AUDITING: A JOURNAL OF PRACTICE & THEORY
Vol. XX, No. XX
MONTH YEAR
pp. 1–27

American Accounting Association
DOI: 10.2308/AJPT-2024-058

Connecting the Dots: Graph Neural Networks for Auditing Accounting Journal Entries

Qing Huang
Marshall University

Marco Schreyer
University of California, Berkeley
University of St. Gallen

Nilson Romero Michiles, Jr.
Federal Data Processing Service (Serpro)
University of Brasilia

Miklos A. Vasarhelyi
Rutgers, The State University of New Jersey

SUMMARY: Based on the double-entry bookkeeping mechanism, each transaction is recorded in at least two ledger accounts, with one debit and the other one credit. Journal entry data, in the context of accounting, contains a rich network of information that can be effectively translated into a graph. This study explores how to use graph neural networks to learn graph representations from journal entry data and to systematically understand the complex patterns and connections that exist in journal entries at the transaction level. The real-world application results demonstrate that the unsupervised graph neural network framework offers a promising methodology for detecting error and fraud in financial audits.[1]

We would like to thank the two anonymous reviewers for their constructive comments. We sincerely appreciate the guidance and support provided by Christopher Koch (editor) and the senior editor. The research of Marco Schreyer is supported by an IFI fellowship (No. 57515245) of the German Academic Exchange Service (DAAD), and a Mobi.Doc fellowship (No. 1031606) of the University of St. Gallen, Switzerland. Qing Huang, Nilson Romero Michiles, Jr., and Miklos A. Vasarhelyi declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

This article is part of Qing Huang's Ph.D. dissertation (Huang 2024) at Rutgers, The State University of New Jersey.

The opinions expressed are those of the authors and do not represent the opinions or positions of the organizations.

Qing Huang, Marshall University, Brad D. Smith Schools of Business, Department of Marketing, Management Information Systems, and Entrepreneurship, Huntington, WV, USA; Marco Schreyer, University of California, Berkeley, International Computer Science Institute, Scientific Computing Group, Berkeley, CA, USA and University of St. Gallen, School of Computer Science, Artificial Intelligence and Machine Learning Group, St. Gallen, Switzerland; Nilson Romero Michiles, Jr., Federal Data Processing Service (Serpro), Brasilia, Distrito Federal, Brazil and University of Brasilia, Institute of Exact Sciences, Computer Science Department, Brasilia, Federal District, Brazil; Miklos A. Vasarhelyi, Rutgers, The State University of New Jersey, Rutgers Business School, Accounting & Information Systems, Newark, NJ, USA.

Any use of generative artificial intelligence (AI) or AI tools by the author(s) has been disclosed in the "Declaration of Generative AI and AI-Assisted Technologies" section.

Supplemental material is available online, as linked in the text.

Editor's note: Accepted by Christopher Koch, under the Senior Editorship of Michael S. Wilkins.

Submitted: May 2024
Accepted: December 2025
Early Access: February 2026

[1] A brief practitioner-oriented version of this work previously appeared in *Expert Focus*, the Swiss CPA professional journal. That earlier version was intentionally simplified for a nonacademic audience and did not include essential components, such as the detailed experimental design, model architecture, optimization procedures, data preprocessing steps, parameter tuning, full empirical results, or rigorous scholarly discussion. Please refer to Huang, Schreyer, Michiles, and Vasarhelyi (2024).

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

Data Availability: Dataset A comes from the Ernst & Young Academic Resource Center (EYARC), https://www.eyarc.site/blog/peach-state-university-hotel-understanding-audit-analytics-case

Keywords: graph neural network; graph representation; auditing; anomaly detection; latent space representation.

I. INTRODUCTION

The formalization of double-entry bookkeeping can be traced back to the late 15th century in Italy, notably through the work of Luca Pacioli (Sangster 2007). At that time, bookkeepers recorded accounting data manually. With the advent of computers and the implementation of enterprise resource planning (ERP) systems, the volume of accounting data has grown substantially, accompanied by a significant transformation in how organizations capture and store this information (Davenport 1998). Despite these technological advancements, auditors remain obligated to examine journal entries and assess the risk of fraud and unusual transactions as part of the financial statement audit. This responsibility is established by standards issued by the International Auditing and Assurance Standards Board (International Auditing and Assurance Standards Board (IAASB) 2009), the Public Company Accounting Oversight Board (Public Company Accounting Oversight Board (PCAOB) 2020), and the American Institute of Certified Public Accountants (American Institute of Certified Public Accountants (AICPA) 2023). Journal entries (JEs) are often regarded as the DNA of accounting information (Gray and Debreceny 2014); accordingly, their examination remains a fundamental component of audit procedures.

Given the centrality of journal entries in audit procedures, there is increasing interest in developing advanced analytical methods for their examination. Graph theory has been successfully applied across various domains, including information systems, knowledge graphs, ecosystems, sociology, and biological networks (Xia etal. 2021). In accounting and auditing, however, prior research has primarily leveraged graph theory for social network analysis (Jans, Alles, and Vasarhelyi 2014) or for visualizing journal entry data (Guo, Yu, and Wilkin 2022). Despite these contributions, a notable gap exists in applying graph-based machine learning techniques, particularly graph representation learning, to accounting data.

This paper addresses the gap by investigating the use of graph neural networks (GNNs) for analyzing journal entry data in audit analytics. The proposed graph neural network auditing (GNNA) framework comprises three main components. First, journal entries are transformed into two matrix representations: (1) a general ledger account co-occurrence matrix, and (2) a transaction-level feature matrix. Second, a GNN is trained to capture the structural and semantic characteristics of both matrices jointly. Third, the learned representations are used for unsupervised anomaly detection, flagging journal entries that deviate from established patterns. This approach offers a novel framework for identifying irregularities in accounting data using unsupervised representation learning.

This paper provides the following contributions to audit analytics and the broader application of machine learning in auditing:

1. We propose a data transformation pipeline that encodes journal entry data into two structured components, a general ledger account co-occurrence matrix and a transaction-level feature matrix, thereby enabling downstream graph-based representation learning.
2. We develop a graph neural network architecture that jointly learns from the transactional content and the relational structure of journal entries and captures unsupervised complex, organization-specific posting patterns.
3. We introduce a representation-based anomaly detection that flags journal entries deviating from learned patterns and provide a domain-adaptable technique for identifying anomalies corresponding to irregular or fraudulent accounting activity.
4. We apply the proposed methodology to real-world data and find that it not only identifies previously untested suspicious transactions but also significantly accelerates testing and enables full population analysis.

The remainder of this paper is structured as follows: Section II discusses related literature in graph analysis, representation learning, and graph representation learning in real-world accounting data. Section III provides background information about graph theory and the graph representation of journal entry data. Section IV explains the proposed methodology. Section V discusses the data used in this study and the main results. Section VI concludes the paper, summarizes the findings, and discusses the limitations and future study.

II. LITERATURE REVIEW

Recently, techniques based on (deep) machine learning have gradually been applied to various audit tasks (Cho, Vasarhelyi, Sun, and Zhang 2020; Sun 2019). Simultaneously, graph representation learning has led to significant academic research (Fey and Lenssen 2019; Hamilton 2020; Hamilton, Ying, and Leskovec 2017). Within the scope of this

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

work, the study focuses the literature review on (1) graph analysis, (2) representation learning, and (3) graph representation learning of accounting data.

Graph Analysis of Accounting Data

Graph theory has been effectively utilized in various fields (Xia etal. 2021). Deep learning on graphs has also been recognized as an influential tool for graph analysis (Xia etal. 2021; Zhang, Cui, and Zhu 2022). Several studies have demonstrated the potential of graph theory for understanding relationships in the accounting and auditing domain. For example, Arya, Fellingham, Glover, Schroeder, and Strang (2000) first represent the double-entry system in directed graphs and then apply this representation to auditing work and financial statement analysis (Arya, Fellingham, Mittendorf, and Schroeder 2004). McGlohon, Bay, Anderle, Steier, and Faloutsos (2009) propose using domain knowledge and link knowledge to detect misstated accounts in general ledger data. Still, their approach was limited to the node or account level. Certain ledger accounts are predictably high risk, as they are more susceptible to manipulation or misstatement due to inherent complexities and subjective estimates. Examples include accounts receivable, inventory, revenue, and capitalized expenses. The real challenge lies in uncovering suspicious transactions or identifying fraud at the edge or transaction level, and node-level anomalies provide limited additional value to auditors, who usually know which accounts are more suspicious already. Jans etal. (2014) use graph analysis techniques to detect segregation of duties violations in auditing business processes, offering another potential application of graph theory in auditing practice. Recently, Guo etal. (2022) suggest visualizing journal entry data at the account level to identify fraud. However, this analysis is constrained to aggregated transaction data, which may overlook critical information at the transaction level.

Representation Learning of Accounting Data

The exploration of representation learning and deep learning techniques in auditing accounting data has become a recent research focus and has produced valuable insights, especially in anomaly detection and the interpretation of JEs. Schreyer, Sattarov, Borth, Dengel, and Reimer (2017), Lenderink (2019), and Župan, Letinić, and Budimir (2018) propose to use deep autoencoders (AENs) and variational autoencoders (VAEs) to detect anomalies within large-scale accounting data. Their works demonstrate the utility of reconstruction error as an effective anomaly detection measure by analyzing anonymized datasets from real-world ERP systems. The capability of learning accounting data representations to enhance audit procedures has also been highlighted in recent studies. Schultz and Tropmann-Frick (2020), Zupan, Letinic, and Budimir (2020), and Nonnenmacher, Kruse, Schumann, and Gómez (2021) detail the application of AENs, VAEs, and long short-term memory (LSTM) architectures for detecting irregularities in accounting JEs. These contributions stress the potential of representation learning to improve audit processes.

Subsequent investigations have expanded the scope of anomaly detection. Schreyer, Sattarov, Schulze, Reimer, and Borth (2019) explore the use of deep adversarial autoencoder (AAE) networks to derive semantically meaningful representations of accounting JEs, using reconstruction error and the divergence from a specified prior distribution as indicators of anomalies. The introduction of the vector quantized-variational autoencoder (VQ-VAE) by Schreyer, Sattarov, Gierbl, Reimer, and Borth (2020) further advanced this field by facilitating the learning of quantized, low-dimensional representations of JEs. Hemati, Schreyer, and Borth (2021) proposed a continuous unsupervised anomaly detection technique using AENs, addressing catastrophic model forgetting and reducing false positives and negatives. A recent study by Yadav, Zhang, and Jin (2023) presents an anomaly detection algorithm using a modified transformer architecture for subledger accounting JEs and demonstrates its superior performance and adaptability compared with traditional methods.

Graph Representation Learning of Accounting Data

Very few studies have focused on graph representation learning in accounting data. Recently, Boersma, Sourabh, Hoogduin, and Kandhai (2018) proposed a method to aggregate and visualize the nodes and edges of financial statement networks; this visualization could help auditors evaluate a company's organizational complexity and indicate risk. Nguyen, Liang, and Akoglu (2023) propose a method that uses graph motifs to detect fraudulent accounting transactions at the graph level within a graph database. Boersma, Wolsink, Sourabh, Hoogduin, and Kandhai (2023) try to apply network representations of companies from their transaction datasets, measure their similarities to classify them further, and use the results to assist in audit work. Although these studies use graph representation learning of financial data, their focus remains on graph structure information at a higher level and fails to identify transaction-level anomalies.

In conclusion, prior literature emphasizes the potential of applying graph theory and deep learning to accounting data. This study integrates previous findings to facilitate graph anomaly detection in large-scale journal entry data at

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

the transaction level. To the best of our knowledge, this work represents the first step toward graph representation learning of real-world accounting data at the transaction level.

III. BACKGROUND

Recent advances in graph-based learning have introduced new methodologies for analyzing financial data. This section summarizes key concepts in graph theory, outlines the graph-structured nature of journal entries, and describes how these can be modeled to support graph representation learning.

Journal Entries as Graphs

Recording journal entries enables organizations to document the effect of business transactions on various accounts. According to the *Collins English Dictionary*, a journal entry is made directly into an organization's general journal to record every business transaction accurately and adequately through a process known as "journalizing" (Schreyer 2023). A typical journal entry includes various categorical and numerical attributes required to record a transaction accurately, such as date, general ledger accounts (name and/or number), amount, and debit or credit designation (Schreyer 2023).

A graph is defined as a set of nodes (vertices) and edges representing relationships between pairs of nodes (West 2001). In financial contexts, graph structures naturally emerge from the relational nature of accounting systems. Under double-entry bookkeeping, each journal entry involves at least two accounts, one of which is debited and the other credited, forming implicit connections between ledger accounts. These relationships can be formalized as graphs, where nodes represent general ledger accounts and edges represent debit-credit pairings within transactions. This structure captures transactional flows across accounts and allows large-scale accounting data to be modeled in a compact, analyzable form.

Transforming journal entries into graphs provides a novel perspective on accounting data. This method enables the application of graph representation learning techniques to audit complex journal entry data by conceptualizing accounts as nodes and journal entries as edges:

- Nodes represent general or subledger accounts on the organization's balance sheet or profit and loss statement, such as accounts receivable or sales.
- Edges represent the flow of financial resources within the organization's accounts, such as debit in the accounts receivable and credit sales.

This graph-based representation enables the modeling of transaction flows across accounts, allowing auditors to identify irregular patterns that may signal errors or fraudulent activity more effectively than traditional rule-based approaches.

Directed versus Undirected Journal Entry Graphs

A key modeling decision involves whether to use directed or undirected graphs. In the accounting domain, the concepts of debit and credit reflect asymmetrical financial movements, such as increases in assets versus decreases in liabilities, making directionality essential. Accordingly, we model journal entries as directed graphs, where edges represent the flow of value from debited to credited accounts. In contrast, undirected graphs, which treat relationships as bidirectional, do not preserve this transactional asymmetry and do not capture all accounting semantics.

An illustrative example of a directed graph representation is presented in Figure 1 (adapted from Huang, Schreyer, Michiles, and Vasarhelyi 2024). The journal entry #100011 debits account receivables (AR) by $1,032.00 and credits sales and value-added tax (VAT) by $1,000.00 and $32.00, respectively (Figure 1, left). This entry can be represented as a directed graph, where AR is the source node, and sales and VAT are target nodes (Figure 1, right). The resulting edges encode the directional flow of value from the debited account to the credited accounts and thus capture the transaction's structural semantics. This formulation is a basis for applying graph representation learning to journal entry data.

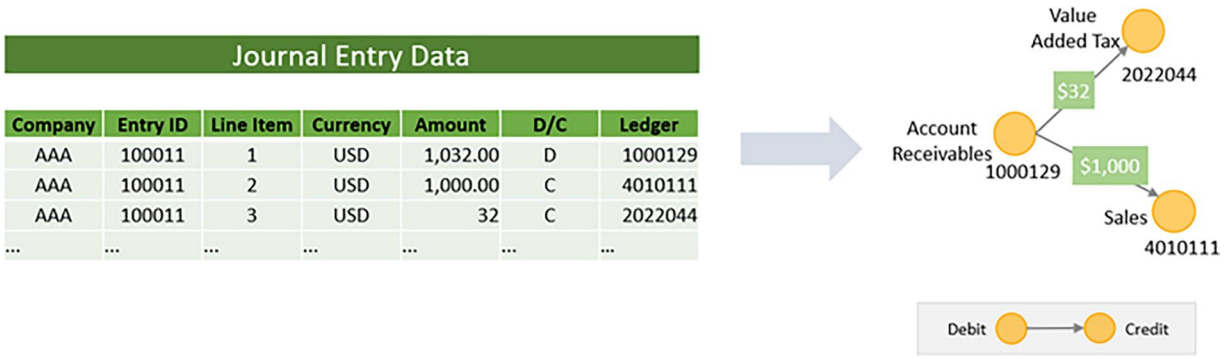
IV. GRAPH NEURAL NETWORK AUDITING FRAMEWORK

This section introduces the GNNA framework (see Figure 2, adapted from Huang etal. 2024), specifically designed to audit journal entry graph structures. The framework utilizes the attribute and graph information in JEs to learn lower-dimensional JE representations. The primary objective of the proposed framework is to learn rich embeddings of journal entries that preserve as much information as possible to support downstream audit tasks, such as anomaly detection and conformance checking. Traditional autoencoder-based methods typically model each line item individually, learning only from two data modalities: categorical attributes (such as account codes, cost center, etc.) and

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

FIGURE 1

Journal Entry Graph Representation



Source: Adapted from Huang et al. (2024, Figure 1).
(The full-color version is available online.)

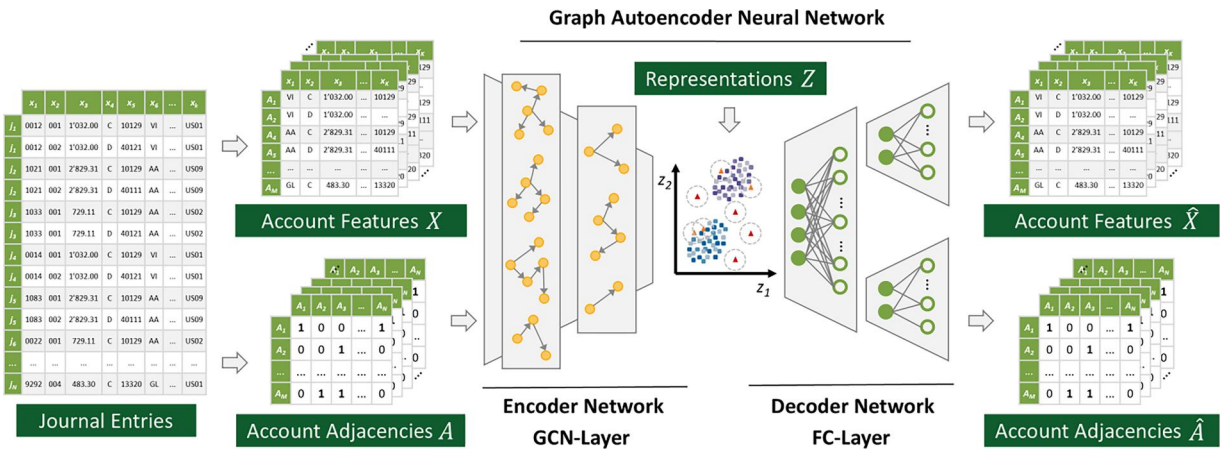
numerical attributes (such as amounts). These approaches do not capture the inherent relational structure of a journal entry, where debit and credit accounts are linked within a single transaction. By representing each entry as a graph and applying graph autoencoder neural networks (GAENs), the model not only learns from the individual accounts and amounts but also captures the pattern of interactions between accounts within the entry. To achieve such an audit objective, the GNNA framework comprises the following three interconnected phases, each targeting specific aspects of the auditing process:

Phase 1—Preprocessing of Journal Entries: The journal entries are transformed into adjacency and feature matrices. The adjacency matrix captures the account dependencies, mapping relationships between accounts involved in transactions. The feature matrix encodes categorical and numerical attributes, such as account types and amounts. Both matrices provide a numeric representation of each JE's characteristics.

Phase 2—Learning of Journal Entry Representations: A GAEN model is optimized to learn latent representations that integrate the JE's account dependencies and attributes. This dual learning objective ensures that

FIGURE 2

Graph Neural Network Auditing Framework (GNNA)



Source: Adapted from Huang et al. (2024, Figure 3).
(The full-color version is available online.)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

relational structures and contextual details are preserved, enabling the identification of anomalies that manifest in the JE's graph structure and attributes.

Phase 3—Auditing the Journal Entry Representations: The learned latent representations are systematically analyzed to identify patterns indicating conformance or irregularities. This analysis provides auditors with a comprehensive yet reliable way to examine the JE population under audit.

The subsequent sections provide a detailed explanation of each phase and detail the underlying methodologies and specific processes that constitute the framework.

Journal Entry Characteristics

Journal entries, as introduced in Section III, are characterized by multiple types of information that must be jointly modeled for effective auditing. These include categorical and numerical attributes and a graph structure capturing account relationships. The data modalities are defined as follows:

1. The categorical attributes, such as posting type, general ledger account, and transaction date, are represented as $x_i^a \in \mathbb{R}^a$, where a denotes the number of categorical features.
2. The numerical attributes, such as transaction amounts, are denoted as $x_i^b \in \mathbb{R}^b$, where b represents the number of numerical features.
3. The graph structure, such as the account debit-credit relationships, are modeled as a directed graph $G_i = (V, E)$, with account nodes V and debit-credit flow edges E.

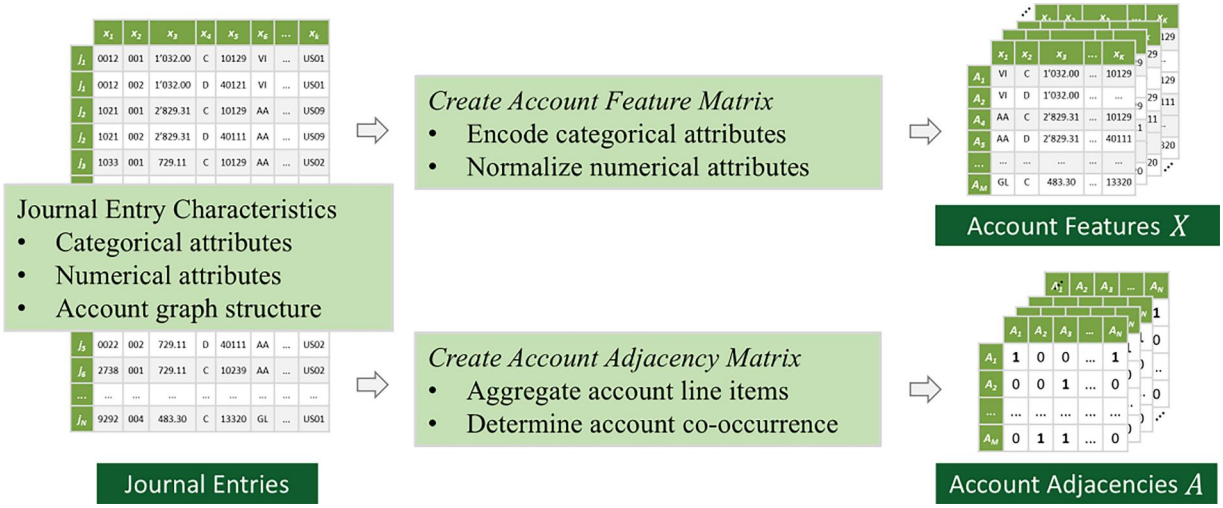
An individual JE is thus represented as a tuple $x_i = \left(x_i^a, x_i^b, G_i\right)$. A journal entry dataset encompassing N entries can then be expressed as $X = \left(x_1^a, x_1^b, G_1\right), \ldots, \left(x_N^a, x_N^b, G_N\right)$, integrating all three data modalities.

Create Account Feature Matrix

The preprocessing begins, as illustrated in Figure 3, with extract and transform categorical and numerical attributes from each journal entry. The categorical features are one-hot encoded, and numerical features are normalized to ensure scale invariance. The features are represented by a feature matrix $X \in \mathbb{R}^{n \times D}$, where n denotes the number of nodes (accounts) and D the dimensionality of the combined feature space. Each row X_i represents the feature vector of the ith node, encoding account-level attributes such as account type, posting key, and transaction amount. This representation integrates semantic information into the node-level input, providing a unified feature space for downstream graph learning. Within the GNNA framework, the feature matrix is used as input to capture the semantic attributes of

FIGURE 3

Journal Entry Preprocessing Steps



(The full-color version is available online.)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

TABLE 1

Feature Matrix Example

Account	Amount (Normalized)	Document Type_a	Document Type_b	…	Profit Center_x
AR	1	1	0	…	0
Sales	0.97	0	1	…	1
…	…	…	…	…	…

journal entries. It encodes account-level information essential for learning contextualized node representations in conjunction with the graph structure. Table 1 illustrates an example feature matrix.

Create Account Adjacency Matrix

Each journal entry is represented as a directed graph, where nodes correspond to general ledger accounts involved in the transaction. If multiple postings are made to the same account within a single entry, the associated amounts are aggregated before the graph is constructed. The graph structure is represented by an adjacency matrix $A \epsilon \mathbb{R}^{n \times n}$, where n represents the number of unique accounts. Each element A_{ij} reflects the presence and magnitude of a directed edge from account i (debited) to account j (credited), capturing the transaction flow between accounts. Within the GNNA framework, the adjacency matrix is used as input to capture the structural properties of journal entries. It encodes topological dependencies essential for learning latent representations through graph convolution operations.

Table 2 presents an illustrative example. Consider journal entry #100011 involving three general ledger accounts: AR, sales (S), and VAT. This transaction can be modeled as a directed graph $G = (V, E)$, where $V = \{\text{AR, S, VAT}\}$ denotes the set of nodes, and E captures the debit-credit flow edges. The corresponding 3×3 adjacency matrix assigns edges from AR to sales and VAT as 1, encoding structural and financial relationships within the entry.

Representation Learning from Journal Entries

To learn from the graph structures of JEs in auditing, we propose a GAEN model. This model encodes graph topology and node attributes into a compact latent space representation, enabling clustering and anomaly detection tasks. Derived from the variational graph autoencoder (VGAE) introduced by Kipf and Welling (2016), the GAEN model is specifically adapted to address the challenges of auditing financial data with limited labels and complex account relationships.

GAEN Architecture

The GAEN architecture encompasses graph convolutional network (GCN) layers, which perform feature propagation and aggregation based on the local neighborhood of each node (Kipf and Welling 2017). Formally, the forward propagation of the lth GCN layer is defined as:

$$H^{(l)}\left(A, H^{(l-1)}\right) = \sigma\left(\widetilde{D^{-1/2}} \widetilde{AD^{-1/2}} H^{(l-1)} W^{(l)}\right),$$

where $H^{(0)} = X$ is the input feature matrix, $H^{(l-1)}$ denotes the activation matrix from the previous layer, $\widetilde{A} = A + I$ is the adjacency matrix with added self-loops I, $\widetilde{D}$ is the degree matrix of $\widetilde{A}$, $W^{(l)}$ is the trainable weight matrix, and $\sigma(\cdot)$

TABLE 2

Adjacency Matrix Example

	AR	Sales	VAT
AR	1	1	1
Sales	0	1	0
VAT	0	0	1

American Accounting Association

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

is a nonlinear activation function. This formulation ensures that node features are aggregated while preserving the original graph structure.

The *encoder network* in the GAEN model consists of a sequence of L GCN layers that transform the input feature matrix X and adjacency matrix A into a latent space representation $Z \epsilon \mathbb{R}^{n \times d}$, where d denotes the latent space's dimension and $d < D$. Formally, the encoder network is defined as:

$$Z = H^{(L)}(A, H^{(L-1)}\Big(\ldots H^{(2)}(A, H^{(1)}\big(A, H^{(0)}\big)\big)\big)),$$

where $X = H^{(0)}$ is the input feature matrix, and each subsequent transformation aggregates features from the local neighborhood while applying trainable weights $W^{(l)}$ and nonlinear activations $\sigma(\cdot)$. This nested formulation explicitly captures the iterative nature of feature propagation and enables informative graph representations.

The *decoder network* of the GAEN model reconstructs both the adjacency matrix $\widehat{A}$ and the feature matrix $\widehat{X}$ from the latent space representation Z. The decoder architecture comprises a sequence of L fully connected layers. These transformations iteratively refine the latent representation Z by aggregating and propagating information through the decoder. Formally, the output of the lth layer is computed as:

$$Z = \sigma\big(H^{(l-1)} W^{(l)} + b^{(l)}\big),$$

where $H^{(l-1)}$ is the input from the preceding layer, $W^{(l)}$is the weight matrix, $b^{(l)}$ is the bias term, $\sigma(\cdot)$ denotes a nonlinear activation function, and $Z = H^{(0)}$. At a specific decoder layer l', the network diverges into two branches for reconstructing the adjacency matrix $\widehat{A}$ and the feature matrix $\widehat{X}$.

The adjacency reconstruction branch applies a fully connected layer followed by a sigmoid activation to reconstruct $\widehat{A}$, the adjacency matrix:

$$\widehat{A} = \sigma\Big(H_{adj}^{(l-1)} W_{adj}^{(l)} + b_{adj}^{(l)}\Big),$$

where $H_{adj}^{(l)}$ is the output of the final layer in the adjacency reconstruction branch, and $W_{adj}^{(l)}$ and $b_{adj}^{(l)}$ are trainable parameters. This reconstruction outputs edge probabilities, capturing the graph's structural connectivity and highlighting potential relationships between nodes.

The feature reconstruction branch applies a fully connected layer followed by a sigmoid activation to reconstruct $\widehat{X}$, the feature matrix:

$$\widehat{X} = \sigma\Big(H_{feat}^{(l-1)} W_{feat}^{(l)} + b_{feat}^{(l)}\Big),$$

where $H_{feat}^{(l)}$ is the output of the final layer in the feature reconstruction branch, and $W_{feat}^{(l)}$ and $b_{feat}^{(l)}$ are trainable parameters. Each row of $\widehat{X}$ corresponds to the reconstructed features of a journal entry line item, ensuring that journal entries are effectively recovered.

By separating the adjacency and feature matrix reconstruction at a designated layer, the decoder ensures that each task is optimized independently while benefiting from a shared latent representation.

GAEN Model Optimization

The optimization of the GAEN model is guided by combined reconstruction loss that jointly minimizes the adjacency matrix reconstruction loss $\mathcal{L}_{adj}$ and the feature matrix reconstruction loss $\mathcal{L}_{feat}$. Formally, the reconstruction loss is defined as:

$$\mathcal{L}_{rec} = \beta \times \mathcal{L}_{adj} + (1 - \beta) \times \mathcal{L}_{feat},$$

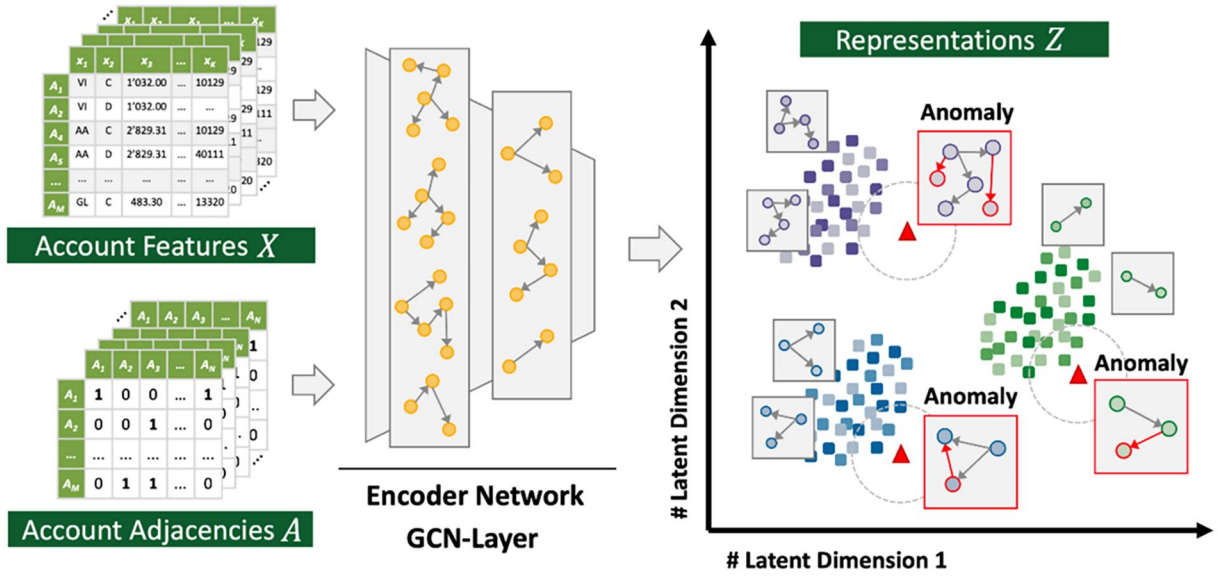
where $\beta \in [0, 1]$ balances the contribution of the two terms $\mathcal{L}_{adj}$ and $\mathcal{L}_{feat}$. The adjacency reconstruction loss evaluates the alignment between the original adjacency matrix A and its reconstruction $\widehat{A}$. Each entry A_{ij} represents the predicted edge weight between nodes i and j, and the adjacency reconstruction loss is computed as:

$$\mathcal{L}_{adj} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \|A_{ij} - \widehat{A}_{ij}\|^2$$

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

FIGURE 4

Journal Entry Representation Audit



Source: Adapted from Huang et al. (2024, Figure 4).
(The full-color version is available online.)

where N denotes the number of nodes and A_{ij} and $\widehat{A}_{ij}$ are the original and reconstructed adjacency matrix. The feature reconstruction loss evaluates the fidelity of the reconstructed feature matrix $\widehat{X}$ compared to the original feature matrix X. Each reconstructed feature vector $\widehat{X}_i$ corresponds to a journal entry line item i, and the feature reconstruction loss is computed as:

$$\mathcal{L}_{feat} = \frac{1}{N \times D} \sum_{i=1}^{N} \sum_{j=1}^{N} \|X_{ij} - \widehat{X}_{ij}\|^2$$

where N denotes the number of nodes, D is the number of node features, and X_{ij} and $\widehat{X}_{ij}$ are the original and reconstructed features for node i and feature j. Normalization factors are applied to both $\mathcal{L}_{adj}$ and $\mathcal{L}_{feat}$ to account for the scale differences between the adjacency and feature matrices. These factors ensure that the contributions of both losses are balanced during optimization, regardless of the dimensionality of the inputs.

By minimizing $\mathcal{L}_{rec}$, the GAEN framework learns a latent space representation Z that captures structural relationships and node-level attributes, facilitating downstream tasks such as anomaly detection and conformance analysis.

Audit of Journal Entry Representations

The proposed GNNA framework learns graph representations by embedding journal entries into a compact latent space. The encoder effectively captures structural and contextual JE features by applying GCN layers, producing latent representations that retain the JE graph's integrity while highlighting unusual patterns. The learned latent representations can be utilized for downstream audit tasks, with a focus on two applications:

- **Anomaly detection:** Journal entries are anomalous if they exhibit significant deviation from cluster centroids in the latent space. Such deviations indicate divergence from standard organizational processes and accounting policies.
- **Conformance checking:** Journal entries are compliant if they are near cluster centroids. Such proximity to cluster centers indicates alignment with standard organizational processes and accounting policies. Figure 4 illustrates this end-to-end process. Account features X (semantic information) and account adjacencies A (structural information) are encoded through graph convolutional layers to produce node-level embeddings. These embeddings are projected into a latent space Z, where structurally and semantically similar journal entries form dense clusters. High-density clusters (colored regions) in the latent space correspond to frequently observed and regular journal entries, supporting conformance checking. In contrast, anomalous journal

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

entries (marked by red triangles and frames) that fall outside clusters (identified by their spatial isolation) are flagged as anomalies. These journal entries often reflect structurally or contextually irregular patterns that deviate from the organization's normative posting behavior.

In brief, the proposed GNNA framework enables auditors to examine graph representations, provides insights into the interconnected nature of accounts, and helps to ensure adherence to accounting practices and identify potential risks.

V. EXPERIMENTAL SETUP AND RESULTS

A series of experiments is conducted to evaluate the GNNA framework under realistic audit conditions. Both quantitative and qualitative assessments are used in this section: the former validates the framework's theoretical performance, and the latter examines its effectiveness in real audit practice.[2]

Experimental Setup

The following introduces the experimental setup, which consists of anomaly injection, graph representation learning, anomaly scoring, and quantitative evaluation using both reconstruction-based and latent-space methods.

Accounting Anomalies

To evaluate the quantitative performance of the proposed methodology, anomalies are injected into the data. Following the approach of Breunig, Kriegel, Ng, and Sander (2000), two types of anomalies are embedded, referred to as global and local accounting anomalies:

- **Global accounting anomalies** correspond to JEs that involve accounts not observed elsewhere in the dataset. Such anomalies often reflect unintentional mistakes, are comparatively simple to detect, and possess a high error risk.
- **Local accounting anomalies** correspond to JEs that involve account combinations not observed elsewhere in the dataset. Such anomalies may reflect intentional deviations, are comparatively difficult to detect, and possess a high fraud risk.

Ultimately, auditors aim to detect both types of accounting anomalies in real-world audits using unsupervised learning (Schreyer etal. 2017).

Dataset and Dataset Preparation

Two journal entry datasets (one case study dataset comes from the Ernst & Young Academic Resource Center (EYARC),[3] and one real-world dataset from an industry partner) are used to evaluate the effectiveness of the proposed methodology:

- **Dataset A (EYARC):** This dataset consists of 37,869 journal entry line items corresponding to 18,934 unique journal entries and 74 general ledger accounts.
- **Dataset B (real-world):** This dataset consists of 1,005,271 journal entry line items corresponding to 252,811 unique journal entries and 526 general ledger accounts.

Each dataset is preprocessed (as described in Section IV) to extract structured representations of journal entries and derive a numerical representation. The categorical attributes are converted into binary vectors using one-hot encoding. The numerical attributes are normalized using the min-max scaling method.

Graph Representation Learning

The GNNA framework consists of a GCN-based encoder and two fully connected decoder networks, as shown in Figure 2 and detailed in Table 3. The encoder comprises multiple graph convolutional layers that jointly capture node attributes and local graph structure. Latent representations are embedded in a two-dimensional space Z to facilitate downstream tasks such as visualization and anomaly detection. The account feature matrix and the adjacency matrix are reconstructed independently from the shared latent space using two decoder branches, each implemented as a fully connected network.

[2] Main code used in this paper is available in the Online Appendix.
[3] https://www.eyarc.site/blog/peach-state-university-hotel-understanding-audit-analytics-case

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

TABLE 3

Number of Neurons per Layer

Net	Dataset	Layer 1	Layer 2	Layer 3	Layer 4	…	Final Layer
Encoder	A	128	64	32	16	…	2
Decoder	A	2	8	16	32	…	128
Encoder	B	64	32	16	8	…	2
Decoder	B	2	4	8	16	…	64

Models are optimized using the Adam optimizer ($\beta_1 = 0.9$, h $\beta_2 = 0.999$) with a cosine learning rate decay schedule. The model is trained to minimize the joint reconstruction loss described in Section V, which balances the reconstruction of the feature and adjacency matrices. The loss weighting parameter β is set to 0.5 in all experiments. Early stopping is applied based on the convergence of the training loss.

Hyperparameter selection plays a critical role in model stability and performance. To identify an effective configuration, the learning rate $\eta \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ and the batch size $\rho \in \{32, 64, 128, 256\}$ were systematically evaluated. Based on empirical results, $\eta = 10^{-4}$ and $\rho = 64$ were selected (please find more detailed parameter tuning process in Appendices A and B).

Anomaly Detection

Two unsupervised anomaly-detection strategies are evaluated to assess the effectiveness of the learned representations. All approaches operate on the outputs of the GNNA framework and are designed to detect global and local accounting anomalies, as previously defined.

1. **Reconstruction error-based detection (GAEN + RE):** Anomalies are scored based on the magnitude of the reconstruction error computed from the joint loss over the feature and adjacency matrices. Entries with higher reconstruction error are considered less well represented by the model and are thus assigned higher anomaly scores.
2. **Latent representation-based detection (GAEN + Algo):** Anomaly scores are computed by applying state-of-the-art outlier detection algorithms to the two-dimensional latent representations produced by the GAEN encoder. Each algorithm assigns higher outlier scores to entries it considers more anomalous, based on its own notion of abnormality. For example, for GAEN + local outlier factor (LOF) (Breunig etal. 2000), anomaly scores are derived from local density estimates, with anomalies identified as points residing in sparsely populated regions. For GAEN + Isolation Forest (IForest) (Liu, Ting, and Zhou 2008), entries that are separated with fewer partitions are assigned higher anomaly scores.

The two detection setups target complementary anomaly characteristics: reconstruction-based detection captures deviations from learned structural and attribute patterns, whereas anomaly detection-based latent-space methods assess statistical separability based on the geometry of the embedding space.

Quantitative Evaluation Measures

The performance of each anomaly detection setup is evaluated using standard binary classification metrics, including accuracy, precision, recall, and F1 score. These metrics quantify the model's ability to correctly identify injected anomalies while minimizing false positives and false negatives. In addition to these threshold-based measures, average precision (AP) is also used to evaluate the performance of proposed methodology. The AP (Everingham, Van Gool, Williams, Winn, and Zisserman 2010) is a comprehensive evaluation metric; it summarizes the precision-recall tradeoff across all possible decision thresholds and is particularly suited for unsupervised anomaly detection settings. Higher AP values indicate that true anomalies consistently receive higher anomaly scores than regular entries. Given an anomaly score S, the AP is computed as:

$$\mathrm{AP}(S) = \sum_{i=1}^{\mathrm{N}} (R_i - R_{i-1}) \cdot P_i,$$

where P_i and R_i denote precision and recall at the ith threshold, respectively. This formulation evaluates the quality of the induced ranking without requiring a fixed classification cutoff.

American Accounting Association

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

Experimental Results

This section empirically examines the proposed approach through both controlled and real-world evaluations. The GNNA framework evaluation is organized around three experiments:

- Experiment 1—Anomaly detection on synthetic accounting data: Anomalies are introduced into Dataset A to quantify the framework's ability to distinguish global and local irregularities in an unsupervised setting.
- Experiment 2—Anomaly detection on real-world accounting data: Auditors review top-ranked journal entries in the real-world Dataset B to assess whether the framework detects materially relevant irregularities.
- Experiment 3—Conformance checking on synthetic accounting data: Anomalies are introduced into synthetic Dataset A to determine whether the learned representations reflect the underlying accounting structure and facilitate interpretability.

Together, the experiments are designed to evaluate the proposed framework's technical performance and practical utility in realistic auditing contexts.

Experiment 1—Quantitative Results on Synthetic Accounting Data

To examine the GNNA framework's performance in a controlled setting, a total of 200 anomalies (100 global and 100 local) were injected into Dataset A. The detection performance was computed by comparing detected anomalies against the known injected labels. The evaluation assesses whether the framework discriminates between anomalies and journal entries (without relying on the supervised labels). Table 4 reports anomaly detection results on the Dataset A, evaluating the GAEN-based methods against the five baselines: IForest, LOF, histogram-based outlier detection (HBOS) (Goldstein and Dengel 2012), principal component analysis (PCA) (Shyu, Chen, Sarinnapakorn and Chang 2003), and AutoEncoder (Aggarwal 2017). The results are presented across three panels corresponding to all anomalies, global anomalies, and local anomalies:

- All anomalies (Panel A): The GAEN + IForest (AP = 0.1895) and GAEN + HBOS (AP = 0.1467) achieve the strongest results, outperforming their respective baselines—IForest (0.1138) and HBOS (0.0318). Both models also have higher accuracy, precision, recall, and F1-scores than all other methods. With the exception of GAEN + RE, the remaining GAEN-based methods achieve higher AP than their standalone methods. These findings indicate that the proposed graph-based methodology enhances anomaly detection on journal entry data across all anomaly types overall.
- Global anomalies (Panel B): The GAEN + HBOS (AP = 0.2131) and GAEN + IForest (AP = 0.2055) achieve the strongest results, both outperforming their respective baselines—HBOS (0.0170) and IForest (0.0883). Both methods also yield higher accuracy, precision, recall, and F1-scores than all standalone methods. Other graph-based methods all achieve higher AP than their standalone methods, suggesting that graph-based methods strengthen anomaly detection for distinct global anomalies.
- Local anomalies (Panel C): The IForest baseline (AP = 0.0610) and GAEN + IForest (AP = 0.0496) achieve the strongest results, while other methods show substantially lower AP values. GAEN-based models also underperform the strongest baseline across remaining matrices. These results imply that graph-based methods may be less effective when the task focuses solely on detecting local anomalies.

The results indicate that the GNNA framework's representations can enhance anomaly detection performance, particularly when paired with density-based or ensemble-based outlier detectors. Gains are most pronounced for global anomalies, where structural irregularity is more evident. For local anomalies, performance remains modest, indicating a need for additional modeling capacity or fine-grained relational supervision. Overall, the findings support the effectiveness of the framework for detecting anomalous journal entries in financial audits.

Experiments 2—Qualitative Evaluation on Real-World Accounting Data

Dataset B is real-world journal entry data tested concurrently with real yearly auditing work and lacks predefined labels. It is used to demonstrate the applicability and usefulness of the proposed methodology in a practical setting. Accordingly, Dataset B undergoes qualitative evaluation: the top-ranked suspicious transactions identified by the methodology are forwarded to auditors for further examination. Auditors then manually verify these transactions to confirm the correctness of the selection. This approach is considered appropriate because it reduces the burden to label a large set of transactions, a task that can sometimes be prohibitively expensive or even impossible in audit work (Fulcer etal. 2025).

As part of the tests, a qualitative assessment was conducted using a database from a partner company for the fiscal year 2022 (Dataset B). In collaboration with the partner company's internal auditors, we reviewed the entries ranked

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

TABLE 4

Quantitative Results

Panel A: Results for All Journal Entry Anomalies

Model	AP ↑	Accuracy ↑	Precision ↑	Recall ↑	F1-Score ↑
GAEN + IForest (ours)	**0.1895**	0.9862	0.3400	**0.3400**	**0.3400**
GAEN + LOF (ours)	0.0863	0.9803	0.0000	0.0000	0.0000
GAEN + PCA (ours)	0.0803	0.9841	0.2400	0.2400	0.2400
GAEN + HBOS (ours)	0.1467	**0.9871**	**0.3487**	0.2650	0.3011
GAEN + RE (ours)	0.0189	0.9796	0.0250	0.0250	0.0250
IForest (baseline)	0.1138	0.9827	0.1729	0.1729	0.1729
LOF (baseline)	0.0448	0.9792	0.0025	0.0025	0.0025
PCA (baseline)	0.0551	0.9814	0.0956	0.0927	0.0941
HBOS (baseline)	0.0318	0.9806	0.0702	0.0702	0.0702
AutoEncoder (baseline)	0.0544	0.9814	0.0956	0.0927	0.0941

The results correspond to the hyperparameter configuration that achieved the highest AP for all anomalies.

Panel B: Results for Global Journal Entry Anomalies

Model	AP ↑	Accuracy ↑	Precision ↑	Recall ↑	F1-Score ↑
GAEN + IForest (ours)	0.2055	**0.9900**	**0.2727**	**0.5400**	**0.3624**
GAEN + LOF (ours)	0.0530	0.9854	0.0000	0.0000	0.0000
GAEN + PCA (ours)	0.1131	0.9889	0.2211	0.4400	0.2943
GAEN + HBOS (ours)	**0.2131**	0.9897	0.2563	0.5100	0.3411
GAEN + RE (ours)	0.0107	0.9847	0.0201	0.0400	0.0268
IForest (baseline)	0.0883	0.9872	0.1392	0.2750	0.1849
LOF (baseline)	0.0258	0.9843	0.0025	0.0050	0.0034
PCA (baseline)	0.0442	0.9863	0.0765	0.1450	0.1002
HBOS (baseline)	0.0170	0.9850	0.0328	0.0650	0.0436
AutoEncoder (baseline)	0.0433	0.9863	0.0765	0.1450	0.1002

The results correspond to the hyperparameter configuration that achieved the highest AP for all anomalies.

Panel C: Results for Local Journal Entry Anomalies

Model	AP ↑	Accuracy ↑	Precision ↑	Recall ↑	F1-Score ↑
GAEN + IForest (ours)	0.0496	0.9860	0.0804	0.1600	0.1070
GAEN + LOF (ours)	0.0386	0.9854	0.0000	0.0000	0.0000
GAEN + PCA (ours)	0.0097	0.9849	0.0302	0.0600	0.0401
GAEN + HBOS (ours)	0.0177	0.9853	0.0263	0.0500	0.0345
GAEN + RE (ours)	0.0097	0.9846	0.0051	0.0100	0.0068
IForest (baseline)	**0.0610**	**0.9867**	**0.1134**	**0.2261**	**0.1510**
LOF (baseline)	0.0240	0.9843	0.0000	0.0000	0.0000
PCA (baseline)	0.0200	0.9852	0.0235	0.0452	0.0309
HBOS (baseline)	0.0178	0.9852	0.0429	0.0854	0.0571
AutoEncoder (baseline)	0.0201	0.9852	0.0235	0.0452	0.0309

The results correspond to the hyperparameter configuration that achieved the highest AP for all anomalies. Dataset A was processed on a desktop equipped with an Intel i7 processor, a GeForce 3060 GPU, and 16 GB of RAM. The proposed methodology took approximately 11 minutes to complete. For detailed hyperparameters, see Appendix C.

highest by (1) GAEN + Algo anomaly scores, and (2) GAEN + RE reconstruction errors to see whether they looked questionable and whether real-time alerts would be valuable. No synthetic anomalies were injected.

As shown in Table 5, in auditing quarter 1 data, we selected the top ten risky transactions ranked by the combination of GAEN with IForest and GAEN reconstruction error, respectively. Upon investigation, for the top ten of

American Accounting Association

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

TABLE 5

Qualitative Results for Dataset B

Period	Model	# of Journal Entry Selected to Investigate	Average Amount of Selected Transactions	Auditor Qualitative Feedback			Sample Situations Filled by Auditors (Not Exhaustive)
				Is There a Potential Error?	Is It Rare/Unusual?	Would You Like to Receive Alerts Like This?	
Q1[a]	GAEN + IForest	10	$8.07	Yes (10.00%)	Yes (80.00%)	Yes (10.00%)	-Duplicate records -Self-postings -Records with 0 value
	GAEN + RE	10	$116,862.87	Yes (40.00%)	Yes (50.00%)	Yes (50.00%)	
Q1 & Q2[b]	GAEN + IForest	10	$591,086.59	Yes (70.00%)	Yes (100.00%)	Yes (90.00%)	-Self-postings -Unexpected reversal -Unexpected Transfer between ledger accounts
	GAEN + RE	10	$293.83	Yes (100.00%)	Yes (100.00%)	Yes (100.00%)	
Fiscal Year[c]	GAEN + IForest	860	$134.04	Yes (98.26%)	Yes (98.84%)	Yes (98.26%)	-Unusual manual entries -Unexplained reclassification entries -Reversal without explanations -Unexpected Transfer between ledger accounts -Self-postings
	GAEN + RE	863	$85.00	Yes (99.53%)	Yes (100.00%)	Yes (99.53%)	

Dataset B was processed on a desktop equipped with an Intel i9 processor, a GeForce 4070i GPU, and 20 GB of RAM. The processing times for the proposed methodology varied depending on the data size: approximately 40 minutes for Q1, around 2 hours for Q1 and Q2, and roughly 23 hours for the entire fiscal year. Notably, the construction of the adjacency matrix accounted for the majority of the processing time.

[a] Q1 has 128,079 journal entry line items across 51,256 journal entries, with an average entry amount of $14,097.30.

[b] Q1 and Q2 have 313,778 journal entry line items across 127,503 journal entries, with an average entry amount of $12,917.95.

[c] The whole fiscal year has 1,005,271 journal entry line items across 252,811 journal entries, with an average entry amount of $17,281.15.

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/1561133/ajpt-2024-058.pdf by guest on 09 March 2026

GAEN + IForest, auditors considered one of them a potential error, eight truly rare or unusual, and only one transaction for which auditors would like to receive similar alerts in the future. For the top ten selected by GAEN + RE, auditors identified four as potential errors, five as truly rare or unusual, and would like to receive similar alerts for five of them in the future. Auditors categorized these errors as self-postings, duplicate records, and records with zero value.

In the audit of quarter 1 and quarter 2 data, we followed the same selection process. For the ten transactions selected by GAEN + IForest, auditors found seven to be potential errors, all to be rare and unusual, and would like to receive alerts for all of them in the future. The ten transactions selected by GAEN reconstruction error were more noteworthy. Auditors considered all of them potential errors or unusual or rare and would like to receive alerts for all of them in the future. These errors or anomalies were categorized as unexpected reversals, unexpected transfers between ledger accounts, and self-postings.

In the audit of the entire fiscal year, we select 860 transactions using GAEN + IForest and 863 under GAEN + RE based on the anomaly scores. For transactions selected by GAEN + IForest, 98.26 percent are potential errors, 98.84 percent are rare or unusual, and auditors would like to receive similar alerts for 98.26 percent of them in the future. For transactions selected by GAEN + RE, auditors think all of them are rare or unusual, 99.53 percent are potential errors, and auditors would like to receive alerts for 99.53 percent of the selected transactions in the future. Auditors categorize these errors or anomalies as unusual manual entries, unexplained reclassification entries, reversals without explanations, unexpected transfers between ledger accounts, and self-postings.

Both GAEN + IForest and GAEN + RE have good performance in catching suspicious transactions, and the GAEN + RE works slightly better than the GAEN + IForest, especially in identifying potential errors. The qualitative results also demonstrate that the proposed methodology can not only detect suspicious transactions related to general ledger accounts combination but also identify other types of risky transactions, such as unexpected reversal, duplicate records, and records with zero values. The mentioned 860 transactions using GAEN + IForest and 863 transactions under GAEN + RE could all be identified using conventional rule-based tests only if auditors had prior knowledge of the specific types of risky or suspicious transactions they intended to detect. However, the case company's internal audit activities are planned and executed in accordance with the results of the risk assessment and the vertical and horizontal analyses of the financial statement accounts, and are further refined by prior audit findings, concerns raised by the board of directors, and the audit team's professional judgment. Under this setting, the transactions our method flagged were never separately tested because they were not identified as risky under the existing approach. In practice, auditors had not previously focused on these patterns and therefore did not include them in their testing scope. This observation further verifies our contribution. Existing risk-based audits rarely perform full-population, relationship-aware testing, so "unknown-unknown" patterns can be missed. Our method checks the entire journal population and highlights entries that deviate from the mainstream transactions, giving auditors a prioritized list for follow-up.

The efficiency implications are also material. According to the company's internal audit report, the 2022 financial statement audit consumed 1,096 hours across 34 tasks. By contrast, our full-population screening ran in approximately 23 hours of computing time, enabling auditors to redirect effort toward investigating the highest-signal items while expanding coverage. We emphasize that this is a complement to, not a replacement for, the existing audit framework. Deployed as a pre-audit scan or concurrent monitoring step, it can improve efficiency and effectiveness of the audit process.

Experiment 3—Conformance Checking on Synthetic Accounting Data

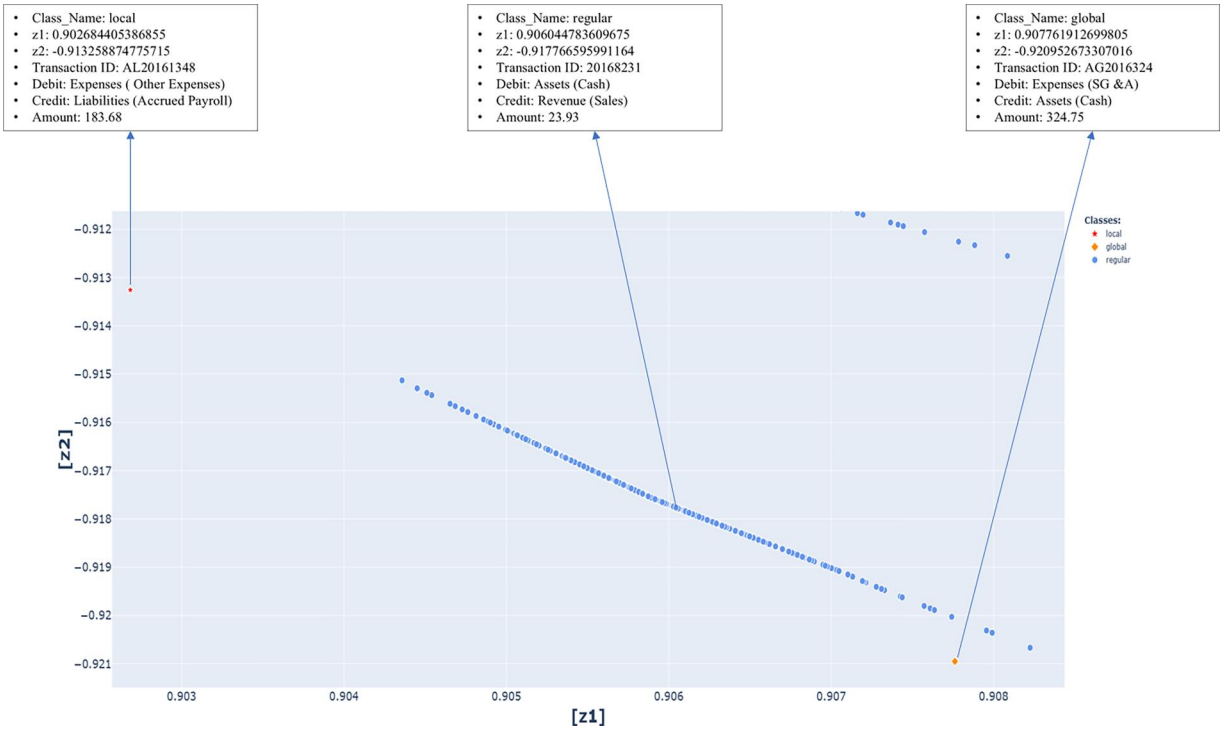
The latent space representations learned by the GNNA framework offer a means of assessing the structural consistency of journal entries. The encoder yields low-dimensional embeddings that preserve relevant semantic and topological characteristics by integrating node-level attributes and graph-based dependencies. In the two-dimensional latent space Z spanned by dimensions $z1$ and $z2$, each point corresponds to an individual journal entry. The spatial configuration reflects relational similarity, with entries closer together likely sharing structural or contextual features. Visual encodings distinguish between entry types: blue circles indicate entries labeled as regular, orange diamonds represent global anomalies, and red stars correspond to local anomalies.

As shown in Figure 5, clusters tend to emerge in regions associated with frequent and consistent posting patterns. One such cluster appears to capture transactions debiting assets accounts and crediting revenue, suggesting that the model learns common accounting structures. In contrast, several points lie outside these dense regions. A red star, for instance, is associated with a transaction debiting expenses and crediting liabilities, while an orange diamond represents an entry debits expenses and credits assets account. These entries differ from their surrounding context and are flagged as anomalous under the model's learned representation.

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

FIGURE 5

Latent Space Representation of Dataset A



(The full-color version is available online.)

VI. SUMMARY AND CONCLUSION

This study proposes the GNNA framework as a graph-based approach for detecting anomalies in journal entries using representation learning. Journal entries are modeled as directed graphs to capture transactional content and structural relationships, which offers a new perspective for analyzing complex accounting data.

The GNNA framework is structured into three core phases: (1) the transformation of journal entries into graph and feature matrix formats, (2) the learning of low-dimensional embeddings through a GAEN architecture, and (3) the auditing of these representations through anomaly scoring and conformance checking. This design enables auditors to identify transaction-level irregularities beyond what rule-based methods typically capture.

The framework is evaluated on both a benchmark and a real-world dataset from an industry partner. Empirical results show that GNNA improves anomaly detection performance, especially when combined with outlier detection methods. A qualitative evaluation with auditors confirms that the flagged transactions are often relevant and actionable in practical audit scenarios.

This study highlights the use of graph autoencoder neural networks to learn latent representations of journal entries. The VGAE model identifies subtle patterns and anomalies within the data, which are crucial for detecting fraudulent activities or errors in financial statements. By integrating graph representation learning with financial auditing practices, the proposed methodology offers a new approach to analyzing journal entries, enables the identification of accounting patterns and anomalies, and contributes to more effective auditing.

Limitations

Although the proposed GNNA framework shows promising results for identifying anomalies, several limitations should be carefully considered when interpreting the findings. First, the study is constrained by dataset diversity. The study uses one academic case dataset and one real-world dataset from an industry partner. Although these datasets provide valuable insights, they do not fully reflect the range of journal entry patterns and accounting practices across different industries and

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

ERP environments. As a result, the generalizability of the framework's performance remains to be evaluated in more diverse audit settings. Second, the research focuses on structured numerical and categorical features and does not incorporate unstructured text fields, such as descriptions or justifications, or temporal information. These types of information often carry valuable signals for identifying irregular or fraudulent activity and could be integrated in future study. Third, the processing time can be slow on large datasets, especially when individual transactions contain many line items that expand the graph size. Finally, the model provides limited explainability. It sometimes does not clearly indicate why specific entries are flagged, and its latent-space representation provides little insight into which attributes drive the anomaly scores. We encourage further research to explore and develop more comprehensive GNN-enabled auditing techniques.

Conclusion

In conclusion, as graph-based machine learning evolves, its application in auditing becomes increasingly practical and impactful. Using graph neural networks for journal entry analysis presents a promising direction for modernizing audit procedures and enhancing anomaly detection at scale. The future role of graph learning in auditing is significant, with frameworks like GNNA offering new ways to uncover irregularities that may otherwise remain hidden.

Auditors contribute domain expertise, professional skepticism, and context-specific judgment. Graph neural networks, in turn, provide powerful tools for modeling complex relationships, detecting patterns, and scaling analysis across large volumes of data. Jointly, these capabilities can support a more effective and adaptive approach to auditing large-scale journal entry data. By integrating unsupervised representation learning, the GNNA framework lays a foundation for a novel direction of data-driven audit analytics, which can support auditors in navigating increasingly complex digital financial audit landscapes.

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

In the preparation of this manuscript, the authors employed the Chat Generative Pretrained Transformer (ChatGPT-4o) developed by OpenAI, which supports a 128,000-token context window and incorporates training data up to October 2023. ChatGPT served as an auxiliary tool for spell checking and reviewing language and grammar. Additionally, the Grammarly software was utilized to assist in the proofreading and refinement of the manuscript. The inclusion of ChatGPT and Grammarly aimed to elevate the manuscript's linguistic quality and enhance the clarity, coherence, and precision of the research presented. The authors have thoroughly reviewed and revised the output from these tools and accept full responsibility for the content and conclusions presented in the manuscript.

REFERENCES

Aggarwal, C. C. 2017. An introduction to outlier analysis. Outlier Analysis 1–34. Cham, Switzerland: Springer. https://link.springer.com/book/10.1007/978-3-319-47578-3

American Institute of Certified Public Accountants (AICPA). 2023. AU-C 240: Consideration of Fraud in a Financial Statement Audit. Durham, NC: AICPA. https://pcaobus.org/oversight/standards/auditing-standards/details/AS2401

Arya, A., J. C. Fellingham, B. Mittendorf, and D. A. Schroeder. 2004. Reconciling financial information at varied levels of aggregation. *Contemporary Accounting Research* 21 (2): 303–324. https://doi.org/10.1506/A0H4-7XMD-DL60-37C5

Arya, A., J. C. Fellingham, J. C. Glover, D. A. Schroeder, and G. Strang. 2000. Inferring transactions from financial statements. *Contemporary Accounting Research* 17 (3): 366–385. https://doi.org/10.1506/L0LW-NX5L-4WUR-9JKL

Boersma, M., S. Sourabh, L. Hoogduin, and D. Kandhai. 2018. Financial statement networks: An application of network theory in audit. *Journal of Network Theory in Finance* 4 (4): 59–85. https://www.risk.net/journal-of-network-theory-in-finance/6202426/financial-statement-networks-an-application-of-network-theory-in-audit

Boersma, M., J. Wolsink, S. Sourabh, L. A. Hoogduin, and D. Kandhai. 2023. Measure cross-sectoral structural similarities from financial networks. *Scientific Reports* 13 (1). https://doi.org/10.1038/s41598-023-34034-w

Breunig, M. M., H. P. Kriegel, R. T. Ng, and J. Sander. 2000. LOF: Identifying density-based local outliers. *ACM SIGMOD Record* 29 (2): 93–104. https://doi.org/10.1145/335191.335388

Cho, S., M. A. Vasarhelyi, T. Sun, and C. Zhang. 2020. Learning from machine learning in accounting and assurance. *Journal of Emerging Technologies in Accounting*17 (1): 1–10. https://doi.org/10.2308/jeta-10718

Davenport, T. H. 1998. Putting the enterprise into the enterprise system. *Harvard Business Review* 76 (4): 121–131.

Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2010. The PASCAL visual object classes (VOC) challenge. *International Journal of Computer Vision* 88 (2): 303–338. https://doi.org/10.1007/s11263-009-0275-4

Fey, M., and J. E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. https://arxiv.org/abs/1903.02428

Fulcer, K., H. Gu, H. Hu, Q. Huang, A. Kogan, M. A. Vasarhelyi, D. Wei, and J. Young. 2025. Application of outlier detection methods in audit analytics. *Accounting Horizons* 39 (3): 143–157. https://doi.org/10.2308/HORIZONS-2023-071

American Accounting Association

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

Goldstein, M., and A. Dengel. 2012. *Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm*. Proceedings of the 35th Annual German Conference on AI, Saarbrücken, Germany, September 24–27. https://www.researchgate.net/profile/Markus-Goldstein/publication/231614824_Histogram-based_Outlier_Score_HBOS_A_fast_Unsupervised_Anomaly_Detection_Algorithm/links/02bfe50eeeeb75d0d3000000/Histogram-based-Outlier-Score-HBOS-A-fast-Unsupervised-Anomaly-Detection-Algorithm.pdf

Gray, G. L., and R. S. Debreceny. 2014. A taxonomy to guide research on the application of data mining to fraud detection in financial statement audits. *International Journal of Accounting Information Systems* 15 (4): 357–380. https://doi.org/10.1016/j.accinf.2014.05.006

Guo, K. H., X. Yu, and C. Wilkin. 2022. A picture is worth a thousand journal entries: Accounting graph topology for auditing and fraud detection. *Journal of Information Systems* 36 (2): 53–81. https://doi.org/10.2308/ISYS-2021-003

Hamilton, W. L. 2020. Graph Representation Learning. Cham, Switzerland: Springer. https://link.springer.com/book/10.1007/978-3-031-01588-5

Hamilton, W. L., R. Ying, and J. Leskovec. 2017. Representation learning on graphs: Methods and applications. https://arxiv.org/abs/1709.05584

Hemati, H., M. Schreyer, and D. Borth. 2021. Continual learning for unsupervised anomaly detection in continuous auditing of financial accounting data. https://arxiv.org/abs/2112.13215

Huang, Q. 2024. Three essays on anomaly detection in accounting: Channel stuffing, journal entry testing, and graph learning. Ph.D. dissertation, Rutgers University. https://doi.org/doi:10.7282/t3-4bz6-ez62

Huang, Q., M. Schreyer, N. Romero Michiles, and M. A. Vasarhelyi. 2024. A graph says more than a thousand journal entries harnessing graph autoencoder networks in auditing. *Expert Focus* (12): 653–659. https://www.efk.admin.ch/wp-content/uploads/publikationen/fachtexte/2024_12_a_graph_says_more_than_a_thousand_journal_entries.pdf

International Auditing and Assurance Standards Board (IAASB). 2009. International Standard on Auditing (ISA) 240: The auditor's responsibility to consider fraud in an audit of financial statements. https://www.iaasb.org/consultations-projects/auditor-s-responsibility-consider-fraud-audit-financial-statements-isa-240

Jans, M., M. G. Alles, and M. A. Vasarhelyi. 2014. A field study on the use of process mining of event logs as an analytical procedure in auditing. *Accounting Review* 89 (5). https://doi.org/10.2308/accr-50807

Kipf, T. N., and M. Welling. 2016. Variational graph auto-encoders. https://arxiv.org/abs/1611.07308

Kipf, T. N., and M. Welling. 2017. Semi-supervised classification with graph convolutional networks. https://arxiv.org/abs/1609.02907

Lenderink, R. J. 2019. Unsupervised outlier detection in financial statement audits. Master's thesis, University of Twente. https://www.scribd.com/document/465192894/Lenderink-MA-EEMCS

Liu, F. T., K. M. Ting, and Z. H. Zhou. 2008. *Isolation Forest*. Proceedings of the Eighth IEEE International Conference on Data Mining, Pisa, Italy, December 15–19. https://doi.org/10.1109/ICDM.2008.17

McGlohon, M., S. Bay, M. G. Anderle, D. M. Steier, and C. Faloutsos. 2009. *SNARE: A link analytic system for graph labeling and risk detection*. Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28–July 1. https://doi.org/10.1145/1557019.155715

Nguyen, H. T., P. J. Liang, and L. Akoglu. 2023. Detecting anomalous graphs in labeled multi-graph databases. *ACM Transactions on Knowledge Discovery from Data* 17 (2): 1–25. https://doi.org/10.1145/3533770

Nonnenmacher, J., F. Kruse, G. Schumann, and J. M. Gómez. 2021. *Using autoencoders for data-driven analysis in internal auditing*. Proceedings of the Annual Hawaii International Conference on System Sciences, Kauai, Hawaii, January 4–9. https://doi.org/10.24251/HICSS.2021.697

Public Company Accounting Oversight Board (PCAOB). 2020. AS 2401: Consideration of Fraud in a Financial Statement Audit. Washington, DC: PCAOB. https://pcaobus.org/oversight/standards/auditing-standards/details/AS2401.

Sangster, A. 2007. The printing of Pacioli's Summa in 1494: How many copies were printed? *Accounting Historians Journal* 34 (1): 125–145. https://doi.org/10.2308/0148-4184.34.1.125

Schreyer, M. 2023. Representation learning in financial auditing using deep neural networks. Ph.D. dissertation, St. Gallen University. https://slsp-hsg.primo.exlibrisgroup.com/discovery/delivery/41SLSP_HSG:HSGswisscovery/12107937600005506

Schreyer, M., T. Sattarov, D. Borth, A. Dengel, and B. Reimer. 2017. Detection of anomalies in large scale accounting data using deep autoencoder networks. https://arxiv.org/abs/1709.05254

Schreyer, M., T. Sattarov, A. Gierbl, B. Reimer, and D. Borth. 2020. *Learning sampling in financial statement audits using vector quantised variational autoencoder neural networks*. ICAIF 2020 - 1st ACM International Conference on AI in Finance. New York, NY: October 15–16. https://doi.org/10.1145/3383455.3422546

Schreyer, M., T. Sattarov, C. Schulze, B. Reimer, and D. Borth. 2019. Detection of accounting anomalies in the latent space using adversarial autoencoder neural networks. https://arxiv.org/abs/1908.00734

Schultz, M., and M. Tropmann-Frick. 2020. *Autoencoder neural networks versus external auditors: Detecting unusual journal entries in financial statement audits*. Proceedings of the Annual Hawaii International Conference on System Sciences. Maui, Hawaii, January 6–10. https://doi.org/10.24251/HICSS.2020.666

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

Shyu, M. L., S. C. Chen, K. Sarinnapakorn, and L. Chang. 2003. *A novel anomaly detection scheme based on principal component classifier*. 3rd IEEE International Conference on Data Mining, Melbourne, FL, November 19–22. https://apps.dtic.mil/sti/tr/pdf/ADA465712.pdf

Sun, T. S. 2019. Applying deep learning to audit procedures: An illustrative framework. *Accounting Horizons* 33 (3): 89–109. https://doi.org/10.2308/acch-52455

West, D. B. 2001. *Introduction to graph theory*. Vol. 2. Upper Saddle River, NJ: Prentice Hall. https://daiwz.net/course/disc_math/2023/West_Intro_Graph_Theory_en.pdf

Xia, F., K. Sun, S. Yu, A. Aziz, L. Wan, S. Pan, and H. Liu. 2021. Graph learning: A survey. *IEEE Transactions on Artificial Intelligence* 2 (2): 109–127. https://doi.org/10.1109/TAI.2021.3076021

Yadav, D., S. Zhang, and T. Jin. 2023. *Transformer based anomaly detection on multivariate time series subledger data*. Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, August 6–10, Long Beach, CA. https://www.amazon.science/publications/transformer-based-anomaly-detection-on-multivariate-time-series-subledger-data

Zhang, Z., P. Cui, and W. Zhu. 2022. Deep learning on graphs: A survey. *IEEE Transactions on Knowledge and Data Engineering* 34 (1): 249–270. https://doi.org/10.1109/TKDE.2020.2981333

Župan, M., S. Letinić, and V. Budimir. 2018. Journal entries with deep learning model. *International Journal of Advance Computational Engineering and Networking (IJACEN)* 6 (10). https://repozitorij.vup.ftrr.hr/object/vup:1277

Zupan, M., S. Letinic, and V. Budimir. 2020. *Accounting journal reconstruction with variational autoencoders and long short-term memory architecture*. CEUR Workshop Proceedings, Sud Sardegna, Italy, June 21–24. https://ceur-ws.org/Vol-2646/05-paper.pdf

APPENDIX A

Parameter Tuning for Graph Autoencoder Neural Network Architecture

Architecture influences how the neural network learns. In this experiment, we evaluate the performance of multiple architectures, each trained for 500 iterations. Table A1, Panels A–D present the results for Dataset A on all anomalies, with averages and standard deviations calculated from the best five seeds. The columns for Best Iteration and Best_AP refer to the highest AP recorded at each 100-iteration-level interval, whereas the remaining metrics are reported at the 500th iteration. Figures A1–A3 visualize the average of highest average precision achieved by each architecture across different seeds for all anomalies, global anomalies, and local anomalies, respectively. As shown in Table A1, Panels A–D and Figures A1–A3, performance initially improves as the number of layers increases but starts to decline beyond a certain depth. Overall, a six-layer architecture produces the best results for all methods. Therefore, we select a six-layer configuration, with the encoder set to 128, 64, 32, 16, 8, 2 and the decoder set to 2, 8, 16, 32, 64, 128.

(continued on next page)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX A (continued)

TABLE A1

Architecture Tuning Table

Panel A: Architecture Tuning 8 Layer
Encoder: 512, 256, 128, 64, 32, 16, 8, 2
Decoder: 2, 8, 16, 32, 64, 128, 256, 512

	GAEN + RE							GAEN + LOF						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	100	0.0293	0.0254	0.9803	0.0600	0.0600	0.0600	500	0.1284	0.1284	0.9801	0.0452	0.0450	0.0451
2222	100	0.0167	0.0161	0.9795	0.0200	0.0200	0.0200	400	0.1466	0.1426	0.9803	0.0644	0.0650	0.0647
3333	100	0.0284	0.0189	0.9786	0.0250	0.0250	0.0250	100	0.1051	0.0806	0.9797	0.0300	0.0300	0.0300
4444	100	0.0251	0.0238	0.9799	0.0400	0.0400	0.0400	200	0.1129	0.0992	0.9795	0.0200	0.0200	0.0200
5555	500	0.0134	0.0134	0.9795	0.0200	0.0200	0.0200	100	0.1161	0.0658	0.9796	0.0201	0.0200	0.0201
6666	100	0.0379	0.0294	0.9807	0.0750	0.0750	0.0750	100	0.0655	0.0440	0.9797	0.0203	0.0200	0.0202
7777	300	0.0197	0.0188	0.9806	0.0700	0.0700	0.0700	200	0.0995	0.0859	0.9797	0.0300	0.0300	0.0300
Mean	—	0.0281	0.0232	0.9802	0.0540	0.0540	0.0540	—	0.1218	0.1073	0.9799	0.0380	0.0380	0.0380
Std. Dev.	—	0.0067	0.0045	0.0005	0.0210	0.0210	0.0210	—	0.0162	0.0270	0.0003	0.0172	0.0175	0.0174

	GAEN + IForest						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	100	0.1067	0.0484	0.9810	0.0941	0.0950	0.0945
2222	100	0.0959	0.0657	0.9830	0.1850	0.1850	0.1850
3333	100	0.0815	0.0243	0.9800	0.0495	0.0500	0.0498
4444	100	0.1388	0.1141	0.9835	0.2100	0.2100	0.2100
5555	500	0.0404	0.0404	0.9806	0.0746	0.0750	0.0748
6666	100	0.1242	0.1187	0.9847	0.2700	0.2700	0.2700
7777	500	0.0592	0.0592	0.9807	0.0750	0.0750	0.0750
Mean	—	0.1094	0.0812	0.9826	0.1668	0.1670	0.1669
Std. Dev.	—	0.0227	0.0327	0.0017	0.0815	0.0813	0.0814

(continued on next page)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/1561133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX A (continued)

TABLE A1 (continued)

Panel B: Architecture Tuning 6 Layer
Encoder: 128, 64, 32, 16, 8, 2
Decoder: 2, 8, 16, 32, 64, 128

	GAEN + RE							GAEN + LOF						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	200	0.0320	0.0312	0.9809	0.0850	0.0850	0.0850	300	0.0983	0.0966	0.9794	0.0150	0.0150	0.0150
2222	200	0.0190	0.0176	0.9800	0.0450	0.0450	0.0450	200	0.1437	0.1185	0.9803	0.0600	0.0600	0.0600
3333	100	0.0262	0.0168	0.9796	0.0250	0.0250	0.0250	200	0.0681	0.0592	0.9800	0.0450	0.0450	0.0450
4444	500	0.0189	0.0189	0.9796	0.0250	0.0250	0.0250	200	0.1474	0.1277	0.9805	0.0697	0.0700	0.0698
5555	500	0.0131	0.0131	0.9795	0.0200	0.0200	0.0200	100	0.1043	0.0985	0.9801	0.0631	0.0650	0.0640
6666	100	0.0444	0.0336	0.9812	0.1122	0.1150	0.1136	100	0.0466	0.0462	0.9797	0.0300	0.0300	0.0300
7777	100	0.0295	0.0268	0.9810	0.0905	0.0900	0.0902	100	0.1127	0.1038	0.9792	0.0194	0.0200	0.0197
Mean	—	0.0302	0.0256	0.9806	0.0715	0.0720	0.0718	—	0.1213	0.1090	0.9802	0.0536	0.0540	0.0538
Std. Dev.	—	0.0093	0.0071	0.0007	0.0356	0.0363	0.0359	—	0.0228	0.0135	0.0003	0.0160	0.0164	0.0162

	GAEN + IForest						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	100	0.1111	0.0977	0.9851	0.2864	0.2850	0.2857
2222	100	0.0934	0.0745	0.9818	0.1337	0.1350	0.1343
3333	300	0.1099	0.1017	0.9840	0.2312	0.2300	0.2306
4444	100	0.1180	0.1099	0.9844	0.2513	0.2500	0.2506
5555	300	0.0514	0.0488	0.9812	0.0960	0.0950	0.0955
6666	500	0.1196	0.1196	0.9842	0.2450	0.2450	0.2450
7777	200	0.0778	0.0721	0.9824	0.1414	0.1350	0.1381
Mean	—	0.1104	0.1007	0.9840	0.2310	0.2290	0.2300
Std. Dev.	—	0.0104	0.0169	0.0010	0.0541	0.0563	0.0552

(continued on next page)

American Accounting Association

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/1561133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX A (continued)

TABLE A1 (continued)

Panel C: Architecture Tuning 4 Layer
Encoder: 32, 16, 8, 2
Decoder: 2, 8, 16, 32

	GAEN + RE							GAEN + LOF						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	500	0.0328	0.0328	0.9808	0.0800	0.0800	0.0800	200	0.1307	0.1226	0.9800	0.0450	0.0450	0.0450
2222	500	0.0438	0.0438	0.9822	0.1500	0.1500	0.1500	100	0.1051	0.0846	0.9797	0.0251	0.0250	0.0251
3333	500	0.0239	0.0239	0.9798	0.0303	0.0300	0.0302	200	0.0687	0.0528	0.9802	0.0410	0.0400	0.0405
4444	100	0.0118	0.0117	0.9793	0.0100	0.0100	0.0100	100	0.1235	0.1228	0.9794	0.0150	0.0150	0.0150
5555	400	0.0127	0.0127	0.9796	0.0250	0.0250	0.0250	100	0.0808	0.0779	0.9794	0.0150	0.0150	0.0150
6666	200	0.0777	0.0753	0.9837	0.2200	0.2200	0.2200	100	0.0877	0.0779	0.9796	0.0202	0.0200	0.0201
7777	400	0.0173	0.0172	0.9806	0.0700	0.0700	0.0700	200	0.1349	0.1128	0.9801	0.0500	0.0500	0.0500
Mean	—	0.0391	0.0386	0.9814	0.1101	0.1100	0.1100	—	0.1164	0.1041	0.9799	0.0363	0.0360	0.0361
Std. Dev.	—	0.0238	0.0228	0.0015	0.0751	0.0752	0.0751	—	0.0197	0.0214	0.0003	0.0129	0.0129	0.0129

	GAEN + IForest						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	500	0.0501	0.0501	0.9810	0.0900	0.0900	0.0900
2222	400	0.0998	0.0971	0.9837	0.2255	0.2300	0.2277
3333	500	0.0796	0.0796	0.9827	0.1709	0.1700	0.1704
4444	400	0.0576	0.0556	0.9825	0.1650	0.1650	0.1650
5555	300	0.0553	0.0538	0.9820	0.1357	0.1350	0.1353
6666	100	0.1278	0.1179	0.9838	0.2250	0.2250	0.2250
7777	500	0.0697	0.0697	0.9826	0.1700	0.1700	0.1700
Mean	—	0.0869	0.0840	0.9831	0.1913	0.1920	0.1916
Std. Dev.	—	0.0276	0.0242	0.0006	0.0311	0.0325	0.0318

(continued on next page)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/1561133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX A (continued)

TABLE A1 (continued)

Panel D: Architecture Tuning 2 Layer
Encoder: 8, 2
Decoder: 2, 8

	GAEN + RE							GAEN + LOF						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	500	0.0777	0.0777	0.9839	0.2300	0.2300	0.2300	100	0.0652	0.0554	0.9792	0.0050	0.0050	0.0050
2222	500	0.0170	0.0170	0.9801	0.0628	0.0650	0.0639	100	0.0660	0.0634	0.9799	0.0443	0.0450	0.0447
3333	100	0.0310	0.0263	0.9801	0.0500	0.0500	0.0500	100	0.0911	0.0794	0.9799	0.0400	0.0400	0.0400
4444	200	0.0154	0.0153	0.9793	0.0100	0.0100	0.0100	200	0.0989	0.0915	0.9799	0.0400	0.0400	0.0400
5555	500	0.0129	0.0129	0.9796	0.0250	0.0250	0.0250	300	0.0908	0.0875	0.9797	0.0300	0.0300	0.0300
6666	100	0.0772	0.0663	0.9830	0.1850	0.1850	0.1850	100	0.0659	0.0639	0.9798	0.0350	0.0350	0.0350
7777	500	0.0224	0.0224	0.9810	0.0900	0.0900	0.0900	200	0.1205	0.1151	0.9801	0.0500	0.0500	0.0500
Mean	—	0.0451	0.0420	0.9816	0.1236	0.1240	0.1238	—	0.0934	0.0875	0.9799	0.0419	0.0420	0.0419
Std. Dev.	—	0.0300	0.0279	0.0017	0.0796	0.0792	0.0794	—	0.0195	0.0187	0.0001	0.0056	0.0057	0.0057

	GAEN + IForest						
Seed	**Best Iteration**	**Best_AP**	**AP**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
1111	200	0.1492	0.1344	0.9839	0.2300	0.2300	0.2300
2222	300	0.0357	0.0344	0.9812	0.1005	0.1000	0.1003
3333	500	0.0443	0.0443	0.9815	0.1150	0.1150	0.1150
4444	100	0.0394	0.0317	0.9796	0.0474	0.0500	0.0487
5555	400	0.0297	0.0294	0.9807	0.0750	0.0750	0.0750
6666	100	0.1904	0.1739	0.9838	0.2250	0.2250	0.2250
7777	300	0.0702	0.0675	0.9826	0.1700	0.1700	0.1700
Mean	—	0.0987	0.0909	0.9826	0.1681	0.1680	0.1681
Std. Dev.	—	0.0676	0.0606	0.0012	0.0601	0.0603	0.0602

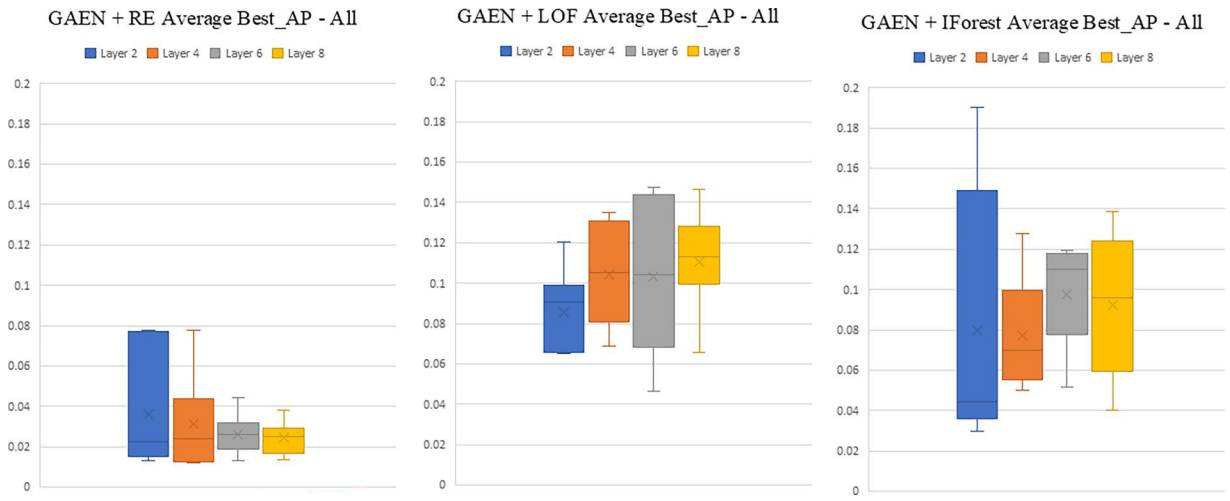
American Accounting Association

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/1561133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX A (continued)

FIGURE A1

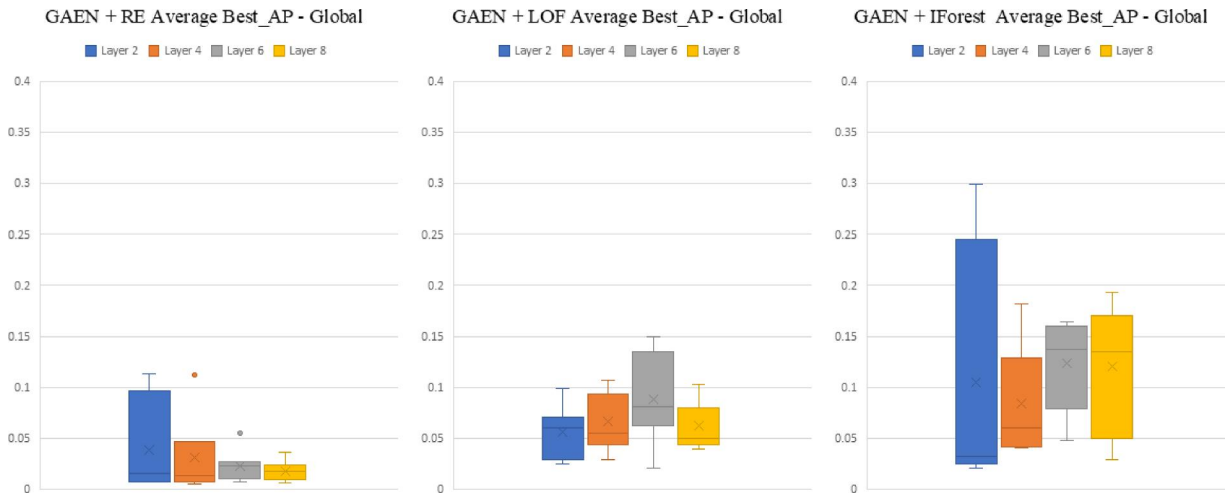
Average Best_AP over Architecture—All Anomaly



(The full-color version is available online.)

FIGURE A2

Average Best_AP over Architecture—Global Anomaly



(The full-color version is available online.)

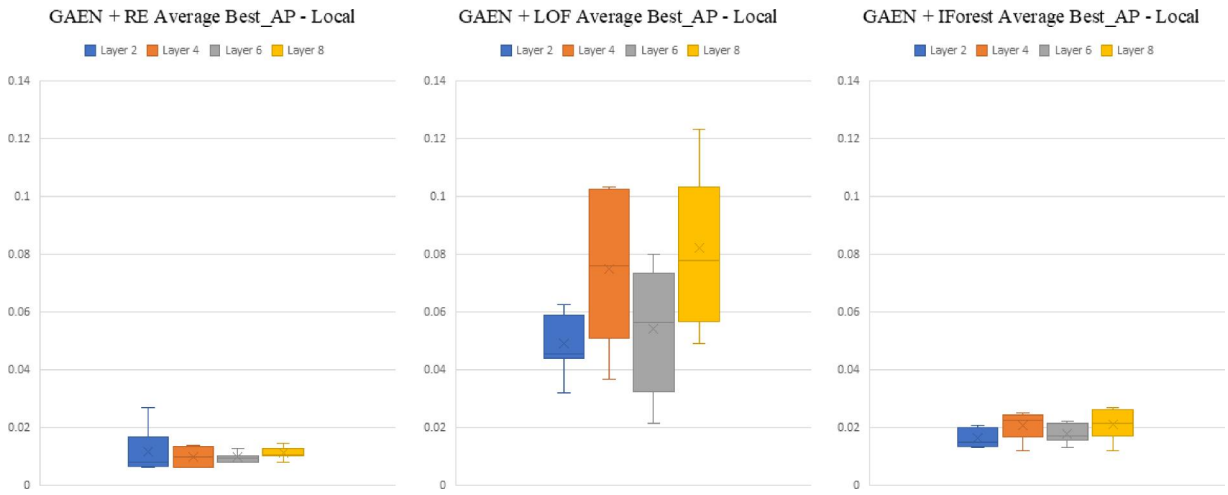
(continued on next page)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX A (continued)

FIGURE A3

Average Best_AP over Architecture—Local Anomaly



(The full-color version is available online.)

APPENDIX B

Parameter Tuning for Graph Autoencoder Neural Network Learning Rate and Batch Size

Learning rate is another important parameter affecting performance. We tested four values, $\eta \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$, and evaluated the Best_AP across different batch sizes. Figure B1 shows the average Best_AP for each learning rate across various batch size for different anomaly types. 10^{-3} produces the highest average Best_AP for all methods and across all types of anomaly and was therefore selected as the final learning rate.

Batch size ρ is also swept through $\rho \in \{32, 64, 128, 256\}$. Figure B2 displays the mean Best_AP across different learning rates for all anomaly types. The average Best_APs remain relatively low but stable for GAEN + RE. Although the performance of GAEN + IForest improves at batch size of 256, others dip or remain inconsistent. Hence, batch size 64 is chosen as a middle ground that generates solid and stable performance across different anomaly categories.

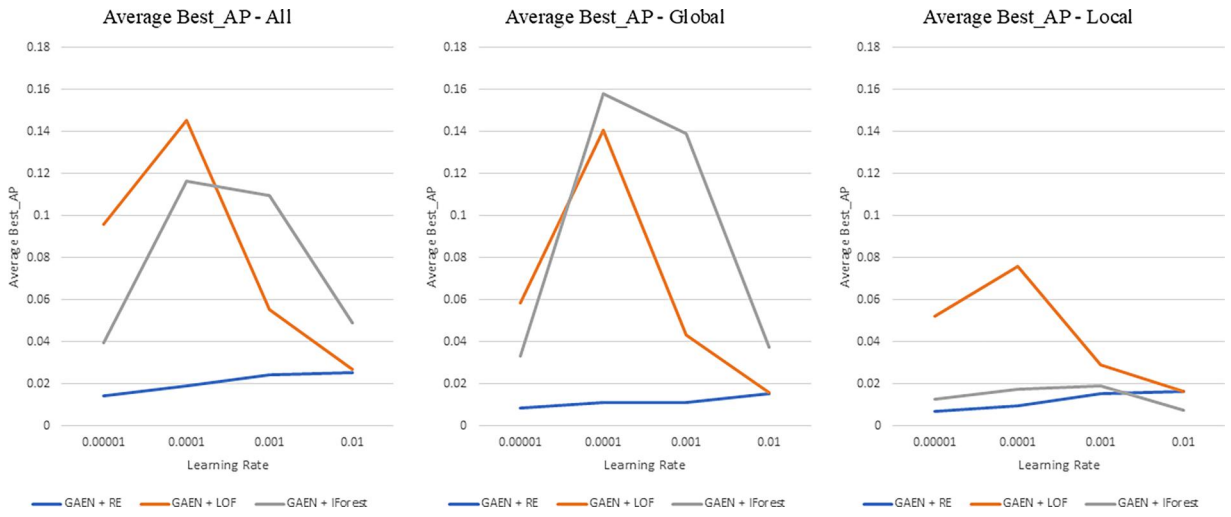
(continued on next page)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX B (continued)

FIGURE B1

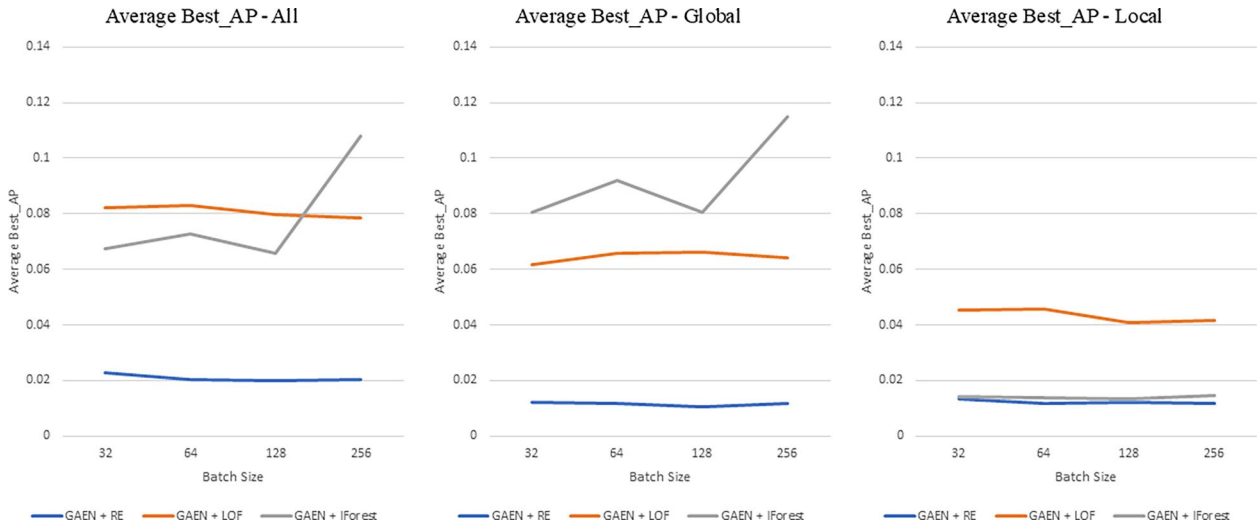
Average Best_AP over Learning Rate



(The full-color version is available online.)

FIGURE B2

Average Best_AP over Batch Size



(The full-color version is available online.)

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

APPENDIX C

Hyperparameters for Each Method in Dataset A

Algorithm	Hyperparameters
IForest, GAEN + IForest	n_estimator = [50, 100, 150]; max_samples = ['auto', 0.6,0.8]
LOF, GAEN + LOF	n_neighbors = [50, 100, 150]; leaf_size = [10, 20, 30]
HBOS, GAEN + HBOS	n_bins = [50, 100, 150]; tol = [0.1, 0.3, 0.5]
PCA	n_components = [1, 10, 80, 140]; svd_solver = ["auto", "full", 'randomized']
AutoEncoder	hidden_neurons = [[512, 256, 128, 64, 32, 16, 8, 2, 2, 8, 16, 32, 64, 128, 256, 512], [128, 64, 32, 16, 8, 2, 8, 16, 32, 64, 128], [8, 2, 2, 8]]; epochs = [100, 200, 500]
GAEN + PCA	n_components = [1, 2]; svd_solver = ["auto", "full", 'randomized']

Downloaded from http://publications.aaahq.org/ajpt/article-pdf/doi/10.2308/AJPT-2024-058/156133/ajpt-2024-058.pdf by guest on 09 March 2026

American Accounting Association