth, continuous landscape suitable for gradient-based methods.

3.1 Metrics for Improved Fraud Detection and Resource Management

As outlined in Section 1, our objective is to adhere to the threshold-based guidelines that establish feasible thresholds, aiming to balance the detection of SAR=1 instances with the costs of manual bank reviews. The Financial Supervisory Commission (FSC) mandates that the threshold-based system capture a reasonable ratio of SAR=1 instances. This requirement is quantitatively assessed using the "recall" metric, as defined in Equation (2), which must achieve a minimum threshold, such as 80%, for illustration purposes in this paper.

$$\begin{aligned} Recall &= \frac{\#TP}{\#TP + \#FN}, \\ HRS &= \frac{\#TN + \#FN}{\#TP + \#TN + \#FP + \#FN}, \end{aligned} \tag{2}$$

where TP, TN, FP, and FN denote the standard confusion matrix elements (True/False Positive/Negative), with # indicating the count of instances in each category. Specifically, TP represents SAR=1 instances correctly identified as SAM=1, FP represents SAR=0 instances incorrectly flagged as SAM=1, and so forth. While traditional classification problems emphasize metrics such as Precision and Recall, banks prioritize operational efficiency. This focus corresponds to the Human Resource Savings (HRS) metric defined in Equation (2), which represents the proportion of instances not requiring manual review.

To illustrate how these metrics are calculated, we refer to Figure 1. In the figure, blue nodes represent four fraudulent instances, while black nodes represent eight regular instances. Thresholds are indicated by the solid red points. For this configuration, the recall is calculated as $\frac{2}{4} = 0.5$, indicating that only half of the fraudulent instances are correctly identified by the system. This recall rate does not meet the FSC requirement of 0.8. Consequently, adjustments must be made to lower the thresholds and increase the number of reviews (thus reducing HRS) to enhance the threshold-based system's ability to detect more SAR=1 instances.

To reflect the practical requirement faced by banks—namely, to satisfy the recall rate mandated by the Financial Supervisory Commission (FSC) while minimizing the cost of manual reviews—we design the following custom loss function:

$$L(\text{HRS}, \text{Recall}) = (1 - \text{HRS}) + W \times \text{ReLU}(\texttt{MinRecall} - \text{Recall}), \tag{3}$$

where HRS denotes Human Resource Savings, and W is a large penalty weight associated with failing to meet the minimum recall threshold `MinRecall`, which we set to 0.8 in this study. The ReLU function ensures that no penalty is

incurred when the recall exceeds the threshold, but applies a linear penalty proportional to W when the recall falls short.

Figure 3 visualizes the resulting loss surface over a sample threshold space. The discrete nature of threshold-based detection (illustrated in Figure 1) creates step-wise transitions in SAM labels, leading to plateau-like flat regions where small threshold changes do not alter label assignments. Combined with the highly uneven data distribution (shown in Figure 2), this produces a rugged, multi-modal loss landscape with numerous local optima. The steep cliffs in the surface correspond to sudden failures to meet the minimum recall constraint, where the penalty term W sharply increases the loss. This complex landscape makes gradient-based optimization challenging without appropriate transformations.

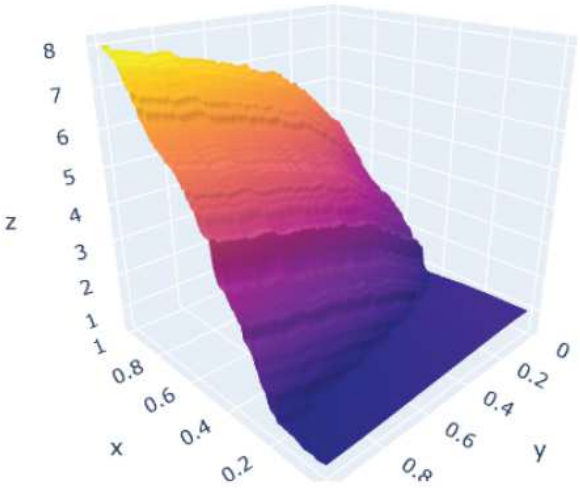


Fig. 3 A Sample Loss Function Surface. The x- and y-axes represent two decision thresholds under evaluation, while the z-axis indicates the corresponding loss value. The surface exhibits a rugged, multi-modal structure arising from discrete threshold transitions (cf. Figure 1) and uneven data distribution (cf. Figure 2), creating numerous local optima that challenge standard optimization methods.

This loss function is compatible with QGS and heuristic-based methods such as simulated annealing, which are discussed in later sections. However, due to its non-differentiable nature, it is not directly applicable to gradient-based optimization algorithms. To address this limitation, the following sections introduce a differentiable surrogate loss function that preserves the original design's regulatory intent while enabling efficient optimization via gradient descent.

3.2 Quantile-Based Grid Search (QGS) for Threshold Combinations

In the following subsections, we use t_i to denote the threshold for feature i as a continuous optimization variable. While these thresholds are initialized from the discrete candidate set $T_i = [t_{i,1}, t_{i,2}, \ldots, t_{i,N'}]$ defined in Section 3.3.1, the rank-scale transformation and gradient descent allow them to take continuous values during optimization. After convergence, the optimal thresholds can be mapped back to their original feature space for practical deployment. Exhaustively enumerating all possible threshold configurations across all unique feature values would result in computational intractability. For Bank T's TWN-A11-01 scenario with four features (credit amount, credit count, debit amount, debit count) across 86,770 transactions, the number of potential threshold combinations approaches 10^{20} when considering all unique data values. Even after excluding duplicate values, the search space remains astronomically large with approximately 10^{13} combinations, making complete brute-force enumeration computationally infeasible.

To address this intractability, we adopt a Quantile-Based Grid Search (QGS) approach, which practitioners commonly employ to make threshold optimization tractable. QGS systematically selects i quantiles for each feature to serve as threshold candidates, then exhaustively evaluates all combinations of these quantile-based thresholds. For instance, with $i = 10$ quantiles across four features, QGS evaluates $10^4 = 10,000$ threshold combinations. While dramatically reducing the search space compared to complete enumeration, QGS still exhibits exponential complexity $O(i^K)$, where K is the number of features and i is the number of quantiles per feature.

Table 2 demonstrates the computational demands of QGS across different quantile granularities for the TWN-A11-01 scenario. When the number of quantiles increases from 10 to 40, computational time increases dramatically from 66 seconds to 17,362 seconds (approximately 4.8 hours), while Human Resource Savings (HRS) remains essentially unchanged at approximately 0.20 and all configurations achieve similar recall rates around 0.80. This pattern reveals a critical limitation: increasing quantile granularity dramatically increases computational cost without meaningful improvement in solution quality. Extrapolating to $i = 1000$ quantiles yields an estimated computation time of 6.78×10^9 seconds (around 200 years), clearly demonstrating QGS's impracticality for fine-grained threshold optimization. Moreover, the selected threshold settings vary significantly across different quantile granularities (as shown in the Threshold rows of Table 2), leading to poor financial interpretability and instability.

Given QGS's computational limitations, some prevalent metaheuristic algorithms, such as simulated annealing [27, 28], have been proposed for threshold optimization in fraud detection. Our subsequent experiments will evaluate these approaches alongside QGS to demonstrate the superiority of our gradient-based method.

Table 2 Estimated time required for QGS computations across different threshold settings, utilizing a computer system equipped with an Intel(R) Core(TM) i7-13700 CPU, 64 GB DRAM, and an RTX3070 graphics card. The table details the computational time, recall rate, and HRS for various numbers of thresholds for each feature of the TWN-A11-01 scenario, illustrating the exponential increase in computational time as the number of thresholds increases.

Number of thresholds		10	20	30	40	1000 (est.)
Times (sec.)		66.14100	1088.44561	5516.77918	17361.85421	6.78×10^9
Recall		0.8000195	0.8008570	0.8002568	0.8001591	
HRS		0.1999806	0.1991430	0.1997433	0.199841	
Threshold	Credit amount	1320920	881735.4509	1084438	1124048	
	Credit count	0.5	0.5	0.5	11.43609	
	Debit amount	290191.3	467395.3536	860889.1	50	
	Debit count	0.5	8.264854	5.426185	0.5	

3.3 Smoothing Rank Scale Gradient Descent(SRSGD)

The loss function defined in Equation 3, which combines HRS and Recall (Equation 2), exhibits a discrete step-wise structure. Table 2 indicates that the number of steps is substantial, making it impractical to evaluate using brute force or simple approximation methods such as QGS. Therefore, this paper proposes transforming the discrete step loss function into a continuous loss function, utilizing gradient descent for solution finding. However, as Figure 2 illustrates, the distribution is uneven, predominantly clustering in the bottom left corner, complicating the design of an appropriate learning rate to ensure convergence of the loss function. To address this challenge, we introduce a rank-scale transformation that normalizes the uneven spacing between threshold candidates, creating a more uniform loss landscape suitable for gradient descent. After optimization in rank space, the optimal thresholds are mapped back to their original feature values.

3.3.1 Possible Discrete Thresholds

In threshold-based systems, thresholds for individual features are typically selected from a predefined set of discretized candidate values, such as $a, b, c, \ldots$, as illustrated in Figure 1. Suppose we have N individuals (i.e., N instances), and we focus on a specific feature x_k, where $k \in \{1, 2, \ldots, K\}$. To reduce redundancy and simplify threshold selection, we merge instances that share identical values of x_k. After deduplication, we obtain a sorted list of unique values, from which we construct an ordered set of candidate thresholds $T_k = [t_{k,1}, t_{k,2}, \ldots, t_{k,N'}]$, where $N' \leq N - 1$. These thresholds partition the feature space into intervals, each defined between two adjacent values and containing instances with the same x_k value. As shown in Figure 1, for the feature "number of transaction times" on the x-axis, five candidate thresholds—namely a, b, c, d, e—are defined. The interval between thresholds c and d contains two instances with identical values $x_k = 15$, illustrating how similar instances are grouped under a single interval. A threshold such as c can

be numerically set as the midpoint between two adjacent observed values; for instance, if the observed values are 10 and 15, then threshold c may be defined as 12.5. This structured discretization facilitates precise segmentation of the feature space for threshold-based detection and provides a solid foundation for subsequent threshold optimization.

However, the loss function L, which jointly considers Human Resource Savings (HRS) and Recall as defined in Equation (3), exhibits a stepwise structure, as illustrated in Fig. 3. This non-smooth landscape renders conventional gradient-based optimization techniques inapplicable. To overcome this limitation, we propose a tailored discrete optimization procedure described as follows.

3.3.2 Continuous Approximation Using Sigmoid Functions

To enable gradient-based optimization, the inherently step-wise indicator functions can be smoothly approximated using sigmoid activations. For instance, the indicator function that determines whether the i-th feature value x_i^j of individual j exceeds threshold t_i can be approximated by:

$$\sigma_{i,j}(t_i) = \frac{1}{1 + e^{-M(x_i^j - t_i)}},$$

where M controls the steepness of the sigmoid curve. When $x_i^j \gg t_i$, $\sigma_{i,j}(t_i)$ approaches 1; conversely, when $x_i^j \ll t_i$, it approaches 0.

Composite decision rules involving multiple features, such as those in Equation (1), can be represented using logical *AND* and *OR* operations. These logic operations can be approximated as follows:

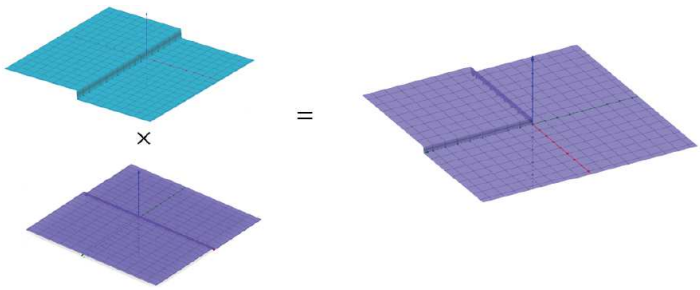


Fig. 4 The logic operator *AND*.

The AND logic for two features i_1 and i_2 can be approximated by the product of their sigmoid activations, as illustrated in Figure 4:

$$\sigma_{i_1,j}(t_{i_1}) \times \sigma_{i_2,j}(t_{i_2}).$$

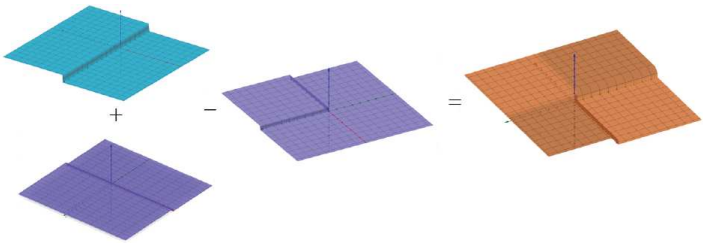


Fig. 5 The logic operator *OR*.

Similarly, the OR logic can be approximated by the sum of their sigmoid activations minus their product, as illustrated in Figure 5:

$$\sigma_{i_1,j}(t_{i_1}) + \sigma_{i_2,j}(t_{i_2}) - \sigma_{i_1,j}(t_{i_1}) \times \sigma_{i_2,j}(t_{i_2}).$$

Using the "AND" and "OR" logic derived above, whether the individual j would be flagged as SAM = 1 by TWN-A11-01 in Equation (1) can be expressed as

$$\begin{aligned} SAM(j, t_1, t_2, t_3, t_4) = &\left(\prod_{i=1}^{2} \sigma_{i,j}(t_i)\right) + \left(\prod_{i=3}^{4} \sigma_{i,j}(t_i)\right) \\ &- \left(\prod_{i=1}^{2} \sigma_{i,j}(t_i) \times \prod_{i=3}^{4} \sigma_{i,j}(t_i)\right), \end{aligned} \tag{4}$$

where $\sigma_{i,j}(t_i)$ denotes the sigmoid function applied to the i-th feature of individual j with threshold t_i.

Based on this formulation, the quantities in Equation (2) can be computed as follows. The number of true positives is given by:

$$\#TP = \sum_{j=1}^{N} SAM(j, t_1, t_2, t_3, t_4) \times SAR(j), \tag{5}$$

where $SAR(j) = 1$ if individual j is a fraudulent instance, and $SAR(j) = 0$ otherwise.

The total number of individuals flagged as SAM = 1 (i.e., $\#TP + \#FP$), which indicates the instances subjected to manual review by the bank's compliance team for further investigation, is given by:

$$\#TP + \#FP = \sum_{j=1}^{N} SAM(j, t_1, t_2, t_3, t_4). \tag{6}$$

These expressions allow Recall and HRS in Equation (2) to be evaluated in a differentiable and optimization-friendly manner. As shown in Equation (3),

the loss function $L(\text{HRS}, \text{Recall})$ incorporates both the human resource savings and the regulatory recall constraint. Notably, both HRS and recall are differentiable with respect to the threshold parameters t_1, t_2, t_3, t_4 through the continuous approximation of $SAM(j, \cdot)$ defined in Equation (4). Therefore, the loss function can be rewritten as a function of the threshold vector $\mathbf{t} = (t_1, t_2, t_3, t_4)$, enabling the use of gradient-based optimization methods such as gradient descent. By minimizing the loss function $L(\mathbf{t})$, we can efficiently identify an optimal set of thresholds for the TWN-A11-01 scenario that balances regulatory compliance with operational efficiency.

3.3.3 Rank-Scale Transformation

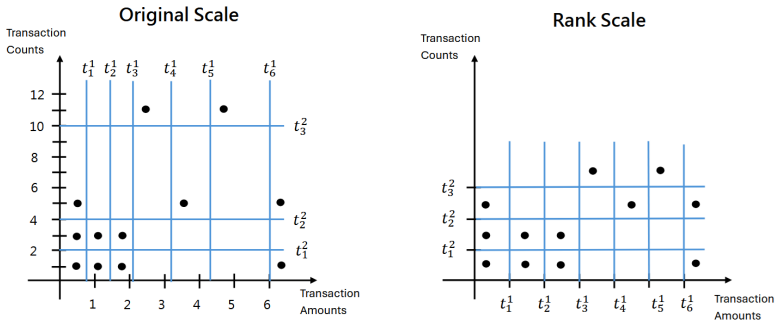


Fig. 6 Illustration of threshold spacing under original scale (left) and rank scale (right). Black dots represent account-level transaction data; blue lines represent threshold candidates. The rank scale equalizes the distance between thresholds, aiding optimization.

As illustrated in the left panel of Figure 6, the distribution of transaction data is highly uneven, with most accounts (depicted as black dots) exhibiting low transaction frequencies and amounts. This concentration in the lower-left corner results in very small intervals between adjacent threshold candidates in this region, while intervals in the sparse regions become disproportionately large (blue lines). Such irregular spacing causes dramatic gradient variations: in dense regions with closely-spaced thresholds, the loss function exhibits extremely steep gradients, while sparse regions produce very flat gradients. These extreme differences—often spanning multiple orders of magnitude—make it impossible to select an appropriate learning rate: a rate suitable for dense regions causes overshooting in sparse regions, while a rate appropriate for sparse regions leads to negligible progress in dense regions.

To overcome this gradient variation challenge, we transform the original feature space into a rank-based scale, as shown in the right panel of Figure 6. In the rank scale representation, the distances between adjacent threshold candidates are normalized to be equal. This equalization fundamentally resolves

the learning rate dilemma: by ensuring uniform spacing, gradient magnitudes become comparable across all regions, enabling a single learning rate to work effectively throughout the optimization process. This stabilizes gradient descent convergence and improves optimization efficiency.

4 Experimental Results and Analysis

To validate the effectiveness and generality of the proposed SRSGD method, we conduct comprehensive experiments across four Anti-Money Laundering (AML) scenarios from two major Taiwanese financial institutions. These scenarios represent diverse detection logics that differ significantly in feature complexity, temporal aggregation, and Boolean structure:

- **TWN-A11-01** (Bank T): Monitors cash deposits and withdrawals using 4 features (credit/debit amounts and transaction counts). This scenario, detailed in Section 2.2, forms the basis for our detailed analysis in Subsections 4.1 and 4.2.
- **TWN-A1A-01** (Bank L): Analyzes credit and debit patterns with an additional ratio constraint, requiring threshold optimization across 5 features as detailed in Appendix A. This includes monitoring the debit-to-credit ratio, which must fall within a specified range simultaneously with individual amount and count thresholds.
- **TWN-AB1-01** (Bank L): Focuses on cross-border remittances using only 2 features (accumulated amount and transaction count) as detailed in Appendix A. This represents the simplest detection logic among the four scenarios, employing a straightforward conjunction of two threshold conditions.
- **TWN-A14-01** (Bank L): Examines debit patterns across multiple time windows, requiring 4 features with complex temporal aggregation as detailed in Appendix A. The scenario combines monthly average amounts with daily accumulations, representing a more sophisticated temporal monitoring strategy.

These four scenarios provide a comprehensive testbed for evaluating SRSGD's robustness across varying levels of complexity:

- **Feature dimensionality**: Ranging from 2 to 5 features
- **Logical structure**: From simple AND operations (TWN-AB1-01) to complex ratio constraints (TWN-A1A-01)
- **Temporal granularity**: Single time windows vs. multi-period aggregations
- **Transaction types**: Domestic transfers (TWN-A11-01, TWN-A1A-01, TWN-A14-01) vs. cross-border remittances (TWN-AB1-01)

In the following subsections, we first conduct detailed analysis on the TWN-A11-01 scenario, examining initialization strategies (Subsection 4.1) and comparing SRSGD against alternative optimization methods (Subsection 4.2). We then present comprehensive validation results across all four scenarios

in Subsection 4.3, demonstrating the method's consistent superiority and generalizability.

4.1 Decide the Initial Threshold

All thresholds must comply with FSC requirements. To ensure that the optimized thresholds remain feasible after model processing, it is critical to choose an appropriate initial threshold. A well-designed initialization not only accelerates convergence but also reduces the risk of becoming trapped in local minima, especially under a highly non-convex loss surface as in Figure 3. Exhaustively testing all threshold combinations is computationally prohibitive. As shown in Section 3.1, the distance from the origin is positively correlated with Human Resource Savings (HRS) but negatively correlated with recall. Motivated by this observation, we calculate the distances of all SAR=1 cases from the origin and select as the initial threshold the nearest case at the 80th percentile of these distances. This initialization strategy guides the optimization toward thresholds that satisfy FSC requirements while approximating the minimum loss defined in Equation (3). The empirical effects of different initialization strategies are summarized in Table 3.

Table 3 compares three initialization strategies within our proposed rank-scale gradient descent framework: (i) setting the initial thresholds at the origin, (ii) our proposed SAR=1 distance-based initialization, and (iii) random initialization averaged over ten trials. The results reveal several key findings.

First, initializing at the origin achieves high recall (0.91 on training and 0.94 on testing sets) but almost no HRS (≈ 0.1), indicating limited cost savings. Second, random initialization yields unstable convergence, with extremely large losses and unsatisfactory recall (0.38 on training and 0.28 on testing), making it infeasible under FSC requirements. This outcome can be explained by two factors: (a) the highly imbalanced nature of AML/CTF data, where SAR=1 instances are sparse, causes random thresholds to frequently misclassify most cases as non-suspicious, leading to very low recall; and (b) the non-convex, plateau-like loss surface makes it difficult for gradient updates to escape from infeasible regions once the initialization falls into them. Consequently, random initialization not only violates the FSC recall requirement but also converges to unstable and high-loss solutions. In contrast, our proposed initialization consistently balances recall (0.80 on training and 0.85 on testing) with substantially higher HRS (0.41–0.43), while minimizing the loss.

Importantly, thresholds selected from the training set generalize well to the testing set, indicating that our approach is not significantly affected by dataset shift. Furthermore, the computation time of our method (3.99 seconds) is the shortest among all strategies, underscoring the efficiency of rank-scale gradient descent optimization. Overall, these results demonstrate that an appropriately designed initialization is essential for ensuring FSC compliance, accelerating convergence, and enhancing cost-effectiveness.

Table 3 Performance of different initialization strategies. Zeros = thresholds initialized at the origin (high recall but negligible HRS); Proposed = initialization from the nearest SAR=1 case at the 80th percentile distance (balanced recall and HRS, fast convergence); Random = thresholds initialized randomly and averaged over ten trials (unstable, low recall). Recall is the proportion of SAR=1 cases correctly identified, and HRS (Human Resource Savings) is the proportion of transactions exempted from manual review. The proposed distance-based initialization provides the best trade-off between recall and HRS while ensuring FSC compliance.

	zeros	initial threshold	random(10 times average)
Train Loss	0.922557	0.585289	125.658646
Train Recall	0.907258	0.802419	0.383065
Train HRS	0.077443	0.414711	0.421997
Test Loss	0.899868	0.572710	155.716278
Test Recall	0.939394	0.848485	0.282828
Test HRS	0.100132	0.427290	0.435248
Times	4.408172	3.990000	4.465917

4.2 Comparison of Optimization Methods

Table 4 compares four optimization strategies. The QGS serves as a benchmark but requires over 17,000 seconds, making it computationally infeasible for real-world AML applications.

We refer to the method introduced in Section 3.3.2 as Smoothing Gradient Descent (SGD). Two variants are considered: the Smoothing Original Scale Gradient Descent (SOSGD) and the Smoothing Rank Scale Gradient Descent (SRSGD), as illustrated in Figure 6.

The SOSGD variant exhibits unstable convergence due to the highly uneven distribution of transaction data, as shown in Figure 3. In dense regions of the feature space, adjacent thresholds are extremely close, whereas in sparse regions the spacing becomes disproportionately large. This irregular spacing causes dramatic gradient fluctuations across the feature space: gradients may vary by several orders of magnitude between dense and sparse regions, making it impossible to tune a single learning rate that works effectively across all regions. This gradient instability leads to inefficient updates and convergence to suboptimal solutions.

The Simulated Annealing method demonstrates the fastest runtime and achieves the highest HRS on the training set. However, it fails to satisfy the FSC recall requirement on the test set (recall = 0.419), indicating overfitting and poor generalization.

In contrast, the SRSGD variant applies a rank-scale transformation that equalizes threshold spacing across the feature space. This normalization stabilizes gradient magnitudes, eliminating the extreme variations that prevent effective learning rate tuning in the original scale, thereby enabling reliable gradient-based optimization, and achieves the best balance between recall and HRS with the lowest computation time. These results confirm that rank-scale smoothing is essential for practical and robust AML threshold optimization.

Table 4 Comparison of optimization methods. This table reports the performance of four optimization strategies for AML threshold selection: the QGS, two variants of the proposed Smoothing Gradient Descent approach (Smoothing Original Scale Gradient Descent, SOSGD, and Smoothing Rank Scale Gradient Descent, SRSGD), and the Simulated Annealing method. All methods are initialized using the threshold selection strategy described in Section 4.1.

	QGS	SOSGD	SRSGD	Simulated Annealing
Train Loss	0.800058	0.633945	0.586219	0.556660
Train Recall	0.842697	0.830645	0.802419	0.803750
Train HRS	0.199942	0.366055	0.413781	0.443340
Test Loss	0.813314	0.640775	0.603689	3.702077
Test Recall	0.919192	0.848485	0.848485	0.419135
Test HRS	0.186686	0.359225	0.396311	0.832323
Train Time(sec.)	17361.854	82.262527	3.996729	1.870929

Figures 7–9 provide a detailed comparison of the optimization dynamics across different methods on both the training and test sets.

Figure 7 illustrates that the discrete gradient descent method quickly becomes trapped on flat plateaus of the loss surface. This behavior stems from the inherently stepwise structure of the discrete objective function, where small perturbations of the thresholds do not alter SAM assignments and therefore produce zero gradients. As a result, even when the search region is expanded, the optimization remains confined to the same loss level and fails to achieve further improvement.

In contrast, Figure 8 demonstrates that the Continuous Sigmoid Transform (CST) exhibits substantially greater robustness with respect to initialization. Even when starting from suboptimal threshold configurations, CST consistently adjusts the thresholds to satisfy the FSC-mandated recall constraint. This behavior highlights CST's ability to provide meaningful gradients throughout the optimization process, enabling recovery from unfavorable starting points and preventing premature stagnation.

Figure 9 further compares the evolution of Human Resource Savings (HRS) across methods. While discrete gradient descent shows limited improvement due to its inability to escape loss plateaus, CST achieves a steadily increasing HRS trajectory on both training and test sets. This result confirms that CST not only meets regulatory recall requirements but also more effectively reduces manual review costs, thereby offering a superior trade-off between compliance and operational efficiency. This contrast highlights the fundamental limitation of gradient-based methods when applied directly to discrete threshold selection, and motivates the necessity of the proposed continuous relaxation.

By applying CST, the originally discontinuous and stepwise threshold selection problem is transformed into a smooth optimization landscape. This reformulation enables the use of gradient-based optimization methods, such as gradient descent, to efficiently update thresholds during training. Figures 10– 12 present the resulting optimization trajectories in terms of Loss, Recall, and Human Resource Savings (HRS), respectively, demonstrating how

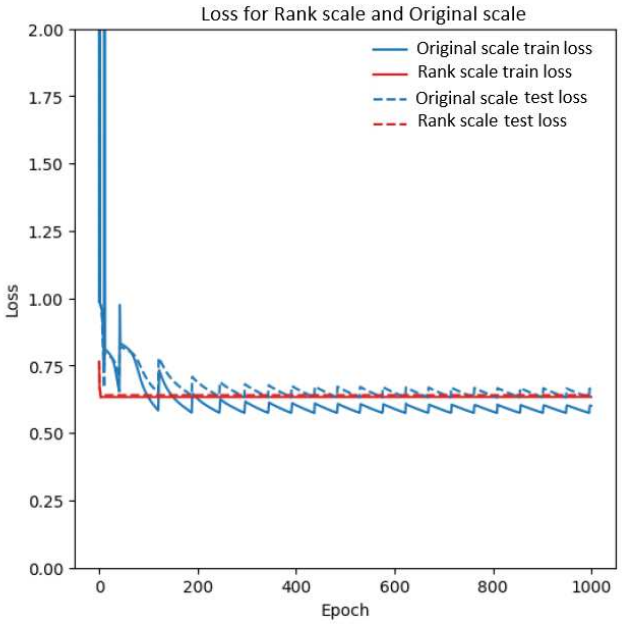


Fig. 7 Loss process for Rank scale and Original scale in training set and test set

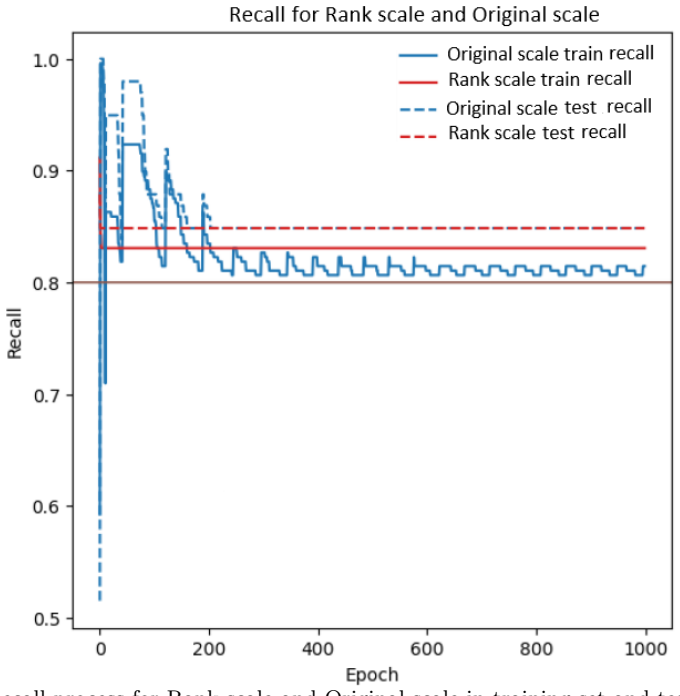


Fig. 8 Recall process for Rank scale and Original scale in training set and test set

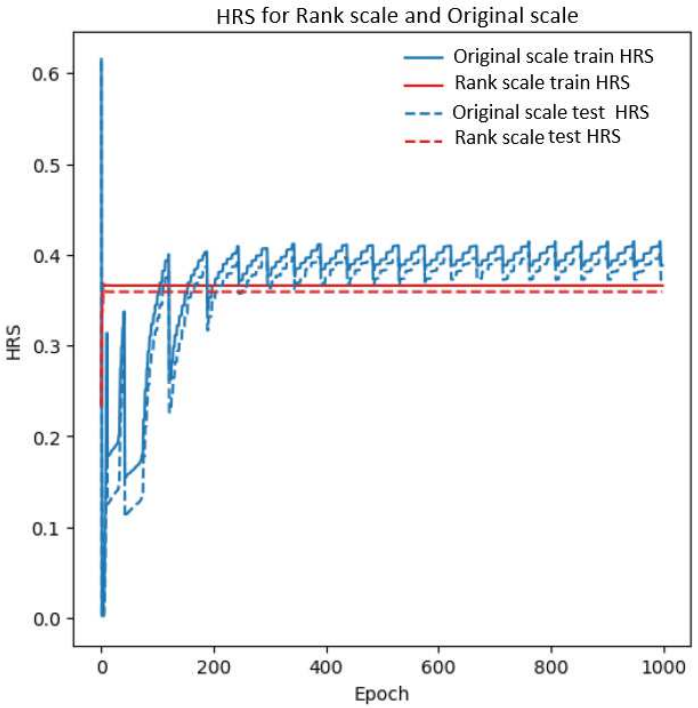


Fig. 9 HRS process for Rank scale and Original scale in training set and test set

the proposed approach improves convergence behavior and overall system performance.

4.3 Multi-Scenario Validation

Having demonstrated SRSGD's effectiveness on TWN-A11-01, we now validate its performance across three additional Bank L scenarios to assess generality and robustness. This cross-scenario evaluation establishes SRSGD as a practical solution for diverse AML detection rules rather than a scenario-specific method.

4.3.1 Experimental Protocol

All experiments follow the same rigorous protocol to ensure fair comparison. For initialization, we employ the distance-based approach detailed in Section 4.1, which selects the 80th percentile nearest SAR=1 case from the origin. Each scenario evaluates four methods: (1) QGS, (2) SOSGD, (3) SRSGD, and (4) Simulated Annealing.

The experimental design employs chronological train-test splitting to reflect realistic deployment conditions. All gradient-based methods use identical hyperparameter settings without scenario-specific tuning. Specifically, we set the temperature parameter $M = 5$ for sigmoid transformation, which controls

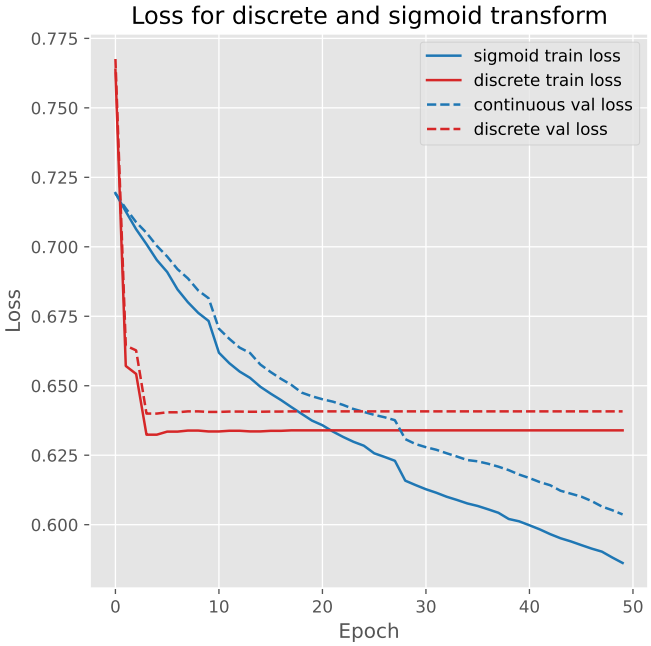


Fig. 10 Loss of the discrete gradient and CST

the steepness of the continuous loss approximation, and define convergence as no threshold updates for ten consecutive iterations. Performance is assessed using three metrics: Loss (Equation 3), Recall, and HRS.

The only modifications across scenarios are those intrinsic to each FSC-defined detection rule (Appendix A). Importantly, no algorithmic modifications or hyperparameter adjustments are made, allowing us to evaluate SRSGD's true generalizability across diverse AML detection scenarios.

4.3.2 Results on Bank L Scenarios

Table 5 reveals consistent patterns across all three Bank L scenarios. SRSGD achieves the best test HRS in each scenario (0.386-0.391) while maintaining regulatory-compliant recall (0.831-0.851), demonstrating robust performance across diverse detection rules with 2-5 features. The computational efficiency advantage remains striking, with training times of 3.9-4.1 seconds representing approximately 4000× speedup over QGS's 16,000-17,000 seconds. Notably, Simulated Annealing exhibits catastrophic test recall failure (0.412-0.415) across all scenarios, confirming the systematic overfitting pattern observed in TWN-A11-01.

To facilitate cross-scenario comparison, Table 6 summarizes SRSGD's performance across all four AML scenarios:

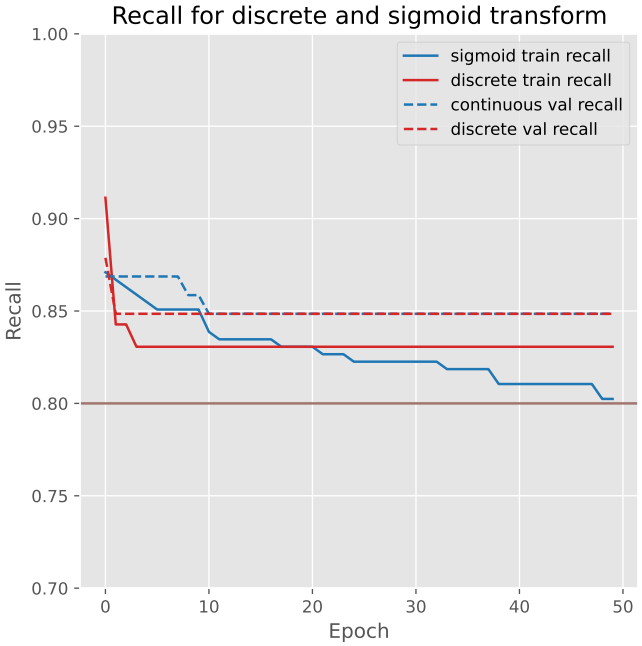


Fig. 11 Recall of the discrete gradient and CST

Key findings: (1) SRSGD achieves best Test HRS in all scenarios (0.380-0.391) while maintaining regulatory compliance (recall ¿0.8), (2) Training time consistently 3.9-4.1 seconds represents 4000× speedup over QGS (16,135-17,361 seconds), and (3) This ablation study, comparing SRSGD with SOSGD (which lacks rank-scale transformation), isolates the rank-scale transformation's critical impact: +6.0-10.3% HRS improvement,[3] 21-22× faster training (82-88s vs. 3.9-4.1s), with similar recall performance.

These consistent advantages across diverse scenarios (2-5 features, different Boolean structures, varying temporal aggregations) demonstrate that rank-scale transformation addresses a fundamental optimization challenge rather than providing scenario-specific benefits.

In stark contrast, Simulated Annealing exhibits systematic failure across all scenarios, with catastrophic performance on test sets. Test recall consistently falls around 0.41-0.42 (requirement: 0.8): TWN-A11-01: 0.419; TWN-A1A-01: 0.415; TWN-AB1-01: 0.412; TWN-A14-01: 0.413.

This systematic pattern suggests severe overfitting. Key indicators: (1) massive train-test divergence (Train Recall 0.80-0.81 vs. Test Recall 0.41-0.42), (2) explosive Test Loss values (3.52-3.70 vs. 0.58-0.80 for other methods), and

[3]Test HRS improvement calculated as relative percentage: (SRSGD HRS − SOSGD HRS)/SOSGD HRS × 100%. For example, TWN-A11-01: $(0.396 - 0.359)/0.359 = 10.3\%$; TWN-AB1-01: $(0.391 - 0.369)/0.369 = 6.0\%$. See Table 4 and Table 5 for complete data.

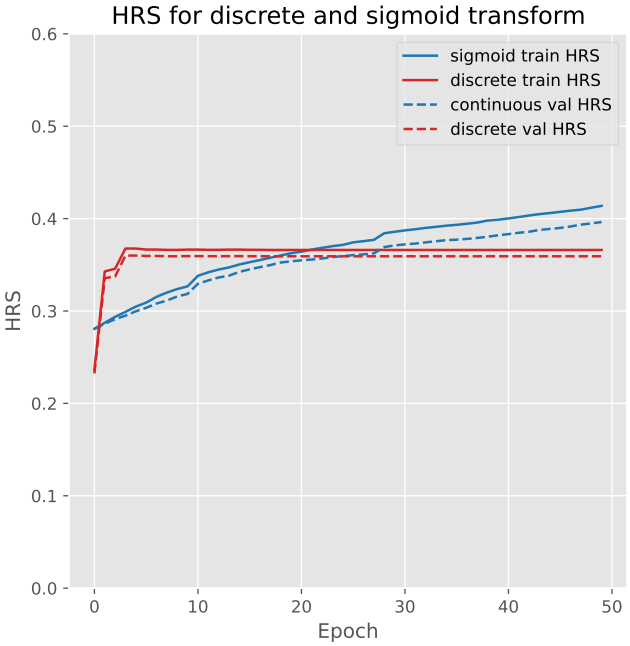


Fig. 12 HRS of the discrete gradient and CST

(3) artificially inflated Test HRS (0.82-0.83) achieved by rejecting 60% of true SAR=1 cases. Simulated Annealing exploits training data by finding thresholds that barely satisfy recall constraints locally but fail to capture genuine suspicious patterns for generalization.

QGS achieves the highest test recall (0.912-0.925) across all scenarios, suggesting that globally optimal recall-maximizing thresholds exist within the search grid. However, this comes at substantial costs: (1) 16,000-17,000 seconds (4.5-4.7 hours) per run makes QGS impractical for production requiring frequent updates—the 4000× slowdown vs. SRSGD would delay deployment and prevent A/B testing, (2) Despite maximizing recall, QGS achieves *lowest HRS* (0.178-0.192), flagging 80-82% of all transactions for manual review with minimal resource savings, and (3) Very conservative thresholds satisfy regulatory requirements but provide little practical efficiency gain.

SRSGD demonstrates remarkable robustness across diverse scenario properties. In terms of feature dimensionality, the method shows strong performance in both TWN-AB1-01 (2 features, HRS: 0.391) and TWN-A1A-01 (5 features, HRS: 0.386) with similar training times (4.08s, 3.94s), showing effective scaling without curse of dimensionality. Regarding logical complexity, TWN-A1A-01's ratio constraint requires optimizing six threshold parameters simultaneously, yet SRSGD converges reliably without modified hyperparameters. For temporal aggregation, TWN-A14-01 combines features over different

Table 5 Performance comparison across three Bank L AML scenarios. Results for TWN-A1A-01 (5 features with ratio constraint), TWN-AB1-01 (2 features for cross-border remittances), and TWN-A14-01 (4 features with multi-window temporal aggregation). SRSGD achieves best HRS and fastest training while maintaining regulatory-compliant recall; Simulated Annealing systematically fails on test recall.

TWN-A1A-01	**QGS**	**SOSGD**	**SRSGD**	**Simulated Annealing**
Train Loss	0.814478	0.621890	0.600598	0.556593
Train Recall	0.835355	0.839283	0.810563	0.812588
Train HRS	0.193390	0.391187	0.417289	0.441486
Test Loss	0.802938	0.657535	0.614366	3.585608
Test Recall	0.911841	0.847832	0.850816	0.414647
Test HRS	0.192138	0.355235	0.386492	0.817746
Train Time(sec.)	16135.02	84.39929	3.944729	1.949487
TWN-AB1-01	**QGS**	**SOSGD**	**SRSGD**	**Simulated Annealing**
Train Loss	0.762995	0.609956	0.588791	0.582993
Train Recall	0.849927	0.836671	0.807232	0.813073
Train HRS	0.193635	0.381296	0.414728	0.435242
Test Loss	0.777113	0.625764	0.575540	3.515370
Test Recall	0.923758	0.843413	0.831837	0.412025
Test HRS	0.191441	0.369491	0.390932	0.826112
Train Time(sec.)	17352.86	85.31472	4.076159	1.765114
TWN-A14-01	**QGS**	**SOSGD**	**SRSGD**	**Simulated Annealing**
Train Loss	0.832558	0.646190	0.593633	0.557404
Train Recall	0.830989	0.836681	0.804698	0.795854
Train HRS	0.281554	0.377058	0.423313	0.459640
Test Loss	0.801115	0.645785	0.618116	3.703123
Test Recall	0.924974	0.856402	0.850427	0.412753
Test HRS	0.178174	0.350753	0.380163	0.825817
Train Time(sec.)	16765.21	87.59342	3.898450	1.889259

Table 6 SRSGD performance summary across four AML scenarios. SRSGD achieves best Test HRS in all scenarios while maintaining regulatory-compliant recall (¿0.8) and providing 4000-4300× computational speedup compared to QGS.

Scenario	**Test Recall**	**Test HRS**	**Train Time (s)**	**Speedup vs. QGS**
TWN-A11-01	0.848	**0.396**	4.00	4340×
TWN-A1A-01	0.851	**0.386**	3.94	4093×
TWN-AB1-01	0.832	**0.391**	4.08	4256×
TWN-A14-01	0.850	**0.380**	3.90	4299×

time scales; rank transformation handles this seamlessly, achieving substantial HRS improvement over SOSGD (+8.4%).

4.3.3 Production Deployment Implications

The multi-scenario validation demonstrates several critical practical advantages for production deployment. First, SRSGD enables plug-and-play deployment without scenario-specific tuning. Compliance officers can apply SRSGD

to any FSC-defined rule without data science expertise, significantly lowering the barrier to adoption. Second, the consistent results across Bank T and Bank L demonstrate that SRSGD's advantages are independent of institution-specific characteristics, which is crucial for industry-wide adoption.

The computational efficiency of SRSGD provides additional practical benefits. The 4-second training time enables rapid iteration, allowing sensitivity analysis and expert knowledge incorporation within minutes rather than hours. This rapid feedback loop facilitates iterative refinement and stakeholder collaboration during threshold calibration. Moreover, the stable method ranking (SRSGD ¿ SOSGD ¿ QGS ¿ Simulated Annealing) remains consistent across all scenarios, suggesting fundamental algorithmic advantages rather than scenario-specific luck.

Finally, Simulated Annealing's systematic failure—despite appearing sophisticated—highlights the critical value of comprehensive multi-scenario validation for any proposed AML threshold optimization method. Methods that perform well on single scenarios may exhibit catastrophic failures when deployed more broadly, underscoring the importance of rigorous cross-scenario testing.

In summary, SRSGD's consistent superiority across four diverse AML scenarios (two institutions, 2-5 features, varying Boolean logic and temporal aggregation) provides compelling evidence for practical utility and generalizability. The combination of best HRS, regulatory compliance (recall ¿0.8), and 4000× computational speedup positions SRSGD as a robust production solution.

5 Conclusion

Regulatory requirements mandate threshold-based anti-money laundering systems for commercial banks, creating a critical optimization challenge: how to set detection thresholds that balance regulatory compliance with operational efficiency. This challenge is particularly acute given the highly uneven distribution of financial transaction data, where most legitimate transactions cluster in dense regions while suspicious activities scatter across the feature space.

We introduce Smoothing Rank Scale Gradient Descent (SRSGD), which addresses this challenge through a novel rank-scale transformation combined with sigmoid-based continuous approximation. The key innovation lies in the rank-scale transformation, which addresses a fundamental challenge in optimizing over unevenly distributed data: extreme gradient variations (spanning multiple orders of magnitude) that make learning rate selection intractable. By transforming thresholds to rank space, where spacing is uniform, gradient magnitudes become stable across the feature space, enabling effective gradient descent with a single learning rate. Unlike conventional gradient descent that adjusts continuous parameters via learning rates, SRSGD performs discrete search among candidate thresholds, selecting at each iteration the threshold yielding steepest loss reduction. This design preserves the discrete nature of

threshold-based systems while enabling efficient optimization through smooth gradient flow.

Comprehensive validation across four diverse AML scenarios from two Taiwanese banks establishes SRSGD's consistent superiority. The method achieves 4000-fold computational speedup over exhaustive grid search, reducing optimization time from 4.5 hours to 4 seconds while nearly doubling operational efficiency (HRS improvement from 0.18-0.19 to 0.38-0.42) and maintaining regulatory-compliant recall above 0.8. Critically, SRSGD demonstrates robust generalization across scenarios varying in feature dimensionality (2-5 features), logical complexity (simple conjunctions to ratio constraints), and temporal aggregation patterns, requiring no scenario-specific hyperparameter tuning. Ablation studies confirm that both rank-scale transformation and quantile-based initialization are essential: SRSGD outperforms standard gradient descent (SOSGD, which lacks rank-scale transformation) by 6-10% in HRS while training 20-fold faster, demonstrating the critical value of transforming the rugged loss landscape. Meanwhile, sophisticated alternatives like Simulated Annealing become trapped in local optima created by the uneven data distribution, exhibiting systematic overfitting with high training recall but catastrophic test failures (recall around 0.41 versus the required 0.8), confirming that the multi-modal loss landscape necessitates our approach's dual innovations.

This framework has been successfully integrated into production anti-money laundering systems at financial institutions in Taiwan, operating under Financial Supervisory Commission regulatory standards. The combination of computational efficiency, operational cost reduction, and robust cross-scenario performance positions SRSGD as a practical solution for threshold optimization in regulatory compliance systems. Beyond anti-money laundering, the rank-scale transformation principle may prove valuable for other domains requiring discrete threshold optimization under skewed data distributions.

Appendix A Other Money Laundry Scenarios

Three typical suspicious scenario guidelines are used for Bank L. All three scenarios take into account the accumulated amounts and the number of transactions.

TWN-A1A-01

This scenario guideline monitors whether the accumulated credit and debit amounts, the number of transactions for each customer during a specific time frame exceed predetermined thresholds, and whether the ratio of debit to credit falls within a specified suspicious range. We follow the guideline to define five features as

$x_1^{TWN\text{-}A1A\text{-}01}$ = the accumulated credit amount during a period of d days,

$x_2^{TWN\text{-}A1A\text{-}01}$ = the number of credit transactions during the period of d days,
$x_3^{TWN\text{-}A1A\text{-}01}$ = the accumulated debit amount during the period of d days,
$x_4^{TWN\text{-}A1A\text{-}01}$ = the number of debit transactions during the period of d days,
$x_5^{TWN\text{-}A1A\text{-}01}$ = the ratio of $x_3^{TWN\text{-}A1A\text{-}01}$ to $x_1^{TWN\text{-}A1A\text{-}01}$

For each feature $x_i^{TWN\text{-}A1A\text{-}01}$ where $i \in \{1, 2, 3, 4\}$, we optimize a single threshold $t_i^{TWN\text{-}A1A\text{-}01}$. Feature $x_5^{TWN\text{-}A1A\text{-}01}$ (the debit-to-credit ratio) requires two thresholds $t_{5,lower}^{TWN\text{-}A1A\text{-}01}$ and $t_{5,upper}^{TWN\text{-}A1A\text{-}01}$ to define a suspicious range. The SAM label for a transaction account is obtained by substituting the features extracted from the account's transaction data into the following Boolean equation:

$$\left(\bigwedge_{i=1}^{4}\left(x_i^{TWN\text{-}A1A\text{-}01} \geq t_i^{TWN\text{-}A1A\text{-}01}\right)\right) \bigwedge \left(t_{5,lower}^{TWN\text{-}A1A\text{-}01} \leq x_5^{TWN\text{-}A1A\text{-}01} \leq t_{5,upper}^{TWN\text{-}A1A\text{-}01}\right)$$

TWN-AB1-01

This scenario guideline focuses on cross-border remittance and considers whether the accumulated amount and the number of transactions for each customer during a specific time frame exceed predetermined thresholds. We follow the guideline to define two features as

$x_1^{TWN\text{-}AB1\text{-}01}$ = the accumulated amount during a period of d days,
$x_2^{TWN\text{-}AB1\text{-}01}$ = the number of cross-border remittances during the period of d days.

For each feature $x_i^{TWN\text{-}AB1\text{-}01}$, we try to find an optimized threshold $t_i^{TWN\text{-}AB\text{-}01}$. The SAM label for a transaction account is obtained by substituting the features extracted from the account's transaction data into the following Boolean equation:

$$\left(x_1^{TWN\text{-}AB1\text{-}01} \geq t_1^{TWN\text{-}AB1\text{-}01}\right) \bigwedge \left(x_2^{TWN\text{-}AB1\text{-}01} \geq t_2^{TWN\text{-}AB1\text{-}01}\right)$$

TWN-A14-01

This scenario guideline considers whether the debit amount, the number of transactions, and the average debit amount during different periods exceed predetermined thresholds. We follow the guideline to define three features as

$x_1^{TWN\text{-}A14\text{-}01}$ = the average accumulated debit amount during a period of m months,
$x_2^{TWN\text{-}A14\text{-}01}$ = the accumulated debit amount during a period of d_2 days,
$x_3^{TWN\text{-}A14\text{-}01}$ = the number of debit transactions during a period of d_2 days.

For features $x_i^{TWN-A14-01}$ where $i \in \{2,3\}$, we optimize thresholds $t_i^{TWN-A14-01}$ directly. For feature $x_1^{TWN-A14-01}$ (average monthly debit), the threshold $t_1^{TWN-A14-01}$ is multiplied by a bank-specific ratio before comparison, as shown in the Boolean equation below.

$$\left(x_1^{TWN\text{-}A14\text{-}01} \geq t_1^{TWN\text{-}A14\text{-}01} \times ratio^{TWN\text{-}A14\text{-}01}\right) \bigwedge \left(\bigwedge_{i=2}^{3} \left(x_i^{TWN\text{-}A14\text{-}01} \geq t_i^{TWN\text{-}A14\text{-}01}\right)\right)$$

where $ratio^{TWN\text{-}A14\text{-}01}$ is an exogenous variable determined by Bank L.

Declarations

Funding

This research was supported by MetaEdge Corporation (https://www.metaedge.com.tw/en/).

Competing Interests

Author Hsin Yun Chang is affiliated with MetaEdge Corporation, which provided funding and data for this research. The remaining authors declare that they have no competing interests.

Data Availability

The datasets analyzed during the current study were provided by MetaEdge Corporation and are not publicly available due to their confidential and proprietary nature. Access to the data is restricted under data sharing agreements with the financial institutions involved.

Author Contributions

All authors contributed to the study conception and design. Material preparation, data analysis were performed by Tian-Shyr Dai, Wen-Wu Ti, and Chin-Yi Hsieh. Algorithm development was carried out by Tian-Shyr Dai and Ying-Ping Chen. Implementation and validation were conducted by Wen-Wu Ti and Mao-Chia Wu. Industry collaboration and data provision were facilitated by Hsin Yun Chang. The first draft of the manuscript was written by Tian-Shyr Dai and Wen-Wu Ti, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

References

[1] Cao, S., Yang, X., Chen, C., Zhou, J., Li, X., Qi, Y.: Titant: Online real-time transaction fraud detection in ant financial. arXiv preprint arXiv:1906.07407 (2019)

[2] Tai, C.-H., Kan, T.-J.: Identifying money laundering accounts. In: 2019 International Conference on System Science and Engineering (ICSSE), pp. 379–382 (2019). IEEE

[3] Shih, Y.-C., Dai, T.-S., Chen, Y.-P., Ti, Y.-W., Wang, W.-H., Kuo, Y.: Fund transfer fraud detection: Analyzing irregular transactions and customer relationships with self-attention and graph neural networks. Expert Systems with Applications **259**, 125211 (2025)

[4] Wang, D., Lin, J., Cui, P., Jia, Q., Wang, Z., Fang, Y., Yu, Q., Zhou, J., Yang, S., Qi, Y.: A semi-supervised graph attentive network for financial fraud detection. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 598–607 (2019). IEEE

[5] Zheng, Y.-J., Zhou, X.-H., Sheng, W.-G., Xue, Y., Chen, S.-Y.: Generative adversarial network based telecom fraud detection at the receiving bank. Neural Networks **102**, 78–86 (2018)

[6] Wiese, B., Omlin, C.: Credit Card Transactions, Fraud Detection, and Machine Learning: Modelling Time with LSTM Recurrent Neural Networks. Springer, ??? (2009)

[7] Jurgovsky, J., Granitzer, M., Ziegler, K., Calabretto, S., Portier, P.-E., He-Guelton, L., Caelen, O.: Sequence classification for credit-card fraud detection. Expert Systems with Applications **100**, 234–245 (2018)

[8] Roy, A., Sun, J., Mahoney, R., Alonzi, L., Adams, S., Beling, P.: Deep learning detecting fraud in credit card transactions. In: 2018 Systems and Information Engineering Design Symposium (SIEDS), pp. 129–134 (2018). IEEE

[9] Cheng, D., Xiang, S., Shang, C., Zhang, Y., Yang, F., Zhang, L.: Spatio-temporal attention-based neural network for credit card fraud detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 362–369 (2020)

[10] Alarab, I., Prakoonwit, S.: Graph-based lstm for anti-money laundering: Experimenting temporal graph convolutional network with bitcoin data. Neural Processing Letters **55**(1), 689–707 (2023)

[11] Vassallo, D., Vella, V., Ellul, J.: Application of gradient boosting algorithms for anti-money laundering in cryptocurrencies. SN Computer Science **2**, 1–15 (2021)

[12] Tang, J., Yin, J.: Developing an intelligent data discriminating system of anti-money laundering based on svm. In: 2005 International Conference on Machine Learning and Cybernetics, vol. 6, pp. 3453–3457 (2005). IEEE

[13] Chouiekh, A., Haj, E.H.I.E.: Convnets for fraud detection analysis. Procedia Computer Science **127**, 133–138 (2018)

[14] Alarab, I., Prakoonwit, S., Nacer, M.I.: Competence of graph convolutional networks for anti-money laundering in bitcoin blockchain. In: Proceedings of the 2020 5th International Conference on Machine Learning Technologies, pp. 23–27 (2020)

[15] Gupta, A., Dwivedi, D.N., Jain, A.: Threshold fine-tuning of money laundering scenarios through multi-dimensional optimization techniques. Journal of Money Laundering Control **25**(1), 72–78 (2022)

[16] Badics, T., Hajtó, D., Tornai, K., Kiss, L., Reguly, I.Z., Pesti, I., Sváb, P., Cserey, G.: Integral representation method based efficient rule optimizing framework for anti-money laundering. Journal of Money Laundering Control **26**(2), 290–308 (2023)

[17] Butgereit, L.: Anti money laundering: Rule-based methods to identify funnel accounts. In: 2021 Conference on Information Communications Technology and Society (ICTAS), pp. 21–26 (2021). IEEE

[18] Khan, N.S., Larik, A.S., Rajput, Q., Haider, S.: A bayesian approach for suspicious financial activity reporting. International Journal of Computers and Applications **35**(4), 181–187 (2013)

[19] Hsin, Y.-Y., Dai, T.-S., Ti, Y.-W., Huang, M.-C.: Interpretable electronic transfer fraud detection with expert feature constructions. In: CIKM Workshops (2021)

[20] Hsin, Y.-Y., Dai, T.-S., Ti, Y.-W., Huang, M.-C., Chiang, T.-H., Liu, L.-C.: Feature engineering and resampling strategies for fund transfer fraud with limited transaction data and a time-inhomogeneous modi operandi. IEEE Access **10**, 86101–86116 (2022)

[21] Jensen, R.I.T., Iosifidis, A.: Fighting money laundering with statistics and machine learning. Ieee Access **11**, 8889–8903 (2023)

[22] Jensen, R.I.T., Iosifidis, A.: Qualifying and raising anti-money laundering alarms with deep learning. Expert Systems with Applications **214**, 119037 (2023)

[23] Fan, J., Shar, L.K., Zhang, R., Liu, Z., Yang, W., Niyato, D., Mao, B., Lam, K.-Y.: Deep learning approaches for anti-money laundering on mobile transactions: Review, framework, and directions. arXiv preprint arXiv:2503.10058 (2025)

[24] Alkhalili, M., Qutqut, M.H., Almasalha, F.: Investigation of applying

machine learning for watch-list filtering in anti-money laundering. IEEE Access **9**, 18481–18496 (2021)

[25] Ti, Y.-W., Hsin, Y.-Y., Dai, T.-S., Huang, M.-C., Liu, L.-C.: Feature generation and contribution comparison for electronic fraud detection. Scientific Reports **12**(1), 18042 (2022)

[26] Xiang, Y., Kim, H.: Pipelined data-parallel cpu/gpu scheduling for multi-dnn real-time inference. In: 2019 IEEE Real-Time Systems Symposium (RTSS), pp. 392–405 (2019). IEEE

[27] Singh, A., Jain, A.: Cost-sensitive metaheuristic technique for credit card fraud detection. Journal of Information and Optimization Sciences **41**(6), 1319–1331 (2020)

[28] Duman, E., Elikucuk, I.: Solving credit card fraud detection problem by the new metaheuristics migrating birds optimization. In: Advances in Computational Intelligence: 12th International Work-Conference on Artificial Neural Networks, IWANN 2013, Puerto de la Cruz, Tenerife, Spain, June 12-14, 2013, Proceedings, Part II 12, pp. 62–71 (2013). Springer