Decentralized Collaborative Learning Framework with External Privacy Leakage Analysis

Tsuyoshi Idé[1], Dzung T. Phan[1], and Rudy Raymond[2]

[1]*IBM Thomas J. Watson Research Center.*
[2] *Global Technology and Applied Research, J.P. Morgan Chase & Co.*

E-mail: {tide,phandu}@us.ibm.com, raymond.putra@jpmchase.com

(Received February 4, 2024)

This paper presents two methodological advancements in decentralized multi-task learning under privacy constraints, aiming to pave the way for future developments in next-generation Blockchain platforms. First, we expand the existing framework for collaborative dictionary learning (CollabDict), which has previously been limited to Gaussian mixture models, by incorporating deep variational autoencoders (VAEs) into the framework, with a particular focus on anomaly detection. We demonstrate that the VAE-based anomaly score function shares the same mathematical structure as the non-deep model, and provide comprehensive qualitative comparison. Second, considering the widespread use of "pre-trained models," we provide a mathematical analysis on data privacy leakage when models trained with CollabDict are shared externally. We show that the CollabDict approach, when applied to Gaussian mixtures, adheres to a Rényi differential privacy criterion. Additionally, we propose a practical metric for monitoring internal privacy breaches during the learning process.

KEYWORDS: Blockchain, collaborative learning, multi-task learning, anomaly detection, variational autoencoder

1. Introduction

The design principles of Blockchain, including decentralization, security, and transparency in transaction management, have had a profound impact on business and industrial applications. While Blockchain technology was initially proposed as a self-governing platform for currency exchange [28], it was later expanded to include general business transactions in the form of smart contracts [8]. However, in many application scenarios, such as product traceability systems, Blockchain has primarily been used as an immutable data storage system where stored data is never transformed in any way. We believe that the true value of Blockchain lies in its potential for value co-creation through knowledge sharing. For example, if car dealers collaboratively store pre-owned car repair records, they may wish to have an anomaly detection (or diagnosis) model trained on the collected data with a machine learning algorithm. This involves the process of sharing and transforming data. We envision that the next generation of Blockchain will seamlessly integrate machine learning algorithms, enabling collaborative learning functionalities.

From a machine learning perspective, collaborative learning can be formalized as decentralized multi-task learning under a data privacy constraint. Figure 1 illustrates the problem setting. The network consists of S participant members ($S > 2$) connected to each other through a sparse peer-to-peer (P2P) network. Each participant, indexed with s, has its own dataset $\mathcal{D}^s = \{\boldsymbol{x}^{s(1)}, \ldots, \boldsymbol{x}^{s(N^s)}\}$, where $\boldsymbol{x}^{s(n)} \in \mathcal{D}^s$ represents the n-th sample of the s-th participant and N^s is the number of samples in $\mathcal{D}^s$. The participants' objective is to develop their own machine learning models. Since this involves learning S distinct models simultaneously, it is referred to as *multi-task learning* in the machine learning literature.

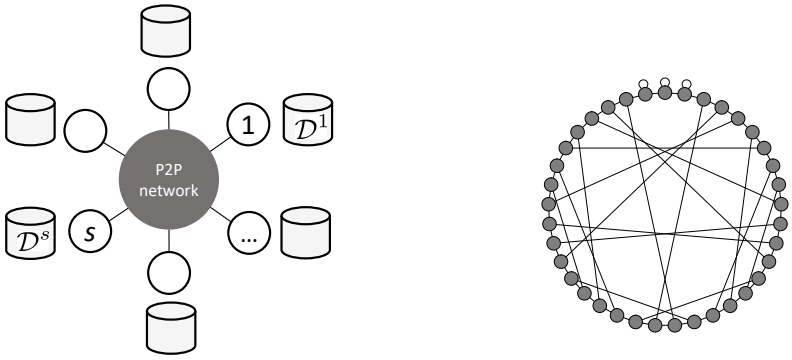

Fig. 1. Illustration of the problem setting of multi-task learning under the privacy and decentralization constraints. *Left*: Each network participant ($s = 1, \ldots, S$) holds its own dataset $\mathcal{D}^s$ privately and builds its own machine learning model with the hope that other participants' data would help improve the model. *Right*: An example of a peer-to-peer (P2P) communication network. A 3-regular expander graph called the cycle with inverse chord is shown for $S = 31$ participants.

While multi-task learning under decentralized and privacy-preservation constraints is an interesting extension of the traditional machine learning paradigm, meeting both constraints is generally challenging. This is because these constraints work in opposite directions: Decentralization implies "sharing with anyone," whereas privacy implies "*not* sharing with anyone." CollabDict (collaborative dictionary learning) [13, 16] is the first framework that successfully addresses both. However, there are two major limitations. *First*, the framework was specifically derived for the sparse Gaussian graphical model (GGM) mixtures, and it is not clear whether it is applicable to more general settings, such as those using deep learning. *Second*, while a previous study [15] analyzes the risk of privacy breach within the network in the training phase (i.e., internal privacy leakage), little is known about external privacy leakage when the trained model is shared with a third party as a pre-trained model.

In this paper, we introduce a novel collaborative learning framework, with a particular focus on unsupervised anomaly detection, based on a multi-task extension of the variational autoencoder (VAE) [17]. Unlike traditional autoencoders (AEs), VAEs are capable of providing a predictive probability distribution of the observed variables. This allows us to quantify the range of normalcy in a principled way, making VAEs more useful than AEs in practice. While VAEs have been extensively used in anomaly detection [1, 6, 39], the multi-task extension under privacy and decentralized constraints has been considered challenging. We show that the proposed multi-task VAE can naturally fit the CollabDict framework. Interestingly, we will point out that the new VAE-based approach shares the same mixture representation for the anomaly score function, demonstrating the general applicability of the original concept of CollabDict. Encouraged by this fact, we provide a theoretical analysis of external privacy leakage under the GGM setting with the aid of the Rényi differential privacy theory [26]. To the best of our knowledge, this is the first work that provides a theoretical guarantee on external privacy leakage under the decentralized setting.

To summarize, our contributions are twofold:

- We propose a novel VAE-based multi-task anomaly detection framework under decentralized and data privacy constraints.

- We provide the first mathematical guarantee of external data leakage when a model trained on CollabDict is used externally by a third party.

2. Related Work

Most of the applications of Blockchain in domains involving real-valued noisy data are related to product traceability [21, 35, 36]. There are few studies in the literature on collaborative learning. Recently, Münsing et al. proposed a Blockchain-based optimization framework for efficiently scheduling a micropower grid [27]. This framework shares the concept of separating local and global variables with ours, but it is not applicable to multi-task anomaly detection and does not discuss privacy preservation. Other partially relevant studies include [38], which discusses privacy-preserving multi-task learning (MTL) in a distributed setting. However, it lacks the context of Blockchain, especially consensus building, and its mathematical treatment of privacy seems incomplete. Another recent paper [13] proposes an MTL-based fault detection framework, but it does not cover privacy preservation and consensus building.

Data privacy protection in the distributed learning setting has been actively studied in the field of federated learning. Chen et al. [7] studied privacy protection in federated learning and proposed an approach based on homomorphic encryption. While the proposed framework allows decentralized secret data exchange, handling the diversity of network participants and therefore multi-task learning is not within the main scope. Masurkar et al. [25] proposed another decentralized, privacy-preserving federated learning framework, but the aspects of multi-task learning and anomaly detection are not discussed. Mahmood et al. [24] address the risk of having malicious participants in privacy-preserving federated learning.

The work by Zhao et al. [45] may be closest to ours, which proposed a deep learning framework in the multi-task and distributed learning setting. The idea of global and local updates, which is shared by the CollabDict framework, is included there. However, it is only in the supervised setting, and the communication protocol relies on a central server.

Training deep learning models in a distributed setting is one of the recent popular topics in machine learning. In the context of VAE, for example, recent works include Polato [32] and Li et al. [20], which proposed federated VAE learning for collaborative filtering. Also, Zhao et al. [46] proposed a secure distributed VAE based on homomorphic encryption. Zhang et al. [44] and Huang et al. [12] addressed the task of anomaly detection from multivariate time-series data using VAE. These works are either not for multi-task learning or not in the decentralized setting. Very recently, much attention has been paid to multi-task reinforcement learning (MTRL) [30, 42], in which the goal is to learn a policy shared by multiple tasks, but standard anomaly detection tasks are typically not within their scope.

3. Problem Setting

We consider a scenario where S parties wish to collaboratively build anomaly detection models through communication on a P2P network. Depending on the communication protocol, the network can be dynamically created, but participants are assumed to know who their peers are during communication. Each network participant, indexed with $s \in \{1, \ldots, S\}$, has its own dataset $\mathcal{D}^s = \{\boldsymbol{x}^{s(1)}, \ldots, \boldsymbol{x}^{s(N^s)}\}$. Here, $\boldsymbol{x}^{s(n)}$ represents the n-th sample of the s-th dataset, and N^s is the number of samples in $\mathcal{D}^s$. It is possible for some participants to have more samples than others. To distinguish between a random variable and its realization, we attach an index specifying an instance. For example, $\boldsymbol{x}^s$ is a random variable representing participant s's data generation mechanism, while $\boldsymbol{x}^{s(n)}$ is a realization (i.e., a numerical vector). We refer to the participant's data generating mechanism as the *system*. We assume that all samples are in $\mathbb{R}^M$ (i.e., M-dimensional real-valued vectors) and follow the same data format. For instance, if the 8th dimension of the second participant's data, $x_8^{2(n)}$, is a temperature measurement in Celsius, then $x_8^{s(n)}$ represents temperatures in Celsius for any s, n.

We focus on an unsupervised learning scenario in anomaly detection. The observed samples

$\{\boldsymbol{x}^{s(n)}\}$ are assumed to be generally noisy. One standard definition of anomaly scores in such a situation is the logarithmic loss (See, e.g., [19, 29, 34, 41]):

$$a^s(\boldsymbol{x}^{s,\mathrm{test}}) = -\ln p^s(\boldsymbol{x}^{s,\mathrm{test}}), \tag{1}$$

which can be also viewed as the negative one-sample likelihood with respect to the predictive distribution. An unusually low likelihood is an indication of anomalousness, yielding a high anomaly score. Here, $p^s(\cdot)$ denotes the estimated probability density function (pdf) of the s-th system. Unlike straightforward metrics such as the reconstruction error [4, 5, 33], the log loss score can handle prediction uncertainty in a systematic manner. The definition of (1) is supported also from an information-theoretic perspective [40].

The task of unsupervised anomaly detection is now a density estimation problem. It is important to note that the pdf depends on s. Thus, the entire task is to learn S pdfs simultaneously. In machine learning, this setting is typically referred to as *multi-task learning*. A key concept in multi-task learning is task-relatedness. However, under the constraint of data privacy, where participants cannot directly share their local data with others, leveraging other participants' data for improving the models is not straightforward. The key research question here is how selfish participants can collaboratively build the model while maintaining data privacy.

4. The CollabDict Framework

CollabDict [13, 16] is an iterative procedure derived from the maximum a posterior (MAP) principle to perform multi-task learning under privacy and decentralized constraints. The algorithm aims to find model parameters of the pdf $p^s(\cdot)$ by maximizing the log marginalized likelihood. It is referred to as "dictionary learning" since it is designed to discover a pool of multiple patterns (i.e., a pattern dictionary) so that diversity over the participants' systems is properly captured. See Subsection 4.5 for further discussion.

In CollabDict, $p^s(\cdot)$ is generally parameterized as $p^s(\cdot \mid \boldsymbol{\phi}^s, \boldsymbol{\theta})$, where $\boldsymbol{\phi}^s$ is the set of local parameters that are privately held within the s-th participant and $\boldsymbol{\theta}$ is the set of global parameters that are shared by all participants. Algorithm 1 shows the overall procedure of CollabDict, where $\{\boldsymbol{\theta}^s\}$ are intermediate variables used to compute $\boldsymbol{\theta}$. At convergence, each participant ends up having $(\boldsymbol{\phi}^s, \boldsymbol{\theta})$ at hand as the final outcome. It consists of three main modules: `local_update`, `consensus`, and `optimize`. In these modules, `local_update` and `optimize` loosely correspond to the expectation and maximization in the EM (expectation-maximization) algorithm [3], respectively. The `consensus` module is unique to privacy-preserving decentralized computation and is discussed in the next sections. Following the previous work, we assume that all the participants are *honest but curious*, meaning that they do not lie about their data and computed statistics but they always try to selfishly get as much information as possible from the others.

4.1 Dynamical consensus algorithm

In the `consensus` module, there are two main technical problems. One is how to perform aggregation without a central coordinator. The other is how to eliminate data privacy breach for a given aggregation algorithm. This and the next subsections discuss these problems.

First, let us consider how aggregation is achieved without a central coordinator. Let $\mathsf{A} \in \{0, 1\}^{S \times S}$ be the adjacency matrix of a graph where nodes represent the network participants (or their systems) and edges represent (potentially bilateral) communication channels between the nodes. We assume that each participant knows their connected neighbors. Since the summation over tensors (vectors, matrices, etc.) can be performed element-wise, without loss of generality, we consider the problem

Algorithm 1 `CollabDict` (collaborative dictionary learning)

Initialize local and global model parameters.
Set hyper-parameters to the agreed-upon values.
repeat
  for $s \leftarrow 1, \ldots, S$ **do**
    $\phi^s, \theta^s \leftarrow \texttt{local\_update}(\phi^s, \theta)$
  end for
  $\theta \leftarrow \texttt{consensus}(\theta^1, \ldots, \theta^S)$
  for $s \leftarrow 1, \ldots, S$ **do**
    $\theta \leftarrow \texttt{optimize}(\theta)$
  end for
until convergence

of computing the average of scalars $\{\xi^s\}$:

$$\bar{\xi} = \frac{1}{S} \sum_{s=1}^{S} \xi^s. \tag{2}$$

For each s, the dynamical consensus algorithm generates a sequence $\xi^s(t)$ with $t = 0, 1, 2, \ldots$ and $\xi^s(0) = \xi^s$ such that $\xi^s(\infty)$ converges to $\bar{\xi}^s$. Upon going from t to $t + 1$, the participant receives the current value from the neighboring nodes and performs an update:

$$\xi^s(t+1) = \xi^s(t) + \frac{1}{S} \sum_{j=1}^{S} \mathsf{A}_{s,j}[\xi^j(t) - \xi^s(t)]. \tag{3}$$

In vector form, this can be written as

$$\boldsymbol{\xi}(t+1) = [\mathsf{I}_S - (1/S)\mathsf{L}]\, \boldsymbol{\xi}(t), \tag{4}$$

where $\boldsymbol{\xi}(t) \triangleq (\xi^1(t), \ldots, \xi^S(t))^\top$, I_S is the S-dimensional identity matrix, and $\mathsf{L} \triangleq \mathsf{D} - \mathsf{A}$ is the graph Laplacian. Here D is the degree matrix defined by $\mathsf{D}_{i,j} = \delta_{i,j} \sum_{s=1}^{S} \mathsf{A}_{i,s}$ with $\delta_{i,j}$ being Kronecker's delta. As can be seen, $\frac{1}{\sqrt{S}}\mathbf{1}_S$ is an ℓ_2-normalized eigenvector of $\mathsf{W} \triangleq \mathsf{I}_S - (1/S)\mathsf{L}$ with an eigenvalue of 1. If A has been properly chosen such that the eigenvalue 1 is unique and the other absolute eigenvalues are less than 1, Eq. (4) will converge to the stationary solution $\boldsymbol{\xi}^*$

$$\boldsymbol{\xi}^* = \left(1 \cdot \frac{1}{\sqrt{S}}\mathbf{1}_S \frac{1}{\sqrt{S}}\mathbf{1}_S^\top\right)\boldsymbol{\xi}(0) = \mathbf{1}_S \frac{1}{S} \sum_{s=1}^{S} \xi^s = \bar{\xi}\mathbf{1}_S. \tag{5}$$

This implies that *all participants have the same value of* $\bar{\xi}$ *upon convergence*, achieving average consensus. The distribution of eigenvalues of A greatly affects the convergence speed. Idé and Raymond [15] demonstrated that the number of iterations until convergence scales with $\ln S$ if the network satisfies the definition of an expander graph. This is a remarkably fast convergence rate. See Fig. 1 (right) for an example of an expander graph.

4.2 Random chunking for data privacy protection

Next, let us consider how data privacy is protected in the dynamical consensus algorithm. One obvious issue with the above algorithm is that the connected peers can see the original value in the first iteration, which is a breach of data privacy. There are two known solutions to address this issue [15].

One is Shamir's secret sharing and the other is random chunking. Here, we will outline the latter. For details of Shamir's secret sharing, please refer to Reference [15].

The idea behind random chunking is simple. Before running the consensus algorithm, the participants divide the data ξ^s into C chunks $\xi^{s[1]}, \ldots, \xi^{s[C]}$ in any arbitrary manner, as long as $\sum_{l=1}^{C} \xi^{s[l]} = \xi^s$. Then, they independently run the consensus algorithm C times, each time for a different chunk, to obtain C averages $\bar{\xi}^{[1]}, \ldots, \bar{\xi}^{[C]}$. Since aggregation is a linear operation, we have:

$$\bar{\xi} = \bar{\xi}^{[1]} + \cdots + \bar{\xi}^{[C]}. \tag{6}$$

For better privacy protection, it is advisable to randomly change the node labels for each aggregation session so that each participant has a different set of connected peers every time.

4.3 Sparse Gaussian graphical model mixture: Model definition

As a concrete example, we provide an algorithm for a sparse Gaussian graphical model (GGM) mixture, which has been extensively discussed in our previous work [16]. In this model, the data observation model of system s is given by

$$p(\boldsymbol{x}^s \mid \boldsymbol{z}^s, \boldsymbol{\mu}, \Lambda) = \prod_{k=1}^{K} \mathcal{N}(\boldsymbol{x}^s \mid \boldsymbol{\mu}_k, (\Lambda_k)^{-1})^{z_k^s}. \tag{7}$$

Here, $\mathcal{N}(\cdot \mid \boldsymbol{\mu}_k, (\Lambda_k)^{-1})$ denotes a Gaussian distribution with the mean vector $\boldsymbol{\mu}_k \in \mathbb{R}^M$ and the precision matrix $\Lambda_k \in \mathbb{R}^{M \times M}$. On the l.h.s., we used the collective notations $\boldsymbol{\mu}$ and Λ representing $\{\boldsymbol{\mu}_k\}$ and $\{\Lambda_k\}$, respectively. In this model, one "pattern" is represented by a Gaussian distribution with $(\boldsymbol{\mu}_k, \Lambda_k)$. Since it is unobserved which pattern a sample $\boldsymbol{x}^s$ belongs to, we need a latent variable representing pattern selection. $\boldsymbol{z}^s$ is the indicator variable over K different patterns with $z_k^s \in \{0, 1\}$ for all s and $\sum_{k=1}^{K} z_k^s = 1$. The total number of patterns, K, must be provided upfront as a hyper-parameter. However, one can automatically determine an appropriate number starting from a sufficiently large K by using the cardinality regularization technique. See Refs. [14, 31] for technical details.

For probabilistic inference, prior distributions are assigned to $\boldsymbol{\mu}_k, \Lambda_k, \boldsymbol{z}^s$:

$$p(\boldsymbol{\mu}_k, \Lambda_k) \propto \mathcal{N}(\boldsymbol{\mu}_k \mid \boldsymbol{m}_0, (\lambda_0 \Lambda_k)^{-1}) \exp\left(-\frac{\rho}{2} \|\Lambda_k\|_1\right) \tag{8}$$

$$p(\boldsymbol{z}^s | \boldsymbol{\pi}^s) = \mathrm{Cat}(\boldsymbol{z}^s | \boldsymbol{\pi}^s) \triangleq \prod_{k=1}^{K} (\pi_k^s)^{z_k^s}, \tag{9}$$

where $\rho, \lambda_0, \boldsymbol{m}_0$ are predefined constants, "Cat" denotes the categorical distribution, and $\|\cdot\|_1$ denotes the ℓ_1-norm. The parameter $\boldsymbol{\pi}^s$ must satisfy $\sum_{k=1}^{K} \pi_k^s = 1$ and $0 \leq \pi_k^s \leq 1$. In this model, task-relatedness is represented by the K sets of model parameters $\{(\boldsymbol{\mu}_k, \Lambda_k)\}_{k=1}^{K}$ that are shared by all the participants, while the distribution $\boldsymbol{\pi}^s$ represents a unique property of the participant s. As above, we will use $p(\cdot)$ and $q(\cdot)$ as symbolic notation representing probability density functions in general, rather than a specific functional form.

4.4 CollabDict implementation of GGM mixture

In this generative model, the local parameter set $\boldsymbol{\phi}^s$ in Algorithm 1 is

$$\boldsymbol{\phi}^s = \left\{(\pi_1^s, \ldots, \pi_K^s), (r_1^{s(1)}, \ldots, r_K^{s(1)}), \ldots, (r_1^{s(N^s)}, \ldots, r_K^{s(N^s)})\right\}, \tag{10}$$

and the intermediate parameters are

$$\boldsymbol{\theta}^s = \left\{(N_1^s, \ldots, N_K^s), (\boldsymbol{m}_1^s, \ldots, \boldsymbol{m}_K^s), (\mathsf{C}_1^s, \ldots, \mathsf{C}_K^s)\right\}. \tag{11}$$

Here, the intuition of these parameters is as follows. $r_k^{s(n)}$ is the probability that $\boldsymbol{x}^{s(n)}$ belongs to the k-th pattern. N_k^s is the average number of samples in $\mathcal{D}^s$ that fall into the k-th pattern. π_k^s is the probability that an arbitrary sample in $\mathcal{D}^s$ belongs to the k-th pattern. $\boldsymbol{m}_k^s, \mathsf{C}_k^s$ correspond to the mean vector and the scatter matrix of the k-th pattern computed only with the samples in $\mathcal{D}^s$.

`local_update` computes these parameters in the following way, assuming that $\boldsymbol{\mu}, \Lambda$ and π_k^s have been properly initialized:

$$r_k^{s(n)} = \frac{\pi_k^s \mathcal{N}(\boldsymbol{x}^{s(n)} \mid \boldsymbol{m}_k, (\Lambda_k)^{-1})}{\sum_{l=1}^K \pi_l^s \mathcal{N}(\boldsymbol{x}^{s(n)} \mid \boldsymbol{m}_l, (\Lambda_l)^{-1})}, \tag{12}$$

$$N_k^s = \sum_{n \in \mathcal{D}^s} r_k^{s(n)}, \quad \pi_k^s = \frac{N_k^s}{\sum_{l=1}^K N_l^s}, \quad \boldsymbol{m}_k^s = \sum_{n \in \mathcal{D}^s} r_k^{s(n)} \boldsymbol{x}^{s(n)}, \quad \mathsf{C}_k^s = \sum_{n \in \mathcal{D}^s} r_k^{s(n)} \boldsymbol{x}^{s(n)} \boldsymbol{x}^{s(n)\top}, \tag{13}$$

For each participant s, these are computed for all $k = 1, \ldots, K$ and $n \in \mathcal{D}^s$.

Once $\boldsymbol{\phi}^s, \boldsymbol{\theta}^s$ are computed for each s, the participants start running the `consensus` routine to compute aggregations. Specifically, it computes

$$\bar{N}_k = \sum_{s=1}^S N_k^s, \quad \bar{\boldsymbol{m}}_k = \frac{1}{\bar{N}_k} \sum_{s=1}^S \boldsymbol{m}_k^s, \quad \bar{\mathsf{C}}_k = \frac{1}{\bar{N}_k} \sum_{s=1}^S \mathsf{C}_k^s \tag{14}$$

with the dynamical consensus algorithm. Upon convergence, all the participants end up having these aggregated values in addition to the local parameters $\boldsymbol{\phi}^s$.

Finally, each participant runs `local_update` to obtain $\boldsymbol{\theta} = \{(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K), (\Lambda_1, \ldots, \Lambda_K)\}$ based on $\bar{N}_k, \bar{\boldsymbol{m}}_k$ and $\bar{\mathsf{C}}_k$:

$$\boldsymbol{\mu}_k = \frac{1}{\lambda_0 + \bar{N}_k} (\lambda_0 \boldsymbol{m}_0 + \bar{N}_k \bar{\boldsymbol{m}}_k), \tag{15}$$

$$\Sigma_k = \bar{\mathsf{C}}_k - \bar{\boldsymbol{m}}_k \bar{\boldsymbol{m}}_k^\top + \frac{\lambda_0}{\lambda_0 + \bar{N}_k} (\bar{\boldsymbol{m}}_k - \boldsymbol{m}_0)(\bar{\boldsymbol{m}}_k - \boldsymbol{m}_0)^\top,$$

$$\Lambda_k = \arg\max_{\Lambda_k} \left\{ \frac{\bar{N}_k + 1}{\bar{N}_k} \ln |\Lambda_k| - \mathrm{Tr}(\Lambda_k \Sigma_k) - \frac{\rho}{\bar{N}_k} \|\Lambda_k\|_1 \right\}, \tag{16}$$

where $|\Lambda_k|$ is the matrix determinant. The objective function in Eq. (16) is convex and can be efficiently solved by the graphical lasso algorithm [11]. Due to the ℓ_1 regularization, the resulting precision matrix is sparse, i.e., has many zero entries. Sparsity not only improves numerical stability but also contributes to better interpretability. Since the matrix elements of Λ_k represent the partial correlation coefficients according to the theory of the Gaussian graphical model [18], one can easily find directly correlated variables for each variable.

4.5 Anomaly score of GGM mixture

Now that we have described how to obtain model parameters $\{\boldsymbol{\pi}^s, \boldsymbol{\mu}, \Lambda\}$, let us provide a concrete expression for the anomaly score in Eq. (1). One issue here is that the observation model (7) has a latent variable $\boldsymbol{z}^s$. One standard procedure to define an anomaly score in such a case is to use the posterior average. In the present multi-task context, it is given by

$$a^s(\boldsymbol{x}^{s,\mathrm{test}}) = \int \mathrm{d}\boldsymbol{z}^s \, q(\boldsymbol{z}^s) \left\{ -\ln p(\boldsymbol{x}^s \mid \boldsymbol{z}^s, \boldsymbol{\mu}, \Lambda) \right\}. \tag{17}$$

Here, $q(\boldsymbol{z}^s)$ is the posterior distribution, given the test sample. Similar to the in-sample posterior in Eq. (12), $q(\boldsymbol{z}^s)$ is a categorical distribution as

$$q(\boldsymbol{z}^s) = \prod_{k=1}^K \left\{ r_k^s(\boldsymbol{x}^{s,\mathrm{test}}) \right\}^{z_k^s}, \qquad r_k^s(\boldsymbol{x}^{s,\mathrm{test}}) = \frac{\pi_k^s \mathcal{N}(\boldsymbol{x}^{s,\mathrm{test}} \mid \boldsymbol{m}_k, (\Lambda_k)^{-1})}{\sum_{l=1}^K \pi_l^s \mathcal{N}(\boldsymbol{x}^{s,\mathrm{test}} \mid \boldsymbol{m}_l, (\Lambda_l)^{-1})} \tag{18}$$

and hence, the integration in Eq. (17) should be understood as the summation over the K different choices of $\boldsymbol{z}^s$. Performing the summation, we have

$$a^s(\boldsymbol{x}^{s,\text{test}}) = -\sum_{k=1}^{K} r_k^s(\boldsymbol{x}^{s,\text{test}}) \ln \mathcal{N}(\boldsymbol{x}^{s,\text{test}} \mid \boldsymbol{\mu}_k, (\Lambda_k)^{-1}). \tag{19}$$

This expression clearly represents the intuition of dictionary learning. For a test sample $\boldsymbol{x}^{s,\text{test}}$, we first evaluate the degree of fit $r_k^s(\boldsymbol{x}^{s,\text{test}})$ for all k's. Some of the K patterns may have dominating weights while others may be negligible. The final anomaly score is a weighted sum over the one-pattern anomaly scores $\{-\ln \mathcal{N}(\boldsymbol{x}^{s,\text{test}} \mid \boldsymbol{\mu}_k, (\Lambda_k)^{-1})\}$. It is straightforward to see that this probabilistic definition of the anomaly score is a natural generalization of the ones based on the reconstruction errors. By using the explicit expression of the Gaussian distribution, we have

$$a^s(\boldsymbol{x}^{s,\text{test}}) = \sum_{k=1}^{K} r_k^s(\boldsymbol{x}^{s,\text{test}}) \left\{ \frac{M}{2} \ln(2\pi) - \frac{1}{2} \ln |\Lambda_k| + \frac{1}{2} (\boldsymbol{x}^{s,\text{test}} - \boldsymbol{\mu}_k)^\top \Lambda_k (\boldsymbol{x}^{s,\text{test}} - \boldsymbol{\mu}_k) \right\}. \tag{20}$$

The third term in the parenthesis is often called the Mahalanobis distance, which has been used as the main anomaly metric in the classical Hotelling's T^2 theory [2]. If $\boldsymbol{x}^{s,\text{test}}$ has a dominant weight at a certain k, the $\boldsymbol{\mu}_k$ can be viewed as a denoised version of $\boldsymbol{x}^{s,\text{test}}$. The similarity to the reconstruction error by the squared loss is obvious.

5. Deep Decentralized Multi-Task Density Estimation

While the sparse GGM mixture-based CollabDict framework serves as a robust and interpretable tool for anomaly detection, its reliance on Gaussian assumptions may limit its effectiveness in systems requiring considerations of higher-order correlations. This section discusses an extension of CollabDict to the variational autoencoder (VAE), a standard deep-learning-based density estimation algorithm.

5.1 Probabilistic model definition

Observing the GGM-mixture model discussed in the previous section, one interesting question is what happens if we replace the observation model in Eq. (7) with the neural Gaussian distribution:

$$p(\boldsymbol{x}^s \mid \boldsymbol{z}^s, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}^s \mid \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}^s), \Sigma_{\boldsymbol{\theta}}(\boldsymbol{z}^s)). \tag{21}$$

Here, $\boldsymbol{z}^s$ is a latent variable associated with $\boldsymbol{x}^s$. We assume that $\boldsymbol{z}^s$ is in $\mathbb{R}^d$ and the mean and the covariance matrix in Eq. (21) are represented by multi-layer perceptrons (MLPs):

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}^s) = \text{MLP}_{\boldsymbol{\theta}}(\boldsymbol{z}^s), \quad \Sigma_{\boldsymbol{\theta}}(\boldsymbol{z}^s) = \text{MLP}_{\boldsymbol{\theta}}(\boldsymbol{z}^s). \tag{22}$$

Typically, $\Sigma_{\boldsymbol{\theta}}$ is assumed to be a diagonal matrix such as $\Sigma_{\boldsymbol{\theta}} = \text{diag}(\boldsymbol{\sigma}_{\boldsymbol{\theta}})^2$, where $\boldsymbol{\sigma}_{\boldsymbol{\theta}} \triangleq (\sigma_{\boldsymbol{\theta},1}, \ldots, \sigma_{\boldsymbol{\theta},M})^\top$. The functional form of the MLPs has a lot of flexibility. One typical choice is (see, e.g., [39]):

$$\boldsymbol{g} = \text{ReLU}(\text{Linear}(\boldsymbol{z}^s)), \quad \boldsymbol{\mu}_{\boldsymbol{\theta}} = \text{Linear}(\boldsymbol{g}), \quad \boldsymbol{\sigma}_{\boldsymbol{\theta}} = \ln(1 + \exp(\text{Linear}(\boldsymbol{g}))), \tag{23}$$

where $\text{Linear}(\cdot)$ performs the Affine transform such that $\text{Linear}(\boldsymbol{z}^s) = \mathsf{W}\boldsymbol{z}^s + \boldsymbol{b}$ (W is a parameter matrix and $\boldsymbol{b}$ is a parameter vector to be learned) and ReLU denotes the rectified linear unit (or the hinge loss function). All the functions operate element-wise. The r.h.s. of the last equation is called the softmax function. Model parameters $\boldsymbol{\theta}$ to be learned in this case are transformation matrices and intercept vectors in the three Affine transforms.

Since the latent variable $\boldsymbol{z}^s$ is assumed to be in $\mathbb{R}^d$ this time, we assign a Gaussian prior to $\boldsymbol{z}^s$:

$$p(\boldsymbol{z}^s) = \mathcal{N}(\boldsymbol{z}^s \mid \boldsymbol{0}, \mathsf{I}_d), \tag{24}$$

where I_d is the d-dimensional identity matrix.

5.2 Multi-task variational autoencoder

Since $\boldsymbol{z}^s$ is unobservable, the marginalized likelihood is used to learn the model parameters $\boldsymbol{\theta}$. The log marginalized likelihood is defined by

$$L_0(\boldsymbol{\theta}) = \ln \prod_{s=1}^{S} \prod_{n \in \mathcal{D}^s} \int \mathrm{d}\boldsymbol{z}^{s(n)} \; p\left(\boldsymbol{x}^{s(n)} \mid \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}^{s(n)}), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z}^{s(n)})\right) p(\boldsymbol{z}^{s(n)}). \tag{25}$$

While the integration is analytically intractable, generating samples from the Gaussian prior is straightforward and hence, one might want to use Monte Carlo estimation for estimating the integral. However, as discussed by Kingma and Welling [17], direct Monte Carlo estimation of the pdf is numerically challenging, since the probability mass is often concentrated in very limited areas and takes almost zero value everywhere else.

One standard approach to address these issues is to leverage the variational lower bound derived with Jensen's inequality:

$$L_0 \geq L \triangleq \sum_{s=1}^{S} \sum_{n \in \mathcal{D}^s} \int \mathrm{d}\boldsymbol{z}^{s(n)} \; q(\boldsymbol{z}^{s(n)}) \ln \frac{p\left(\boldsymbol{x}^{s(n)} \mid \boldsymbol{z}^{s(n)}, \boldsymbol{\theta}\right) p(\boldsymbol{z}^{s(n)})}{q(\boldsymbol{z}^{s(n)})}. \tag{26}$$

The inequality holds for any distribution $q(\boldsymbol{z}^{s(n)})$. Notice that the integrand is now $\ln p$, which is more tractable than the pdf itself. Similar to the EM iteration for the mixture model, one can use a two-stage learning strategy: 1) Optimize $q(\cdot)$, given $\boldsymbol{\theta}$, and 2) optimize $\boldsymbol{\theta}$, given $q(\cdot)$. Unlike the GGM mixture case, however, the first stage does not lead to an analytic solution due to the nonlinear dependency on $\boldsymbol{z}^s$ in the MLPs. Thus, we introduce another parametric model to approximate the $q(\boldsymbol{z}^s)$ that would provide the tightest bound:

$$q(\boldsymbol{z}^s) \approx q(\boldsymbol{z}^s \mid \boldsymbol{x}^s, \boldsymbol{\phi}^s) = \mathcal{N}\left(\boldsymbol{z}^s \mid \boldsymbol{m}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^s), \mathsf{H}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^s)\right), \tag{27}$$

where the mean and the covariance matrix are assumed to be MLPs parameterized with $\boldsymbol{\phi}^s$ in a similar fashion to Eqs. (22)-(23). The first stage is now an optimization problem over $\boldsymbol{\phi}^1, \ldots, \boldsymbol{\phi}^S$, given $\boldsymbol{\theta}$. Specifically, for $s = 1, \ldots, S$, we solve $\boldsymbol{\phi}^s = \arg\max_{\boldsymbol{\phi}^s} L^s(\boldsymbol{\phi}^s, \boldsymbol{\theta})$, given the latest $\boldsymbol{\theta}$, where

$$L^s(\boldsymbol{\phi}^s, \boldsymbol{\theta}) \triangleq \sum_{n \in \mathcal{D}^s} \int \mathrm{d}\boldsymbol{z}^{s(n)} \; q(\boldsymbol{z}^{s(n)} \mid \boldsymbol{x}^{s(n)}, \boldsymbol{\phi}^s) \ln \frac{p\left(\boldsymbol{x}^{s(n)} \mid \boldsymbol{z}^{s(n)}, \boldsymbol{\theta}\right) p(\boldsymbol{z}^{s(n)})}{q(\boldsymbol{z}^{s(n)} \mid \boldsymbol{x}^{s(n)}, \boldsymbol{\phi}^s)}. \tag{28}$$

In the second stage, with an updated $\{\boldsymbol{\phi}^1, \ldots, \boldsymbol{\phi}^S\}$, we solve

$$\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}} \sum_{s=1}^{S} L^s(\boldsymbol{\phi}^s, \boldsymbol{\theta}). \tag{29}$$

This learning strategy closely resembles the variational autoencoder (VAE) [17] with one major exception that the "encoder" distribution $q(\boldsymbol{z}^s \mid \boldsymbol{x}^s, \boldsymbol{\phi}^s)$ now depends on s, allowing us to capture situations that are specific to the s-th system, and hence, to perform multi-task learning. Figure 2 illustrates the overall architecture of the model. Following the VAE literature, the observation model $p(\boldsymbol{x}^s \mid \boldsymbol{z}^s, \boldsymbol{\theta})$ can be called the decoder distribution.

5.3 Training procedure on CollabDict

Now let us examine how the above two-stage procedure is applied to the CollabDict framework. First, we observe that the objective function $L^s(\boldsymbol{\phi}^s, \boldsymbol{\theta})$ is expressed as

$$L^s(\boldsymbol{\phi}^s, \boldsymbol{\theta}) = \frac{1}{2} \sum_{n \in \mathcal{D}^s} \left\{ \ln |\mathsf{H}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s(n)})| + d - \mathrm{Tr}\left(\mathsf{H}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s(n)})\right) + \|\boldsymbol{m}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s(n)})\|_2^2 \right\}$$

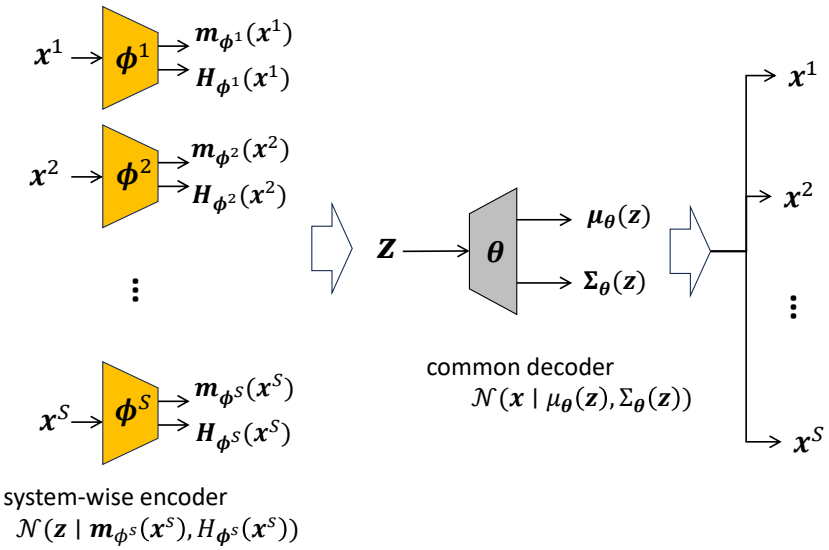

Fig. 2. Overall architecture of the proposed multi-task VAE, which operates under decentralized and privacy-preserving constraints.

$$+\frac{1}{2}\sum_{n\in\mathcal{D}^s}\left\langle -M\ln(2\pi)-\ln|\Sigma_\theta^{s(n)}|-(x^{s(n)}-\mu_\theta^{s(n)})^\top(\Sigma_\theta^{s(n)})^{-1}(x^{s(n)}-\mu_\theta^{s(n)})\right\rangle_v, \tag{30}$$

where $\mathrm{Tr}(\cdot)$ represents the matrix trace and $\|\cdot\|_2$ represents the vector ℓ_2-norm. H_{ϕ^s} is assumed to be diagonal, such that $\mathsf{H}_{\phi^s} = \mathrm{diag}(h_{\phi^s})^2$ with $h_{\phi^s} = (h_{\phi^s,1},\ldots,h_{\phi^s,M})^\top$. In the second line, $\langle\cdot\rangle_v$ represents the expectation with respect to $v \sim \mathcal{N}(v \mid 0, \mathsf{I}_d)$, and

$$\mu_\theta^{s(n)} \triangleq \mu_\theta\left(h_{\phi^s}(x^{s(n)})\odot v + m_{\phi^s}(x^{s(n)})\right), \quad \Sigma_\theta^{s(n)} \triangleq \Sigma_\theta\left(h_{\phi^s}(x^{s(n)})\odot v + m_{\phi^s}(x^{s(n)})\right), \tag{31}$$

where $\odot$ denotes the element-wise product of vectors. The integration is analytically intractable and has to be done numerically. Fortunately, Monte Carlo sampling from $\mathcal{N}(v \mid 0, \mathsf{I}_d)$ is straightforward. For an arbitrary function $F(\cdot)$, it follows:

$$\langle F(v)\rangle_v \approx \frac{1}{J}\sum_{j=1}^{J} F(v^{[j]}), \quad \text{where} \quad v^{[1]},\ldots,v^{[J]} \sim \mathcal{N}(\cdot \mid 0, \mathsf{I}_d). \tag{32}$$

This is used to evaluate the second line of $L^s(\phi^s,\theta)$ in Eq. (30).

In the proposed multi-task VAE model, stochastic gradient descent (SGD) is employed for parameter estimation. In the `local_update` module of Algorithm 1, for each s, the local parameter ϕ^s is updated as

$$\phi^s \leftarrow \phi^s + \eta\frac{\partial L^s(\phi^s,\theta)}{\partial\phi^s}, \tag{33}$$

where θ is fixed to its latest numerical value and η is the learning rate, which is a hyper-parameter. The mini-batch approach [43] can be used here, but we have omitted it for simplicity. With the updated ϕ^s, the local gradient with respect to θ is computed as

$$\theta^s = \frac{\partial L^s(\phi^s,\theta)}{\partial\theta}. \tag{34}$$

Note that these calculations can be performed locally using only $\mathcal{D}^s$.

In the `consensus` module, the dynamical consensus algorithm performs simple aggregation:

$$\partial\boldsymbol{\theta} = \texttt{consensus}(\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^S) = \sum_{s=1}^{S} \boldsymbol{\theta}^s. \tag{35}$$

Finally, in the `optimization` module, with the latest $\boldsymbol{\phi}^s$ and $\partial\boldsymbol{\theta}$, the global parameter $\boldsymbol{\theta}$ is updated:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta\, \partial\boldsymbol{\theta}. \tag{36}$$

Note that this operation is done by each participant locally in the absence of a central coordinator.

5.4 Anomaly score

By running CollabDict for the multi-task VAE model, we can obtain the model parameters $\{\boldsymbol{\phi}^s\}, \boldsymbol{\theta}$. We now turn to the general definition of the anomaly score in Eq. (17). Using the encoder and decoder distributions, we have

$$a^s(\boldsymbol{x}^{s,\text{test}}) = -\int \mathrm{d}\boldsymbol{z}^s\; q(\boldsymbol{z}^s \mid \boldsymbol{x}^{s,\text{test}}, \boldsymbol{\phi}^s) \ln p(\boldsymbol{x}^{s,\text{test}} \mid \boldsymbol{z}^s, \boldsymbol{\theta}), \tag{37}$$

$$= -\int \mathrm{d}\boldsymbol{z}^s\; \mathcal{N}\Big(\boldsymbol{z}^s \mid \boldsymbol{m}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s,\text{test}}), \mathsf{H}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s,\text{test}})\Big) \ln \mathcal{N}(\boldsymbol{x}^{s,\text{test}} \mid \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}^s), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z}^s)). \tag{38}$$

As discussed in Sec. 3, the logarithmic loss or its expectation for an anomaly score has been used in the literature for a few decades often with a solid information-theoretical background [19, 29, 34, 40, 41]. In the particular context of VAE-based anomaly detection, the anomaly score in Eq. (37) was first introduced by An and Cho [1], who called it the "reconstruction probability" despite the fact that it is not a pdf. Since then, Eq. (37) has been accepted as one of the standard definitions [39].

Because of the nonlinearity of MLPs, the integration is analytically intractable. Fortunately, sampling from a multivariate Gaussian with a diagonal covariance matrix is straightforward. Monte Carlo sampling leads to the following expression:

$$a^s(\boldsymbol{x}^{s,\text{test}}) = -\frac{1}{J}\sum_{j=1}^{J} \ln \mathcal{N}(\boldsymbol{x}^{s,\text{test}} \mid \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z}^{[j]}), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\boldsymbol{z}^{[j]})), \tag{39}$$

$$\boldsymbol{z}^{[1]}, \ldots, \boldsymbol{z}^{[J]} \sim \mathcal{N}\Big(\cdot \mid \boldsymbol{m}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s,\text{test}}), \mathsf{H}_{\boldsymbol{\phi}^s}(\boldsymbol{x}^{s,\text{test}})\Big). \tag{40}$$

It is interesting to compare this with the one derived from the GGM mixture model in Eq. (19). In both cases, the anomaly score is a linear combination of the log-loss score of individual patterns, demonstrating the versatility of the CollabDict framework across a wide range of model classes.

Table I summarizes the comparison between two model classes for CollabDict. Although deep learning models are generally considered more expressive than non-deep models, such advantages come with a few potential issues.

In the expression of the anomaly score, the GGM mixture model deterministically stores the pattern dictionary (Eq. (19)) while the multi-task VAE generates a pattern dictionary "on-demand" through Monte Carlo samples from the posterior distribution (Eq. (39)). In the latter, one critical issue is *posterior collapse* [9, 22, 37], which refers to the situation where the posterior $q(\boldsymbol{z}^s \mid \boldsymbol{x}^s, \boldsymbol{\phi}^s)$ collapses towards the prior $p(\boldsymbol{z})$, losing the information of the input sample $\boldsymbol{x}^{s,\text{test}}$. Posterior collapse occurs when the model is overly expressive. When it occurs, the anomaly score can be arbitrarily large even for normal samples, which is a devastating situation in anomaly detection. Due to the randomness of SGD and Monte Carlo sampling, the occurrence of posterior collapse is often unpredictable. Although the issue is reminiscent of rank deficiency in Gaussian mixtures, a few established

Table I. Comparison between decentralized GGM mixture and multitask VAE.

	decentralized GGM mixture	decentralized multi-task VAE
pattern dictionary	deterministic	on-demand
interpretability	$r_k^s(\boldsymbol{x}^s)$: sample-wise pattern weight $\boldsymbol{\pi}^s$: system-wise pattern weight $\boldsymbol{\mu}_k$: pattern center Λ_k: variable interdependency	$\boldsymbol{m}_{\phi^s}(\boldsymbol{x})$: system-specific embedding $\mathsf{H}_{\phi^s}(\boldsymbol{x})$: embedding confidence $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{z})$: sample-wise reconstruction $\Sigma_{\boldsymbol{\theta}}(\boldsymbol{z})$: reconstruction confidence
data privacy	theoretical analysis exist	(largely open)
model complexity control	well understood	(largely open)
model instability sources	parameter initialization	parameter initialization latent dimension d SGD hyperparameters (η, etc.) posterior collapse Monte Carlo sampling
Computational cost	low	high

techniques are available for Gaussian mixtures to properly regularize the model complexity such as ℓ_1-regularization on Λ_k and ℓ_0-regularization on $\boldsymbol{\pi}^s$ [14, 31]. The latter is used as a tool to determine K automatically.

Another aspect in favor of the GGM mixture is the variety of factors towards model instabilities. In multi-task VAE, tuning hyper-parameters, such as the latent dimension d, learning rate η, batch size, etc., requires a lot of trial and error. This can be problematic in practice as the number of samples, especially anomalous samples, is often limited. Brute-force data-driven parameter tuning strategies, developed in domains that have access to internet-scale datasets (such as speech, text, and images), are often not very useful in real-world applications. This is mainly because the problem setting in real-world scenarios is typically domain-specific, and the benchmark tasks are not directly relevant.

Finally, due to the black-box nature, evaluating the risk of privacy breach is generally challenging in deep learning models. In the next section, we provide some mathematical discussions on data privacy preservation under the GGM mixture model.

6. Privacy Preservation in CollabDict

This section discusses privacy preservation in CollabDict in the GGM mixture setting. Our main focus in this section is to provide a privacy guarantee when the trained model is used by a third party as a "pre-trained model." Additionally, we provide a practical means to monitor internal privacy breaches.

6.1 Differentially privacy in CollabDict

The learned pattern dictionary is shared not only among the participant nodes of the network but also externally as a pre-learned anomaly detection model, depending on business use-cases. In such a scenario, we are concerned about the risk that any of the raw samples are reverse-engineered from the published model.

The notion of differential privacy [10] is useful to quantitatively evaluate the privacy risk. Assume that we have created a new dataset $\tilde{\mathcal{D}}$ by perturbing a single sample, say the n'-th sample of the s-th participant: $\boldsymbol{x}^{s(n')} \to \tilde{\boldsymbol{x}}^{s(n')}$. As illustrated in Fig. 3, when publishing the global state variable, *e.g.*, $\boldsymbol{\mu}_k$,

we say that $\boldsymbol{\mu}_k$ has differential privacy if the state variable computed on $\tilde{\mathcal{D}}$ is not distinguishable from that computed on $\mathcal{D}$ under a data release mechanism. Here the *mechanism* means an appropriately defined data sanitization method, which is to add random noise in this case.

Formally, we define differential privacy as follows:

Definition 1 (Rényi differential privacy [26])**.** *A mechanism $f : \mathcal{D} \to \boldsymbol{\eta}$ is said to have $(1, \epsilon)$-Rényi differential privacy if there exists a constant ϵ for any adjacent datasets $\mathcal{D}, \tilde{\mathcal{D}}$ satisfying*

$$\mathrm{KL}[f \mid \mathcal{D}, \tilde{\mathcal{D}}] \triangleq \int \mathrm{d}\boldsymbol{\eta}\; f(\boldsymbol{\eta} \mid \mathcal{D}) \ln \frac{f(\boldsymbol{\eta} \mid \mathcal{D})}{f(\boldsymbol{\eta} \mid \tilde{\mathcal{D}})} \leq \epsilon. \tag{41}$$

In words, the mechanism is differentially private if the adjacent datasets are not distinguishable in terms of the Kullback-Leibler (KL) divergence.

This definition is a special case of the more general (α, ϵ)-Rényi differential privacy [26], which also includes the original definition of ϵ-differential privacy [10] as $\alpha \to \infty$. A major motivation for this extension is that the ϵ-differential privacy cannot properly handle the Gaussian mechanism and thus is not generally appropriate for real-valued noisy data.

In our Bayesian learning framework, the pattern center parameters $\{\boldsymbol{\mu}_k\}$ are potentially the most vulnerable quantity because their posterior mean is represented as a linear combination of the raw samples. One reasonable choice for the data release mechanism is the posterior distribution. While we MAP-estimated $\boldsymbol{\mu}_k$ in Sec. 4.4, it is also possible to find the posterior distribution, denoted by $q(\boldsymbol{\mu}_k \mid \Lambda_k)$. As derived in Appendix A, we have

$$q(\boldsymbol{\mu}_k \mid \Lambda_k) = \mathcal{N}(\boldsymbol{\mu}_k \mid \boldsymbol{w}_k, (\lambda_k \Lambda_k)^{-1}), \quad \boldsymbol{w}_k \triangleq \frac{1}{\lambda_0 + \bar{N}_k}(\lambda_0 \boldsymbol{m}_0 + \bar{N}_k \bar{\boldsymbol{m}}_k), \quad \lambda_k \triangleq \lambda_0 + \bar{N}_k. \tag{42}$$

Notice that the posterior mean $\boldsymbol{w}_k$ is the same as the MAP estimate in Eq. (15). Let $\tilde{\boldsymbol{w}}_k$ be the corresponding posterior mean in the perturbed dataset $\tilde{\mathcal{D}}$. Assuming the other parameters $\{\lambda_k, \Lambda_k\}$ are fixed, we have

$$\begin{aligned}
&\int \mathrm{d}\boldsymbol{\eta}\; \mathcal{N}(\boldsymbol{\eta} \mid \boldsymbol{w}_k, (\lambda_k \Lambda_k)^{-1}) \ln \frac{\mathcal{N}(\boldsymbol{\eta} \mid \boldsymbol{w}_k, (\lambda_k \Lambda_k)^{-1})}{\mathcal{N}(\boldsymbol{\eta} \mid \tilde{\boldsymbol{w}}_k, (\lambda_k \Lambda_k)^{-1})} \\
&\quad = \frac{1}{2} \lambda_k (\boldsymbol{w}_k - \tilde{\boldsymbol{w}}_k)^\top \Lambda_k (\boldsymbol{w}_k - \tilde{\boldsymbol{w}}_k), \\
&\quad = \frac{1}{2} \lambda_k \left(\frac{r_k^{s(\tilde{n})}}{\lambda_k} \right)^2 (\boldsymbol{x}^{s(\tilde{n})} - \tilde{\boldsymbol{x}}^{s(\tilde{n})})^\top \Lambda_k (\boldsymbol{x}^{s(\tilde{n})} - \tilde{\boldsymbol{x}}^{s(\tilde{n})}) \qquad (43) \\
&\quad \leq \frac{1}{2\lambda_k} \max_j \{r_k^{s(j)}\}^2 \times \|\Lambda_k\|_2 \times \|\boldsymbol{x}^{s(\tilde{n})} - \tilde{\boldsymbol{x}}^{s(\tilde{n})}\|_2^2 \qquad (44)
\end{aligned}$$

where $\|\cdot\|_2$ denotes the ℓ_2 norm. The matrix ℓ_2 norm is the same as the maximum singular value of the matrix.

Here let us assume that the pairwise distance between samples in the dataset is upper-bounded by $R < \infty$. This can be easily done by removing obvious outliers in the preprocess. As discussed previously, our Bayesian learning algorithm allows for automatically removing irrelevant mixture components during the iteration. Hence, we can further assume that we only retain sufficiently large $\bar{N}_k$ after the `local_update` procedure in Algorithm 1, *i.e.*, $\bar{N}_k > \delta > 0$.

Theorem 1. *We assume that any pairwise ℓ_2 distance between the samples is upper-bounded by $R < \infty$, and the components whose $\bar{N}_k < \delta$ are discarded after the* `local_update` *procedure in Algorithm 1, and $\lambda_0 > 0$. Then the release mechanism using the posterior distribution has $(1, \epsilon)$-Rényi differential privacy for some constant ϵ.*

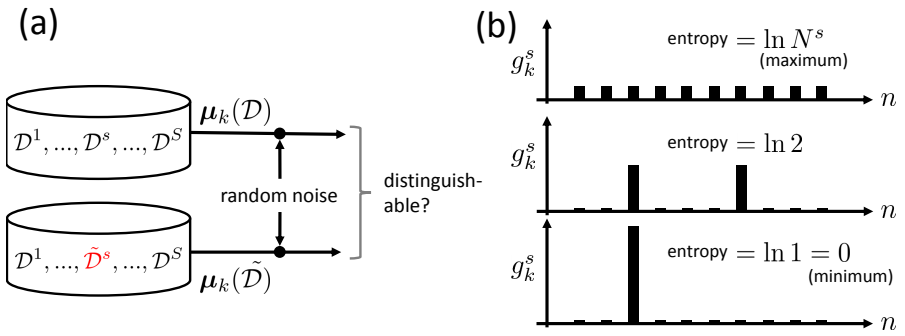

Fig. 3. Illustration of (a) differential privacy and (b) the entropy ℓ-diversity. In the latter, the uniform distribution gives the maximum entropy.

Proof. We need to bound the right hand-side of Eq. (44) by a constant. From the definition of λ_k, it is plain to see that $1/\lambda_k < 1/\lambda_0$. Note that $r_k^{s(j)} < 1$ for all s and j. It implies that

$$\frac{1}{2\lambda_k} \max_j \{r_k^{s(j)}\}^2 \|\boldsymbol{x}^{s(\tilde{n})} - \tilde{\boldsymbol{x}}^{s(\tilde{n})}\|_2^2 \leq \frac{R^2}{2\lambda_0}. \tag{45}$$

As shown in Appendix B, there exists an upper-bound in $\|\Lambda_k\|_2$. Letting B the upper-bound, we have

$$\mathrm{KL}[f \mid \mathcal{D}, \tilde{\mathcal{D}}] \leq \frac{KBR^2}{2\lambda_0} \triangleq \epsilon, \tag{46}$$

which completes the proof. □

This theorem guarantees the differential privacy of the CollabDict protocol when publishing the global state variables to external parties.

6.2 Entropy ℓ-diversity in CollabDict

Another aspect of privacy that we are concerned with is privacy preservation in consensus building. In this case, the most vulnerable quantity is potentially $\boldsymbol{m}_k^s$ because it is a linear combination of the raw samples (see Eq. (13)). The question is whether the data release mechanism for $\boldsymbol{m}_k^s$ poses any risk of reverse-engineering any of the samples in $\mathcal{D}^s$ through $\boldsymbol{m}_k^s$.

To address this problem, let us define a probability distribution over $n \in \{1, \ldots, N^s\}$:

$$g_k^s(n) = \frac{\mathcal{N}(\boldsymbol{x}^{s(n)} | \boldsymbol{m}_k, (\Lambda_k)^{-1})}{\sum_{m=1}^{N^s} \mathcal{N}(\boldsymbol{x}^{s(m)} | \boldsymbol{m}_k, (\Lambda_k)^{-1})}, \tag{47}$$

which is a normalized version of the sample weight $r_k^{s(n)}$ in Eq. (12). If, for example, only a single n dominates the distribution, $\boldsymbol{m}_k^s$ will be a faithful surrogate of the raw sample $\boldsymbol{x}^{s(n)}$. Therefore, this sample is exposed to a peer consensus node by sharing $\boldsymbol{m}_k^s$.

The notion of *entropy ℓ-diversity* is useful for qualitatively evaluating such a risk:

Definition 2 (entropy ℓ-diversity [23])**.** *A release mechanism with a probability distribution over database entries is said to have the entropy ℓ-diversity is the entropy of the distribution is lower-bounded with* $\ln \ell$.

As illustrated in Fig. 3 (b), entropy is a useful measure of the non-identifiability of the individual samples. The value ℓ provides an intuitive measure of how many elements are outstanding in the distribution.

Therefore, if privacy preservation among the consensus nodes is a significant concern, the participants should monitor the value

$$E^s \triangleq \max_k \left\{ -\sum_{n=1}^{N^s} g_k^s(n) \ln g_k^s(n) \right\} \tag{48}$$

to determine if it is greater than a threshold $\ln l_0$. If $E^s < \ln l_0$ holds during consensus building, the participant should consider adding random noise to the samples. To avoid introducing unwanted bias to the model, the participant can use the posterior $\mathcal{N}(\cdot \mid \boldsymbol{w}_k, (\lambda_k \Lambda_k)^{-1})$ for the noise, ensuring privacy preservation as guaranteed by Theorem 1.

7. Conclusion

In this paper, we have presented two methodological advancements in decentralized multi-task learning under privacy constraints, aiming to pave the way for future developments in next-generation Blockchain platforms. First, we extended the collaborative dictionary learning (CollabDict) framework, which has been limited to GGM mixture models so far, to incorporate variational autoencoders (VAEs) for enhanced anomaly detection. While VAEs have been widely applied to anomaly detection problems, to the best of our knowledge, this is the first framework satisfying both decentralization and data privacy constraints.

Furthermore, we proposed a theoretical framework to analyze the risk of data privacy leakage when models trained with the CollabDict framework are shared. Given the widespread application of "pre-trained models" in large-scale deep learning, analyzing external data privacy breach is of significant practical importance. We showed that the CollabDict approach, when applied to GGM mixtures, meets $(1, \epsilon)$-Rényi differential privacy criterion. We also proposed a practical metric for monitoring internal privacy breaches during the learning process based on the notion of the entropy ℓ-diversity.

One promising avenue for future research involves conducting instability analyses of VAE-based anomaly detection methods. In particular, addressing the issue of posterior collapse in VAEs, despite recent advancements, remains a significant challenge. Further research is needed both theoretically and empirically.

Appendix A: Posterior distribution of mean parameter

This section provides the derivation of the posterior distribution of $\boldsymbol{\mu}_k$ discussed in Sec. 6.1.

In the GGM mixture model, we have assigned prior distributions to $\{(\boldsymbol{\mu}_k, \Lambda_k)\}$ and $\boldsymbol{z}^{s(n)}$, and therefore, we can find the posterior distributions for these model parameters using Bayes' rule. We assume the following form for the posterior distributions:

$$Q(\boldsymbol{\mu}, \Lambda, \mathsf{Z}) = \prod_{k=1}^{K} q(\boldsymbol{\mu}_k \mid \Lambda_k) q(\Lambda_k) \prod_{s=1}^{S} \prod_{n \in \mathcal{D}^s} q(\boldsymbol{z}^{s(n)}), \tag{A·1}$$

where Z is a collective notation for $\{\boldsymbol{z}^{s(n)}\}$. For $q(\boldsymbol{z}^{s(n)})$, we have provided the posterior as

$$q(\boldsymbol{z}^{s(n)}) = \prod_{k=1}^{K} \left(r_k^{s(n)} \right)^{z_k^{s(n)}} \tag{A·2}$$

in Sec. 4.4. Let us find the posterior distribution for $\boldsymbol{\mu}_k$ under the assumption that Λ is point-estimated. Following the variational Bayes framework [3], we seek $q(\boldsymbol{\mu}_k \mid \Lambda_k)$ in the form:

$$q(\boldsymbol{\mu}_k \mid \Lambda_k) \propto \left\langle p(\boldsymbol{\mu}, \Lambda) \prod_{s=1}^{S} \prod_{n \in \mathcal{D}^s} p(\boldsymbol{x}^{s(n)} \mid \boldsymbol{z}^{s(n)}, \boldsymbol{\mu}, \Lambda) p(\boldsymbol{z}^{s(n)} \mid \boldsymbol{\pi}^s) \right\rangle_{\mathsf{Z}}, \tag{A·3}$$

where $\langle\cdot\rangle_{\mathsf{Z}}$ denotes the expectation with respect to $\prod_{s,n} q(\boldsymbol{z}^{s(n)})$ in Eq. (A·2). By taking the logarithm on both sides and selecting the terms that depend on $\boldsymbol{\mu}_k$, we have

$$\ln q(\boldsymbol{\mu}_k \mid \Lambda_k) = \text{c.} - \frac{1}{2}\operatorname{Tr}(\Lambda_k \mathsf{Q}_k), \tag{A·4}$$

where c. is a constant and

$$\mathsf{Q}_k \triangleq \bar{N}_k \bar{\mathsf{C}}_k - \lambda_k \boldsymbol{w}_k \boldsymbol{w}_k^\top - \lambda_0 \boldsymbol{m}_0 \boldsymbol{m}_0^\top + \lambda_k (\boldsymbol{\mu}_k - \boldsymbol{w}_k)(\boldsymbol{\mu}_k - \boldsymbol{w}_k)^\top, \tag{A·5}$$

$$\lambda_k \triangleq \lambda_0 + \bar{N}_k, \quad \boldsymbol{w}_k \triangleq \frac{1}{\lambda_0 + \bar{N}_k}(\lambda_0 \boldsymbol{m}_0 + \bar{N}_k \bar{\boldsymbol{m}}_k). \tag{A·6}$$

Here, $\bar{N}_k, \bar{\mathsf{C}}_K, \bar{\boldsymbol{m}}_k$ have been defined in Eq. (14). Since $\ln q(\boldsymbol{\mu}_k \mid \Lambda_k)$ in Eq. (A·4) is a quadratic form in $\boldsymbol{\mu}_k$, we have

$$q(\boldsymbol{\mu}_k \mid \Lambda_k) = \mathcal{N}\left(\boldsymbol{\mu}_k \mid \boldsymbol{w}_k, (\lambda_k \Lambda_k)^{-1}\right). \tag{A·7}$$

This is the posterior distribution in Eq. (42) in Sec. 6.1.

Appendix B: Upper-boundedness of precision matrix norm

This section proves the following theorem, which was used in the proof of Theorem 1:

Theorem 2. *The ℓ_2 norm of Λ_k as the solution of Eq.* (16) *is upper-bounded.*

Proof. Let us denote the objective function in Eq. (16) as $-f(\Lambda_k)$:

$$f(\Lambda_k) \triangleq -b_k \ln|\Lambda_k| + \operatorname{Tr}(\Lambda_k \Sigma_k) + \frac{\rho}{\bar{N}_k}\|\Lambda_k\|_1, \tag{B·1}$$

where we have defined $b_k \triangleq \frac{\bar{N}_k + 1}{\bar{N}_k}$. Here, we note that $b_k \ln|\Lambda_k| \leq M b_k \ln\|\Lambda_k\|_2$ holds since $\|\Lambda_k\|_2$ is the largest eigenvalue of Λ_k, which is positive semi-definite. In Theorem 1, we assumed $^\exists\delta,\ \bar{N}_k > \delta > 0$. Hence,

$$M b_k = M\left(1 + \frac{1}{\bar{N}_k}\right) < M\left(1 + \frac{1}{\delta}\right) \triangleq b, \tag{B·2}$$

$$\frac{\rho}{\bar{N}_k} \geq \frac{\rho}{S \max_s N_k^s} \geq \frac{\rho}{S \max_s N^s} \triangleq c. \tag{B·3}$$

Since both Λ_k and Σ_k are positive semi-definite, $\operatorname{Tr}(\Lambda_k \Sigma_k) \geq 0$ holds. Then, we have

$$f(\mathsf{I}_M) \geq f(\Lambda_k) \geq -b_k \ln|\Lambda_k| + \frac{\rho}{N_k}\|\Lambda_k\|_1 \geq -b \ln\|\Lambda_k\|_2 + c\|\Lambda_k\|_1,$$

where I_M is the M-dimensional identity matrix. Hence,

$$\|\Lambda_k\|_1 \leq \frac{f(\mathsf{I}_M)}{c} + \frac{b}{c}\ln\|\Lambda_k\|_2. \tag{B·4}$$

Similarly, we have

$$\operatorname{Tr}(\Lambda_k \Sigma_k) \leq f(\Lambda_k) + b_k \ln|\Lambda_k| \leq f(\mathsf{I}_M) + b_k \ln|\Lambda_k| \leq f(\mathsf{I}_M) + b \ln\|\Lambda_k\|_2. \tag{B·5}$$

Define $h = \max_{i,j}\left|[\Sigma_k - \mathsf{I}_M]_{i,j}\right|$, where $[\cdot]_{i,j}$ is the operator selecting the (i, j)-element of the matrix. The pairwise bounded assumption implies that h is bounded from above. Since

$$\operatorname{Tr}(\Lambda_k \Sigma_k) = \operatorname{Tr}(\Lambda_k) + \operatorname{Tr}(\Lambda_k(\Sigma_k - \mathsf{I}_M)) \geq \operatorname{Tr}(\Lambda_k) - h\|\Lambda_k\|_1$$

holds, we deduce that

$$\|\Lambda_k\|_2 \leq \mathrm{Tr}(\Lambda_k) \leq \mathrm{Tr}(\Lambda_k \Sigma_k) + h\|\Lambda_k\|_1. \tag{B·6}$$

Combining Eqs. (B·4), (B·5), and (B·6), we have

$$g(\Lambda_k) \triangleq \|\Lambda_k\|_2 - \left(1 + \frac{d}{c}\right) b \ln \|\Lambda_k\|_2 - \left(1 + \frac{d}{c}\right) f(\mathrm{I}_M) \leq 0 \tag{B·7}$$

Since $g(\cdot)$ is a strictly convex and increases to infinity as t tends to infinity, $\|\Lambda_k\|_2$ is bounded by the largest solution of $g(t) = 0$ for $t \geq (1 + \frac{h}{c})b$. □

References

[1] J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, 2(1):1–18, 2015.

[2] T. W. Anderson. *An Introduction to Multivariate Statistical Analysis*. Wiley-Interscience, 3rd. edition, 2003.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[4] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Survey*, 41(3):1–58, 2009.

[5] L. Chapel and C. Friguet. Anomaly detection with score functions based on the reconstruction error of the kernel PCA. In *Proceedings of Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014.*, pages 227–241. Springer, 2014.

[6] T. Chen, X. Liu, B. Xia, W. Wang, and Y. Lai. Unsupervised anomaly detection of industrial robots using sliding-window convolutional variational autoencoder. *IEEE Access*, 8:47072–47081, 2020.

[7] Y. Chen, J. Li, F. Wang, K. Yue, Y. Li, B. Xing, L. Zhang, and L. Chen. Ds2pm: A data sharing privacy protection model based on blockchain and federated learning. *IEEE Internet of Things Journal*, 2021.

[8] K. Christidis and M. Devetsikiotis. Blockchains and smart contracts for the internet of things. *IEEE Access*, 4:2292–2303, 2016.

[9] B. Dai, Z. Wang, and D. Wipf. The usual suspects? reassessing blame for vae posterior collapse. In *Proceedings of the 37th International Conference on International conference on machine learning (ICML 20)*, pages 2313–2322. PMLR, 2020.

[10] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[11] J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

[12] C. Huang, Y. Chai, Z. Zhu, B. Liu, and Q. Tang. A novel distributed fault detection approach based on the variational autoencoder model. *ACS omega*, 7(3):2996–3006, 2022.

[13] T. Idé. Collaborative anomaly detection on blockchain from noisy sensor data. In *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 120–127. IEEE, 2018.

[14] T. Idé, D. T. Phan, and J. Kalagnanam. Multi-task multi-modal models for collective anomaly detection. In *Proceedings of the 17th IEEE Intl. Conf. on Data Mining (ICDM 17)*, pages 177–186, 2017.

[15] T. Idé and R. Raymond. Decentralized collaborative learning with probabilistic data protection. In *2021 IEEE International Conference on Smart Data Services (SMDS)*, pages 234–243. IEEE, 2021.

[16] T. Idé, R. Raymond, and D. T. Phan. Efficient protocol for collaborative dictionary learning in decentralized networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2585–2591, 2019.

[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[18] S. L. Lauritzen. *Graphical Models*. Oxford, 1996.

[19] W. Lee and D. Xiang. Information-theoretic measures for anomaly detection. In *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001*, pages 130–143, 2000.

[20] L. Li, J. Xiahou, F. Lin, and S. Su. Distvae: Distributed variational autoencoder for sequential recommendation. *Knowledge-Based Systems*, 264:110313, 2023.

[21] Q. Lu and X. Xu. Adaptable blockchain-based systems: a case study for product traceability. *IEEE Software*, 34(6):21–27, 2017.

[22] J. Lucas, G. Tucker, R. Grosse, and M. Norouzi. Understanding posterior collapse in generative latent variable models, 2019.

[23] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), Mar. 2007.

[24] Z. Mahmood and V. Jusas. Blockchain-enabled: Multi-layered security federated learning platform for preserving data privacy. *Electronics*, 11(10):1624, 2022.

[25] A. S. Masurkar, X. Sun, and J. Dai. Using blockchain for decentralized artificial intelligence with data privacy. In *Proceedings of the 2023 International Conference on Computing, Networking and Communications (ICNC)*, pages 195–201. IEEE, 2023.

[26] I. Mironov. Renyi differential privacy. In *Proceedings of the 30th IEEE Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[27] E. Münsing, J. Mather, and S. Moura. Blockchains for decentralized optimization of energy resources in microgrid networks. In *Proceedings of the 2017 IEEE Conference on Control Technology and Applications (CCTA)*, pages 2164–2171. IEEE, 2017.

[28] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[29] K. Noto, C. Brodley, and D. Slonim. Anomaly detection using an ensemble of feature models. In *Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM 10)*, pages 953–958. IEEE, 2010.

[30] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *Proceedings of the 34th International Conference on Machine Learning (ICML 17)*, pages 2681–2690. PMLR, 2017.

[31] D. T. Phan and T. Idé. ℓ_0-regularized sparsity for probabilistic mixture models. In *Proceedings of the 2019 SIAM International Conference on Data Mining (SDM 19)*, pages 172–180. SIAM, 2019.

[32] M. Polato. Federated variational autoencoder for collaborative filtering. In *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN 21)*, pages 1–8. IEEE, 2021.

[33] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

[34] S. Staniford, J. A. Hoagland, and J. M. McAlerney. Practical automated detection of stealthy portscans. *Journal of Computer Security*, 10(1-2):105–136, 2002.

[35] K. Toyoda, P. T. Mathiopoulos, I. Sasase, and T. Ohtsuki. A novel blockchain-based product ownership management system (POMS) for anti-counterfeits in the post supply chain. *IEEE Access*, 5:17465–17477, 2017.

[36] D. Tse, B. Zhang, Y. Yang, C. Cheng, and H. Mu. Blockchain application in food supply information security. In *Proceedings of the 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM).*, pages 1357–1361. IEEE, 2017.

[37] Y. Wang, D. Blei, and J. P. Cunningham. Posterior collapse and latent variable non-identifiability. *Advances in Neural Information Processing Systems*, 34:5443–5455, 2021.

[38] L. Xie, I. M. Baytas, K. Lin, and J. Zhou. Privacy-preserving distributed multi-task learning with asynchronous updates. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1195–1204. ACM, 2017.

[39] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, Z. Li, Y. Liu, Y. Zhao, D. Pei, Y. Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications. In *Proceedings of the 2018 world wide web conference*, pages 187–196, 2018.

[40] K. Yamanishi. *Learning with the Minimum Description Length Principle*. Springer Nature, 2023.

[41] K. Yamanishi, J. Takeuchi, G. Williams, and P. Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. In *Proceedings of the Sixth ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pages 320–324, 2000.

[42] S. Zeng, M. A. Anwar, T. T. Doan, A. Raychowdhury, and J. Romberg. A decentralized policy gradient approach to multi-task reinforcement learning. In *Proceedings of the thirty-seventh conference on Uncertainty in Artificial Intelligence (UAI 21)*, pages 1002–1012. PMLR, 2021.

[43] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola. *Dive into deep learning*. Cambridge University Press, 2023.

[44] K. Zhang, Y. Jiang, L. Seversky, C. Xu, D. Liu, and H. Song. Federated variational learning for anomaly detection in multivariate time series. In *Proceedings of the 2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, pages 1–9. IEEE, 2021.

[45] Y. Zhao, J. Chen, D. Wu, J. Teng, and S. Yu. Multi-task network anomaly detection using federated learning. In *Proceedings of the 10th international symposium on information and communication technology*, pages 273–279, 2019.

[46] Z. Zhao, X. Liang, H. Huang, and K. Wang. Deep federated learning hybrid optimization model based on encrypted aligned data. *Pattern Recognition*, 148:110193, 2024.