Social and Physical Attributes-Defined Trust Evaluation for Effective Collaborator Selection in Human-Device Coexistence Systems

Botao Zhu and Xianbin Wang
Dept. of Electrical and Computer Engineering, Western University, London, Ontario N6A 3K7 CANADA

***Abstract*—In human-device coexistence systems, collaborations among devices are determined by not only physical attributes such as network topology but also social attributes among human users. Consequently, trust evaluation of potential collaborators based on these multifaceted attributes becomes critical for ensuring the eventual outcome. However, due to the high heterogeneity and complexity of physical and social attributes, efficiently integrating them for accurate trust evaluation remains challenging. To overcome this difficulty, a canonical correlation analysis-enhanced hypergraph self-supervised learning (HSLCCA) method is proposed in this research. First, by treating all attributes as relationships among connected devices, a relationship hypergraph is constructed to comprehensively capture inter-device relationships across three dimensions: spatial attribute-related, device attribute-related, and social attribute-related. Next, a self-supervised learning framework is developed to integrate these multi-dimensional relationships and generate device embeddings enriched with relational semantics. In this learning framework, the relationship hypergraph is augmented into two distinct views to enhance semantic information. A parameter-sharing hypergraph neural network is then utilized to learn device embeddings from both views. To further enhance embedding quality, a CCA approach is applied, allowing the comparison of data between the two views. Finally, the trustworthiness of devices is calculated based on the learned device embeddings. Extensive experiments demonstrate that the proposed HSLCCA method significantly outperforms the baseline algorithm in effectively identifying trusted devices.**

***Index Terms*—CCA, hypergraph, relationship, self-supervised learning, trust**

I. Introduction

Human-device coexistence systems feature a deep integration of human social attributes with intelligent devices, enabling them to act as socially connected entities, not just computational units [1]. Human social network structures govern device deployment, connection strategies, and collaborator selection, further shaping system operation and performance. As a result, the connectivity and collaboration between devices are not only influenced by physical networks and resources, but also by social relationships among human users. For example, in the metaverse, two friends attending the same virtual concert from different physical locations can have their VR headsets establish a temporary, prioritized connection to optimize their shared experience.

In such systems, the selection of collaborators significantly affects task execution efficiency, cooperation stability, and overall system performance. Trust, as a core metric for evaluating the reliability and risk level of collaborators, is becoming essential for ensuring effective collaboration. From the perspective of task completion, trust is defined as the ability of a collaborator to fulfill the task owner's specific task requirements [2]. In existing research, some studies assess device trustworthiness by leveraging social attributes, such as interest similarity, through predefined rules or models [3]. Others focus on historical device behaviors, such as task success rates and communication stability, employing machine learning for trust evaluation [4]. However, these traditional trust evaluation models fail to deliver accurate results in human-device coexistence systems, as they are incapable of effectively capturing the complex relationships in both physical and social domains. To achieve accurate trust evaluation in such systems, the design of trust evaluation models should consider the following two key aspects: i) Multi-source trust fusion: trust evaluation models should integrate not only a device's physical capabilities and physical behavioral performance in collaboration but also its social attributes. For instance, a device possessing significant computational power might still be excluded from task allocation if it demonstrates limited social connectivity with other entities due to potential risks; ii) Semantics-enhanced representation: trust evaluation models must be capable of semantically understanding and representing both social and physical attributes to facilitate more accurate trust assessment.

Therefore, this study proposes a canonical correlation analysis-enhanced hypergraph self-supervised learning (HSLCCA) method to semantically represent and integrate both social and physical attributes for accurate trust evaluation in human-device coexistence systems. The main contributions of this paper are summarized as follows.

- We are the first to propose integrating devices' social and physical attributes for trust evaluation. By treating all attributes as relationships connecting devices, we propose a novel hypergraph-based approach to comprehensively capture multi-dimensional inter-device relationships.
- A self-supervised learning framework is proposed to fuse these multi-dimensional relationships and generate device embeddings enriched with complex semantic information.

In this learning framework, the relationship hypergraph is augmented into two distinct views to enhance semantic diversity. A parameter-sharing hypergraph neural network (HGNN) is then applied to both views to learn informative device embeddings.

- To further improve the quality of embeddings, a CCA approach is employed to compare the data from both views, optimizing the model's learning performance. Finally, the trustworthiness of devices is computed based on the learned device embeddings.

II. System Model and Problem Definition

A human-device coexistence system comprising $\boldsymbol{A} = \{a_1, \ldots, a_I\}$ devices is considered. Devices are connected through various attributes, and each attribute is regarded as a relationship connecting devices. Let $\boldsymbol{S}$ denote the set of all relationships. To ensure reliable collaboration, each device must evaluate the trustworthiness of potential collaborators based on these diverse relationships before selecting the most trustworthy one. Accordingly, trust is formally defined as follows:

Definition 1 (Relationship-defined trust): For any pair of devices $a_i, a_j \in \boldsymbol{A}$, the trust of device a_i in a_j is a measure of a_i's confidence in the reliability of a_j based on a set of relationships $\boldsymbol{S}$, which is given by

$$T_{a_i \to a_j} = TRUST(a_i, a_j, \boldsymbol{A}, \boldsymbol{S}). \tag{1}$$

According to Definition 1, trust evaluation between two devices requires not only considering their direct relationships but also accounting for the potential influence of other devices and relationships within the system. Therefore, accurate trust evaluation requires a system-level perspective that captures the influence of all interconnected relationships. The goal of trust evaluation is to identify the most reliable collaborator for a task initiator. If a_i is the task initiator, the problem of selecting the most trusted collaborator can be expressed as follows:

$$\arg \max_{a_j \in \boldsymbol{A}, a_j \neq a_i} T_{a_i \to a_j}. \tag{2}$$

To address this problem, the HSLCCA algorithm is proposed to facilitate comprehensive relationship representation, effective fusion of multi-dimensional relationships, and accurate trust evaluation. Based on the evaluation results, the most reliable collaborator can be efficiently identified.

III. Trust Evaluation via CCA-Enhanced Hypergraph Self-Supervised Learning

Due to the complexity and diversity of relationships, an effective tool is required to represent these relationships, thereby enabling accurate trust evaluation. Graphs are the most commonly used tool to represent relationships. However, they can only capture low-order point-to-point relationships between devices, such as the topology connectivity relationship, and are unable to capture high-order relationships between multiple devices, such as the common interest relationship. Hypergraphs are gaining attention as a powerful tool for modeling relationships among multiple nodes, including both low-order and high-order relationships [5]. A hypergraph $\mathcal{H} = (\mathcal{A}, \mathcal{E})$ consists of a set of nodes $\mathcal{A}$ and a set of hyperedges $\mathcal{E}$. Each hyperedge $e \in \mathcal{E}$ connects a group of nodes, capturing the relationship among them. A node $a \in \mathcal{A}$ can establish various types of relationships with other nodes within the system. The hypergraph structure can be represented by an incidence matrix $H \in \{0,1\}^{|\mathcal{A}| \times |\mathcal{E}|}$, where each entry $h_{ae} = 1$ if $a \in e$ and $h_{ae} = 0$ otherwise [6].

Leveraging the advantages of hypergraphs, this section presents HSLCCA, a hypergraph-based trust evaluation approach, for human-device coexistence systems. The proposed approach comprises three primary components: i) Hypergraph-driven relationship representation: Hypergraphs are used to capture relationships among devices, resulting in a relationship hypergraph $\mathcal{H}^{\text{all}}$; ii) CCA-enhanced hypergraph self-supervised learning: This process involves fusing multi-dimensional relationships and generating device embeddings enriched with relational semantics; iii) Trust computation: Trust between devices is calculated based on their embeddings.

A. Hypergraph-Driven Relationship Representation

To comprehensively uncover relationships, we divide them into three categories: spatial attribute-related, device attribute-related, and social attribute-related. In the following sections, we will discuss how to represent each category using hypergraphs.

1) Spatial attribute-Related: Spatial attribute-related relationships describe the spatial distribution and connectivity of devices within a physical space. These relationships encompass attributes related to network topology and physical space. For example, in large-scale systems, devices located within the same physical area are more likely to have direct network links and shorter communication paths, resulting in lower latency and more stable interactions. Devices that ensure reliable interactions typically exhibit higher trustworthiness. Therefore, incorporating these relationships into trust evaluation is essential. In this study, the spatial attribute-related relationships include the network connectivity relationship $s^{\text{net}} \in \boldsymbol{S}$ and the physical proximity relationship $s^{\text{phy}} \in \boldsymbol{S}$.

Network connectivity relationship: s^{net} indicates whether a direct communication link exists between any two devices, as directly connected devices have a higher potential for collaboration. If there is a communication link between devices a_i and a_j, we represent this connection with a hyperedge $e^{\text{net}}_{a_i,a_j} = \{a_i, a_j\}$. By transforming all direct communication links into hyperedges, a network connectivity relationship hypergraph can be obtained, denoted as $\mathcal{H}^{\text{net}} = \{\ldots, e^{\text{net}}_{a_i,a_j}, \ldots\}$.

Physical proximity relationship: s^{phy} reflects the relative positions of devices in physical space. Devices that are geographically closer often experience lower transmission

delays and more stable communication, which positively influences collaborative efficiency. To effectively capture this relationship, a clustering technique is employed to partition devices based on their physical locations. Through iterative refinement, it can group nearby devices into the same cluster while distinguishing those that are spatially distant. This clustering process can be represented as:

$$\{c_1, \ldots, c_K\} = Clustering(\boldsymbol{A}), \tag{3}$$

where $c_k, k = 1, \ldots, K$ represents a cluster. Subsequently, c_k is encapsulated by an edge to form a hyperedge e_k^{phy}. Ultimately, all clusters are transformed into hyperedges, collectively forming a physical proximity relationship hypergraph $\mathcal{H}^{\text{phy}}$.

2) Device Attribute-Related: Device attribute-based relationships reflect the relevance among devices based on their exhibited characteristics, such as resource configurations, functional roles, and behavioral patterns. For example, devices from the same manufacturer often share similar resource profiles and operational behaviors. Likewise, devices with similar functions are more likely to collaborate for task execution. In this category, two primary relationships are considered: the collaboration relationship $s^{\text{his}} \in \boldsymbol{S}$ and the resource similarity relationship $s^{\text{res}} \in \boldsymbol{S}$.

Collaboration relationship: s^{his} captures the historical cooperation performance among devices, reflecting their efficiency and compatibility. Stable and effective historical collaborations indicate strong interoperability between devices, thereby increasing the likelihood of future cooperation. If a group of devices a_i, a_j, and a_m previously collaborated to complete a task t, a hyperedge e_t^{his} is created to encapsulate these three devices, representing the collaboration. The weight of the hyperedge indicates the effectiveness of the collaboration, where 1 denotes a successful collaboration and 0 denotes a failed collaboration. All historical collaborations are converted into corresponding hyperedges to form the collaboration relationship hypergraph $\mathcal{H}^{\text{his}} = \{\ldots, e_t^{\text{his}}, \ldots\}$.

Resource similarity relationship: s^{res} indicates the similarity of devices in terms of computing power, storage, and other aspects. Devices with similar resources tend to exhibit similar operating modes and behavioral characteristics. Since there are many types of devices in the system, for simplification, we categorize them based on basic device types, such as smartphone, computer, etc. Assuming that a_i, a_j, and a_m are smartphones, a hyperedge e^{res} can be used to encapsulate these three devices, indicating that they belong to the same device type. Ultimately, all such hyperedges representing different device types can be obtained, collectively forming a resource similarity relationship hypergraph $\mathcal{H}^{\text{res}}$.

3) Social Attribute-Related: The above two categories explore the relationships between devices from a physical perspective. However, collaboration between devices extends beyond these relationships, as they also influence each other through complex social attributes. For example, devices may build trust through common partners, forming social groups or circles.

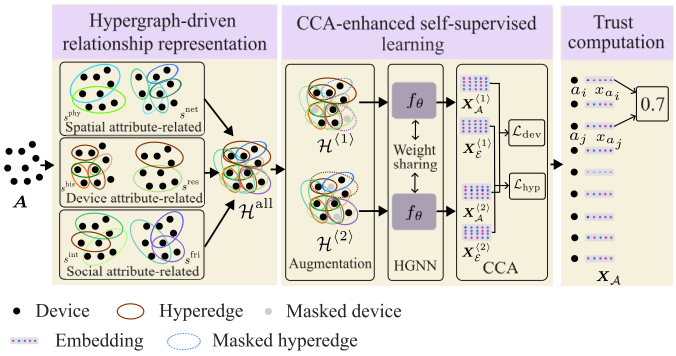


Fig. 1. The proposed HSLCCA method, including relationship representation, fusion, and trust computation.

Therefore, social attributes are essential for comprehensive trust evaluation. In this study, two social attribute-based relationships are considered: the common interest relationship $s^{\text{int}} \in \boldsymbol{S}$ and the common friend relationship $s^{\text{fri}} \in \boldsymbol{S}$.

Common interest relationship: s^{int} refers to the similarity between devices in terms of specific functions, use cases, or user preferences, such as health monitoring or security surveillance. Devices with common interests are more likely to collaborate effectively. Assuming the system contains a total of B interests, each interest $b \in B$ can be represented as a hyperedge, denoted by e_b^{int}, that encapsulates all devices sharing that interest. Ultimately, the collection of all hyperedges forms a common interest relationship hypergraph $\mathcal{H}^{\text{int}} = \{\ldots, e_b^{\text{int}}, \ldots\}$.

Common friend relationship: s^{fri} refers to the connection between two devices established by sharing the same friend. For instance, if device a_i maintains connections with both devices a_j and a_m, then a_j and a_m are considered to have a common friend relationship through a_i. This relationship can be represented by a hyperedge e_{a_j,a_m}^{fri} that connects the indirectly linked devices. The set of all such hyperedges constitutes the common friend relationship hypergraph, denoted as $\mathcal{H}^{\text{fri}} = \{\ldots, e_{a_j,a_m}^{\text{fri}}, \ldots\}$.

To integrate all relationships, the obtained hypergraphs are combined to generate a new relationship hypergraph $\mathcal{H}^{\text{all}} = \mathcal{H}^{\text{net}} \cup \mathcal{H}^{\text{phy}} \cup \mathcal{H}^{\text{his}} \cup \mathcal{H}^{\text{res}} \cup \mathcal{H}^{\text{int}} \cup \mathcal{H}^{\text{fri}} = (\mathcal{A}, \mathcal{E})$, where $\mathcal{A} = \boldsymbol{A}$, and $\mathcal{E}$ is the set of all hyperedges. For consistency in notation, all hyperedges are re-expressed as $\mathcal{E} = \{e_n\}_{n=1}^{|\mathcal{E}|}$. Each hyperedge is associated with a weight w_n, and the matrix of weights is $\boldsymbol{W} \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$. The feature matrix of all devices in $\mathcal{H}$ is represented as $\boldsymbol{X}_{\mathcal{A}} \in \mathbb{R}^{|\mathcal{A}|\times d}$, and the incidence matrix of $\mathcal{H}$ is given by $\boldsymbol{H} \in \mathbb{R}^{|\mathcal{A}|\times|\mathcal{E}|}$. The degree of devices is denoted by the diagonal matrix $\boldsymbol{D}_a \in \mathbb{R}^{|\mathcal{A}|\times|\mathcal{A}|}$, where each element $\delta(a_i) = \sum_{e_n\in\mathcal{E}} w_n h(a_i, e_n)$. The degree of hyperedges is denoted by the diagonal matrix $\boldsymbol{D}_e \in \mathbb{R}^{|\mathcal{E}|\times|\mathcal{E}|}$, where each element $\delta(e_n) = \sum_{a_i\in e_n} h(a_i, e_n)$ representing the number of devices connected by e_n.

B. CCA-Enhanced Hypergraph Self-Supervised Learning

Although $\mathcal{H}^{\text{all}}$ captures all relationships between devices, trust between any pair of devices cannot be directly calculated from these relationships. To address this issue, devices and their relationships need to be mapped into the same dimensional space. Hypergraph learning can achieve this goal, which uses a function $f_\theta : \mathcal{H}^{\text{all}} \rightarrow (\boldsymbol{X}_\mathcal{A}, \boldsymbol{X}_\mathcal{E})$ to learn the devices' embeddings $\boldsymbol{X}_\mathcal{A}$ and hyperedges' embeddings $\boldsymbol{X}_\mathcal{E}$ within the same dimensional space. To train f_θ, we use the CCA-enhanced self-supervised learning approach. The core idea is to generate two augmented views from the raw data, offering different contexts or semantics, and then train a model through self-supervised learning to maximize the agreement between the two views. Therefore, the model training process comprises three steps: 1) Two data views are generated through hypergraph augmentation; 2) Hypergraph embedding is performed using HGNN; 3) The model is trained via CCA-enhanced self-supervised learning.

1) Hypergraph Augmentation: To create two augmented views, two types of data augmentation are utilized: device masking and membership masking. In device masking, devices are randomly masked with a certain probability, which is given by

$$Aug(\mathcal{A}, p^{\mathcal{A}}) = \mathcal{A} \odot \boldsymbol{M}^{\mathcal{A}}, \tag{4}$$

where $\odot$ represents element-wise multiplication, $\boldsymbol{M}^{\mathcal{A}}$ is the mask matrix in which each element is independently sampled from a Bernoulli distribution $\mathcal{B}(1-p^{\mathcal{A}})$, and $p^{\mathcal{A}}$ is the probability of devices being masked. Membership masking refers to randomly removing some node-hyperedge memberships from the hypergraph, which is formulated as

$$Aug(\boldsymbol{H}, p^{\boldsymbol{H}}) = \boldsymbol{H} \odot \boldsymbol{M}^{\boldsymbol{H}}, \tag{5}$$

where $\boldsymbol{M}^{\boldsymbol{H}}$ is the masking matrix whose elements are sampled from a Bernoulli distribution $\mathcal{B}(1-p^{\boldsymbol{H}})$, and $p^{\boldsymbol{H}}$ is the masking probability. Finally, we can obtain two views of $\mathcal{H}^{\text{all}}$

$$\mathcal{H}^{\langle 1\rangle} = (Aug(\mathcal{A}, p^{\mathcal{A}}), Aug(\boldsymbol{H}, p^{\boldsymbol{H}})), \tag{6}$$

$$\mathcal{H}^{\langle 2\rangle} = (Aug(\mathcal{A}, p^{\mathcal{A}}), Aug(\boldsymbol{H}, p^{\boldsymbol{H}})). \tag{7}$$

2) Hypergraph Embedding via HGNN: To generate device and hyperedge embeddings for the two augmented views, a parameter-sharing HGNN is employed. Each view is fed into the HGNN, which applies a two-stage neighborhood aggregation process: device-to-hyperedge and hyperedge-to-device. HGNN iteratively updates the representation of each hyperedge by aggregating representations of its incident devices, which is given by

$$\boldsymbol{x}_{e_n}^{(l)} = f_{\mathcal{A}\rightarrow\mathcal{E}}^{(l)}\left(\boldsymbol{x}_{e_n}^{(l-1)}, \left\{\boldsymbol{x}_{a_i}^{(l-1)} : a_i \in e_n\right\}\right), \tag{8}$$

where $\boldsymbol{x}_{e_n}^{(l-1)}$ and $\boldsymbol{x}_{a_i}^{(l-1)}$ are the embeddings of e_n and a_i at layer $(l-1)$, respectively. In addition, the representation of each device is updated iteratively through aggregating representations of its incident hyperedges

$$\boldsymbol{x}_{a_i}^{(l)} = f_{\mathcal{E}\rightarrow\mathcal{A}}^{(l)}\left(\boldsymbol{x}_{a_i}^{(l-1)}, \left\{\boldsymbol{x}_{e_n}^{(l)} : a_i \in e_n\right\}\right). \tag{9}$$

Formally, equations (8) and (9) in the l-th layer of HGNN can be represented in matrix form

$$\boldsymbol{X}_\mathcal{E}^{(l)} = \phi\left(\boldsymbol{D}_e^{-1}\boldsymbol{H}^T\boldsymbol{X}_\mathcal{A}^{(l-1)}\boldsymbol{\Theta}_\mathcal{E}^{(l)}\right), \tag{10}$$

$$\boldsymbol{X}_\mathcal{A}^{(l)} = \phi\left(\boldsymbol{D}_a^{-1}\boldsymbol{H}\boldsymbol{W}\boldsymbol{X}_\mathcal{E}^{(l)}\boldsymbol{\Theta}_\mathcal{A}^{(l)}\right), \tag{11}$$

where $\boldsymbol{X}_\mathcal{E}^{(l)} \in \mathbb{R}^{|\mathcal{E}|\times d}$ and $\boldsymbol{X}_\mathcal{A}^{(l)} \in \mathbb{R}^{|\mathcal{A}|\times d}$ are hyperedge and device embeddings at the l-th layer. d is the embedding dimensionality. $\boldsymbol{\Theta}_\mathcal{E}^{(l)}$ and $\boldsymbol{\Theta}_\mathcal{A}^{(l)}$ are trainable parameters for $f_{\mathcal{A}\rightarrow\mathcal{E}}^{(l)}$ and $f_{\mathcal{E}\rightarrow\mathcal{A}}^{(l)}$, respectively. $\phi(\cdot)$ denotes a nonlinear activation function. Finally, HGNN outputs the device and hyperedge embeddings for each augmented view, denoted as $(\boldsymbol{X}_\mathcal{A}^{\langle 1\rangle}, \boldsymbol{X}_\mathcal{E}^{\langle 1\rangle})$ for $\mathcal{H}^{\langle 1\rangle}$ and $(\boldsymbol{X}_\mathcal{A}^{\langle 2\rangle}, \boldsymbol{X}_\mathcal{E}^{\langle 2\rangle})$ for $\mathcal{H}^{\langle 2\rangle}$.

3) CCA-Enhanced Self-Supervised Learning: To obtain more meaningful device embeddings, it is necessary to perform comparison and self-supervised learning on the two augmented views. Traditional contrastive self-supervised learning methods require a large number of negative samples, resulting in high computational overhead and efficiency challenges. CCA has gained increasing attention in multi-view learning due to its capability to capture both linear and non-linear relationships between views without the need for negative sampling, thereby significantly reducing computational complexity. As a multivariate analysis method, CCA aims to maximize the correlation between data representations from different views, enhancing the effectiveness of multi-view learning models [7]. Therefore, we use CCA to design two optimization objectives for self-supervised learning to maximize the agreement between two augmented views.

Device-level CCA optimization objective: The device embeddings are first normalized along the instance dimension such that each feature dimension has zero mean and a standard deviation of $1/\sqrt{|\mathcal{A}|}$

$$\widehat{\boldsymbol{X}}_\mathcal{A}^{\langle v\rangle} = \frac{\boldsymbol{X}_\mathcal{A}^{\langle v\rangle} - \mu(\boldsymbol{X}_\mathcal{A}^{\langle v\rangle})}{\sigma(\boldsymbol{X}_\mathcal{A}^{\langle v\rangle})\sqrt{|\mathcal{A}|}}, v = 1, 2, \tag{12}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ denote computing mean value and standard deviation, respectively. Then, we construct the device-level CCA loss function

$$\mathcal{L}_{\text{dev}} = \mathcal{L}_{\text{dev}}^{\text{inv}} + \lambda_{\text{dev}}\mathcal{L}_{\text{dev}}^{\text{dec}}, \tag{13}$$

where λ_{dev} is a non-negative hyperparameter trading off two terms, and $\mathcal{L}_{\text{dev}}^{\text{inv}}$ and $\mathcal{L}_{\text{dev}}^{\text{dec}}$ are the invariance term and the decorrelation term, respectively. $\mathcal{L}_{\text{dev}}^{\text{inv}}$ preserves the device-wise invariant information, which is given by

$$\mathcal{L}_{\text{dev}}^{\text{inv}} = \| \widehat{\boldsymbol{X}}_\mathcal{A}^{\langle 1\rangle} - \widehat{\boldsymbol{X}}_\mathcal{A}^{\langle 2\rangle} \|_F^2, \tag{14}$$

where $\| \cdot \|_F^2$ represents the square of Frobenius norm. $\mathcal{L}_{\text{dev}}^{\text{dec}}$ is used to prevent dimension collapse, which is defined as

$$\mathcal{L}_{\text{dev}}^{\text{dec}} = \| (\widehat{\boldsymbol{X}}_\mathcal{A}^{\langle 1\rangle})^\top \widehat{\boldsymbol{X}}_\mathcal{A}^{\langle 1\rangle} - \boldsymbol{I} \|_F^2 + \| (\widehat{\boldsymbol{X}}_\mathcal{A}^{\langle 2\rangle})^\top \widehat{\boldsymbol{X}}_\mathcal{A}^{\langle 2\rangle} - \boldsymbol{I} \|_F^2, \tag{15}$$

where $\boldsymbol{I} \in \mathbb{R}^{d\times d}$ is the identity matrix.

Hyperedge-level CCA optimization objective: This aims to distinguish the representations of the same hyperedge

across the two augmented views from other hyperedges, helping the model retain hyperedge-wise information. Similar to the device-level optimization objective, the hyperedge-level optimization objective can be formulated by the following equations

$$\widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle v\rangle} = \frac{\boldsymbol{X}_{\mathcal{E}}^{\langle v\rangle} - \mu(\boldsymbol{X}_{\mathcal{E}}^{\langle v\rangle})}{\sigma(\boldsymbol{X}_{\mathcal{E}}^{\langle v\rangle})\sqrt{|\mathcal{E}|}}, v = 1, 2, \tag{16}$$

$$\mathcal{L}_{\text{hyp}} = \mathcal{L}_{\text{hyp}}^{\text{inv}} + \lambda_{\text{hyp}}\mathcal{L}_{\text{hyp}}^{\text{dec}}, \tag{17}$$

$$\mathcal{L}_{\text{hyp}}^{\text{inv}} = \| \widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle 1\rangle} - \widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle 2\rangle} \|_F^2, \tag{18}$$

$$\mathcal{L}_{\text{hyp}}^{\text{dec}} = \| (\widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle 1\rangle})^\top \widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle 1\rangle} - \boldsymbol{I} \|_F^2 + \| (\widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle 2\rangle})^\top \widehat{\boldsymbol{X}}_{\mathcal{E}}^{\langle 2\rangle} - \boldsymbol{I} \|_F^2, \tag{19}$$

where $\mathcal{L}_{\text{hyp}}$, $\mathcal{L}_{\text{hyp}}^{\text{inv}}$, and $\mathcal{L}_{\text{hyp}}^{\text{dec}}$ are the hyperedge-level CCA loss, hyperedge-wise invariant term, and hyperedge-wise decorrelation term, respectively. By integrating (13) and (17), the total loss is formulated as

$$\mathcal{L} = \mathcal{L}_{\text{dev}} + \lambda_1\mathcal{L}_{\text{hyp}} + \lambda_2 \| \Theta \|^2, \tag{20}$$

where λ_1 and λ_2 are the weights of $\mathcal{L}_{\text{hyp}}$ and L_2 regularization, respectively. Θ is the set of learnable parameters in HGNN. The proposed HSLCCA jointly optimizes device-level CCA loss and hyperedge-level CCA loss, which enables the learned embeddings of devices and hyperedges to preserve both the device- and hyperedge-level structural information.

C. Trust Computation

After obtaining the embeddings of devices, their trust values can be directly calculated based on these embeddings, as the inter-device relationships have been fused within the embedding space. The trust of device a_i in device a_j is obtained by calculating the similarity between a_i's embedding $\boldsymbol{x}_{a_i}$ and a_j's embedding $\boldsymbol{x}_{a_j}$

$$T_{a_i \to a_j} = \frac{\boldsymbol{x}_{a_i} \cdot \boldsymbol{x}_{a_j}}{\| \boldsymbol{x}_{a_i} \| \| \boldsymbol{x}_{a_j} \|}, \boldsymbol{x}_{a_i}, \boldsymbol{x}_{a_j} \in \boldsymbol{X}_{\mathcal{A}}. \tag{21}$$

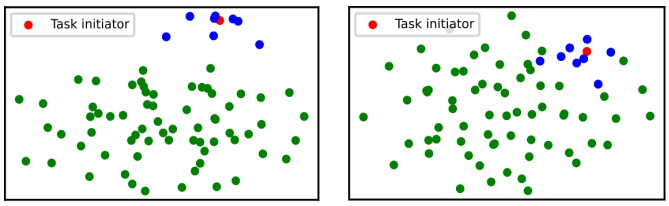
Finally, a_i evaluates the trust values of all potential collaborators and selects the one with the highest trust value as the trusted collaborator.

IV. Experiments

A. Experimental Setup

1) Dataset: To assess the effectiveness of the proposed HSLCCA method, we utilize the Sigcomm-2009 dataset [8]. The dataset includes rich information associated with devices or users, such as friendships, shared interests, activities, and message logs. It comprises 76 nodes and 18,226 interaction records collected over a period of four days. As the original dataset lacks geographic coordinates for the nodes, we assign each node a position in a two-dimensional space. Because the dataset is generated by only one type of phone, we define three distinct phone types and randomly assign one type to each node.

2) Hyperparameters: Both $p^{\boldsymbol{H}}$ and $p^{\boldsymbol{\mathcal{A}}}$ are set to 0.5. λ_{dev} is set to 0.0002, and λ_{hyp} is set to 0.0035 [9]. The weights λ_1 and λ_2 are assigned values of 1 and 0.05, respectively. The embedding size is set to 512. HGNN is trained using the Adam optimizer with a weight decay of 10^{-5}. The proposed model is implemented by Python and PyTorch.



(a) HSLCCA (b) TCH

Fig. 2. Comparison of t-SNE visualization of the device embeddings produced by HSLCCA and TCH. (The red node represents the task initiator, the blue points are the top 8 nodes with the highest trust values evaluated by the task initiator, and the green points represent other nodes.)

B. Comparison of t-SNE Visualization of Device Embeddings

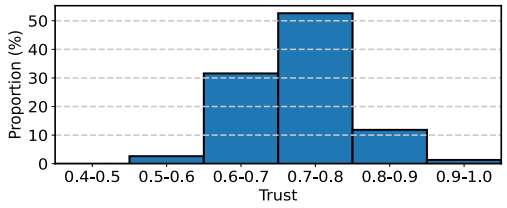
Since the device embeddings generated by the proposed HSLCCA contain complex relationships, we visually compare these embeddings with those generated by TCH [10] in this subsection. All high-dimensional embeddings are projected into a two-dimensional space using the t-SNE tool, which preserves the relative proximity of similar data points during dimensionality reduction, enabling a more intuitive visualization of the differences. Node 5 is chosen as the task initiator, and the top 8 nodes with the highest trust values evaluated by the task initiator are marked in blue. As shown in Fig. 2, compared with the TCH method, the HSLCCA method brings these top trusted nodes closer to the task initiator, forming a compact cluster. This indicates that the HSLCCA method more effectively captures the structure and semantic proximity in the relationships.

To quantitatively assess clustering quality, the silhouette score (SS) is employed. An SS value close to 1 indicates well-separated and compact clusters, reflecting strong intra-cluster cohesion and clear inter-cluster distinction. Conversely, a negative SS value suggests that nodes are incorrectly assigned. The SS value obtained by HSLCCA is 0.41, while the TCH method yields a value of 0.01. This significant difference demonstrates that HSLCCA excels at distinguishing the most trusted nodes from the others, leading to more accurate trust evaluation.

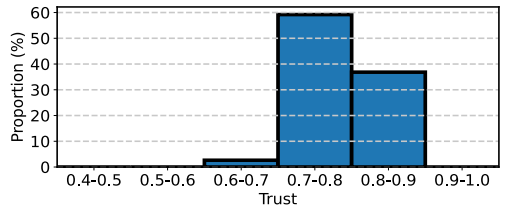
Fig. 3 presents a comparison of the trust value distribution of nodes, where the horizontal axis denotes trust value intervals and the vertical axis indicates the proportion of nodes within each interval. It is observed that the TCH method yields high trust values predominantly in the (0.8, 0.9] interval. In contrast, the HSLCCA method identifies a small number of nodes with trust values in the higher (0.9, 1.0] interval. This result further highlights the effectiveness of HSLCCA in distinguishing the most trusted nodes.

C. Sensitivity Analysis of $p^{\boldsymbol{H}}$ and $p^{\mathcal{A}}$

Fig. 4 shows the impact of $p^{\boldsymbol{H}}$ and $p^{\mathcal{A}}$, with each grid value representing a SS value. As we can see, when both $p^{\mathcal{A}}$ and $p^{\boldsymbol{H}}$ fall within the range of 0.4 to 0.7, the SS value is relatively high, indicating that nodes with high trust values can be effectively identified. When $p^{\mathcal{A}}$ and $p^{\boldsymbol{H}}$ are small, the



(a) HSLCCA



(b) TCH

Fig. 3. Comparison of the trust value distribution of nodes.

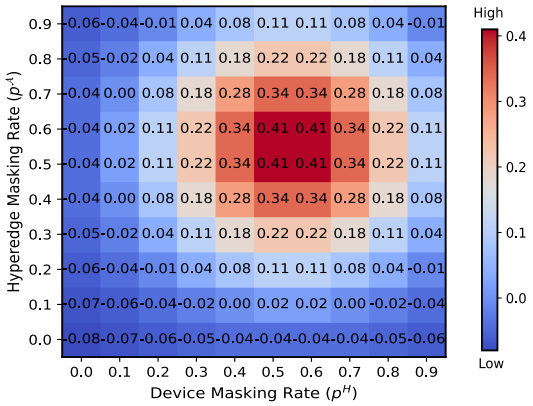


Fig. 4. Impact of $p^{\mathcal{A}}$ and $p^{\boldsymbol{H}}$ on SS.

two views become similar, which is insufficient to capture the discriminant ability of HGNN. Conversely, when $p^{\mathcal{A}}$ and $p^{\boldsymbol{H}}$ are large, the underlying semantics of the original relationship hypergraph are broken, resulting in ineffective identification of trusted nodes.

D. Comparison of selected trustworthy nodes when changing the number of nodes

In this subsection, the influence of the total number of nodes on the selection of the most trusted node is examined. As shown in Fig. 5, the x-axis represents the total number of nodes in the system, while the numbers above the bars indicate the identifiers of the selected most trusted nodes. For example, when the total number of nodes is 30, the HSLCCA method identifies node 16 as the most trusted, with a trust value of 0.93. As the number of nodes increases, the most trusted nodes selected by both methods also vary. Nevertheless, the nodes selected by HSLCCA consistently exhibit the highest trust values. Furthermore, node 29 is selected as the most trusted node at both 60 and 70 nodes by HSLCCA, despite slight fluctuations in its trust value. This indicates that an increase in the number of nodes enriches the social relationship structure,

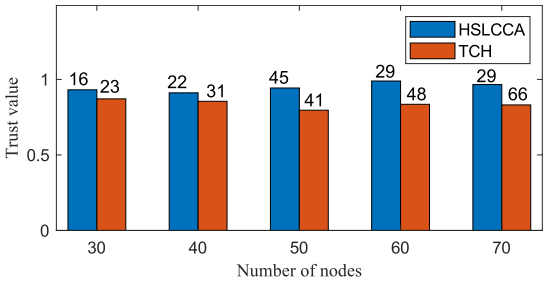


Fig. 5. Comparison of selected trustworthy nodes.

thereby influencing the trust evaluation outcomes.

V. Conclusion

To ensure reliable collaboration in human-device coexistence systems, this paper proposes the HSLCCA approach, which integrates attribute representation, fusion, and trust evaluation to identify trustworthy collaborators. First, hypergraphs are employed to explore and represent the complex relationships among devices from three dimensions. Subsequently, the CCA-enhanced hypergraph self-supervised learning is used to fuse and learn these complex relationships, ultimately incorporating latent relationship information into the device embeddings. Finally, trust values between devices are determined based on their embeddings. Extensive experiments demonstrated that the proposed HSLCCA method can effectively distinguish between trusted and untrusted nodes, and consistently select the most trusted nodes compared to the baseline algorithm.

References

[1] M. A. Díaz *et al.*, "Human-in-the-loop optimization of wearable robotic devices to improve human–robot interaction: a systematic review," *IEEE Trans. Cybern.*, vol. 53, no. 12, pp. 7483-7496, Dec. 2023.

[2] B. Zhu, X. Wang, L. Zhang, and X. S. Shen, "Chain-of-trust: A progressive trust evaluation framework enabled by Generative AI," *IEEE Netw.*, vol. 39, no. 5, pp. 44-50, Sept. 2025.

[3] P. Dong, J. Ge, X. Wang, and S. Guo, "Collaborative edge computing for social internet of things: applications, solutions, and challenges," *IEEE Trans. Comput. Social Syst.*, vol. 9, no. 1, pp. 291-301, Feb. 2022.

[4] D. Zhang, F. R. Yu, R. Yang, and L. Zhu, "Software-defined vehicular networks with trust management: A deep reinforcement learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 1400-1414, Feb. 2022,

[5] B. Zhu and X. Wang, "Networked physical computing: A new paradigm for effective task completion via hypergraph aided trusted task-resource matching," *IEEE Trans. Netw. Sci. Eng.*, Jul. 2025, Early Access, doi: 10.1109/TNSE.2025.3592859.

[6] B. Zhu and X. Wang, "Hypergraph-aided task-resource matching for maximizing value of task completion in collaborative IoT systems," *IEEE Trans. Mobile Comput.*, vol. 23, no. 12, pp. 12247–12261, Dec. 2024.

[7] H. Zhang, Q. Wu, J. Yan, D. Wipf, and P. S. Yu, "From canonical correlation analysis to self-supervised graph neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2021, pp. 76-89.

[8] A. Pietiläinen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot, "Mobiclique: middleware for mobile social networking," in *Proc. ACM Workshop Online Social Netw. (WOSN)*, New York, USA, 2009, pp. 49–54.

[9] X. Yang, W. Liu, W. Liu, and D. Tao, "A Survey on canonical correlation analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2349-2368, Jun. 2021.

[10] S. Sagar *et al.*, "Trust computational heuristic for social internet of things: a machine learning-based approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Dublin, Ireland, 2020, pp. 1-6.