Deep Heterogeneous Contrastive Hyper-Graph Learning for In-the-Wild Context-Aware Human Activity Recognition

WEN GE*, GUANYI MOU*, EMMANUEL O. AGU, and KYUMIN LEE

Human Activity Recognition (HAR) is a challenging, multi-label classification problem as activities may co-occur and sensor signals corresponding to the same activity may vary in different contexts (e.g., different device placements). This paper proposes a Deep Heterogeneous Contrastive Hyper-Graph Learning (DHC-HGL) framework that captures heterogenous Context-Aware HAR (CA-HAR) hypergraph properties in a message-passing and neighborhood-aggregation fashion. Prior work only explored homogeneous or shallow-node-heterogeneous graphs. DHC-HGL handles heterogeneous CA-HAR data by innovatively 1) Constructing three different types of sub-hypergraphs that are each passed through different custom HyperGraph Convolution (HGC) layers designed to handle edge-heterogeneity and 2) Adopting a contrastive loss function to ensure node-heterogeneity. In rigorous evaluation on two CA-HAR datasets, DHC-HGL significantly outperformed state-of-the-art baselines by 5.8% to 16.7% on Matthews Correlation Coefficient (MCC) and 3.0% to 8.4% on Macro F1 scores. UMAP visualizations of learned CA-HAR node embeddings are also presented to enhance model explainability. Our code is publicly available[1] to encourage further research.

CCS Concepts: • **Human-centered computing** → **Ubiquitous and mobile computing**; • **Computing methodologies** → *Supervised learning*.

Additional Key Words and Phrases: human activity recognition, heterogeneous graph, hypergraph, graph neural networks

ACM Reference Format:
Wen Ge, Guanyi Mou, Emmanuel O. Agu, and Kyumin Lee. 2023. Deep Heterogeneous Contrastive Hyper-Graph Learning for In-the-Wild Context-Aware Human Activity Recognition. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 4, Article 159 (December 2023), 23 pages. https://doi.org/10.1145/3631444

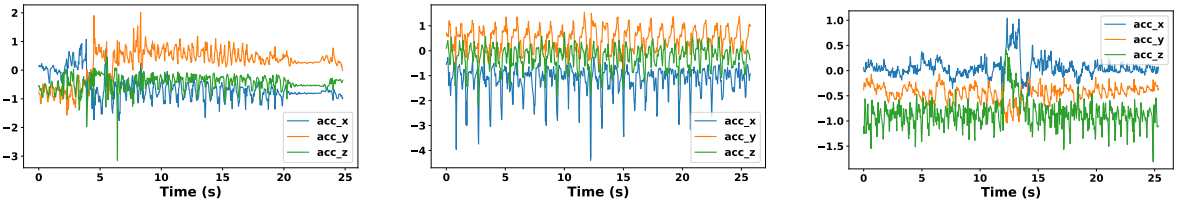
1 INTRODUCTION

Human Activity Recognition (HAR) [35], which involves recognizing human activities from sensor signals generated by Inertial Measurement Units (IMUs) attached to human bodies, is an important task in Context-Aware systems and has diverse real-world applications [8, 24]. HAR has drawn great interest from both academia and industry domains due to the nearly ubiquitous ownership of smart devices that contain various types of sensors [14–16, 41]. HAR is a challenging problem for several reasons. First, activities may co-occur (e.g., sitting and talking simultaneously, as shown in Fig. 2), making HAR a challenging multi-label classification problem. Secondly, some instances may have some missing labels (i.e., the ground truth of some labels is unknown). Moreover, sensor signals for the same activity may vary significantly due to the impact of contextual factors such as phone placement [7, 19] and user performance style [3, 9], as shown in Fig. 1.

*Equal Contribution.
[1]https://github.com/GMouYes/DHC_HGL

Authors' address: Wen Ge, wge@wpi.edu; Guanyi Mou, gmou@wpi.edu; Emmanuel O. Agu, emmanuel@wpi.edu; Kyumin Lee, kmlee@wpi.edu.

2474-9567/2023/12-ART159 $15.00
https://doi.org/10.1145/3631444



(a) $User_1$ *walking* with *phone in bag* (b) $User_1$ *walking* with *phone in hand* (c) $User_2$ *walking* with *phone in hand*

Fig. 1. Accelerometer signal corresponding to the Walking activity in various contexts and performers from real-world *Extrasensory* dataset [40]. Comparing Fig. 1a and Fig. 1b, we observe disparate accelerometer readings of the same activity under different contexts. Meanwhile, as we compare Fig. 1b and Fig. 1c, different users might perform the same activity differently, producing distinct sensor readings, even with the same contextual factors.

Inspired by these observations, recent research found it beneficial to incorporate the context of sensor signals and user information during data analyses and formulate the Context-Aware Human Activity Recognition (CA-HAR) task [32, 48] to effectively boost HAR performance. In a CA-HAR task, a pattern recognition model infers both human activities being performed and the user's context from a set of sensor signals. In this paper, we focus on a neural network CA-HAR model to predict the user's current activity along with their context (i.e., smartphone placement), which are provided as labels in the dataset.

Prior CA-HAR work generally falls into three categories: 1) non-graph methods, 2) feature-dependent graph methods, and 3) feature-independent graph methods. Early prior work [2, 14, 15, 41] mostly utilized non-graph methods, where researchers performed handcrafted features and then directly applied existing machine learning methods for activity recognition without trying to construct a graph. To improve performance, other researchers proposed feature-dependent graph methods [23, 29], which designed specialized graph structures such as graphs derived from geographic features. A location-based graph improves HAR performance by factoring in the location where activities are performed. For instance, people are more likely to sleep at home or perform workouts in gyms. However, location-based graphs require that users agree to allow the collection of privacy-sensitive information such as GPS coordinates. Unfortunately, prior work has found that less than 50% of users are willing to grant such access permission [11], limiting the applicability of location-based graph approaches.

Feature-independent graph approaches are more general and do not rely on specific features within a dataset. Instead, graphs are formed solely based on three common elements across all datasets: 1) the user, 2) the context (placement of a device that eventually collects signals), and 3) the activity. Thus, label correlations can be learned across instances by assimilating message-passing algorithms with their graph neural network counterparts.

In this paper, we build upon this line of feature-independent approaches, and argue that the CA-HAR task is naturally expressed as a supervised graph learning task, where nodes are associated with the <user, context, activity+> tuples, and the sensor signals are graph edges. While inferencing, given the previously unseen test sensor signal as an edge in the graph, the CA-HAR task is to classify the edge (defined by its connecting nodes) through a similarity kernel [16]. Such a CA-HAR graph differs from ordinary graph because it has both a hypergraph property (an edge can connect to more than two nodes) and a heterogeneity property (there are more than one type of nodes and edges), thus making it a special heterogeneous hypergraph. Heterogeneous hypergraphs have been leveraged in other domains for real-world applications including location-based social network modeling [44, 45], document recommendation [51], contagion analysis [38], spam detection [39], and link prediction [12]. To the best of our knowledge, only a few works have explored creating feature-independent graphs for the CA-HAR task. The most relevant prior research was conducted by Ge *et al.* [16], which proposed a

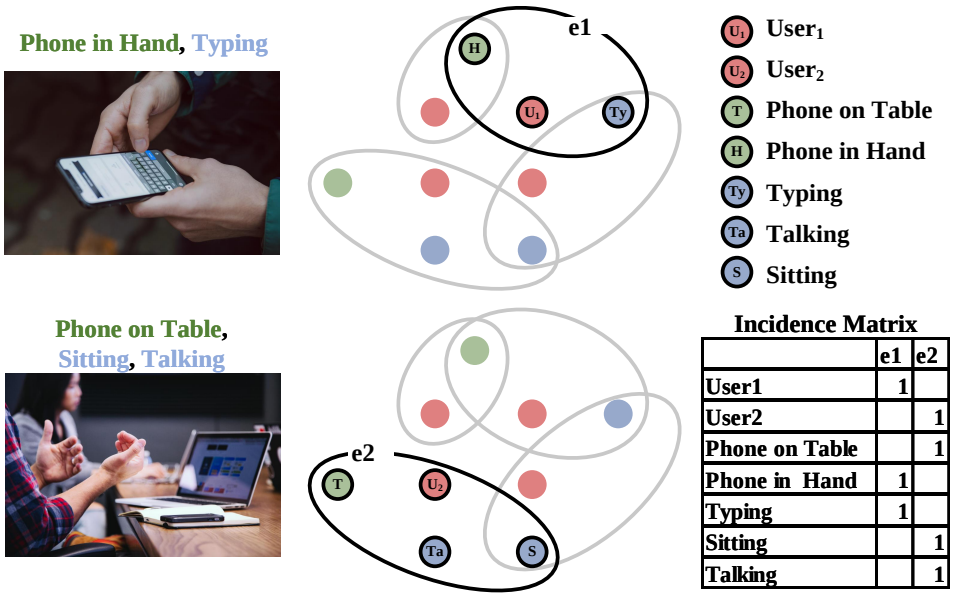


Fig. 2. Real-world examples mapping into our heterogeneous hypergraph. A CA-HAR task has three types of nodes: users u, activities a, and sensor context c. We use red, green, and blue colors to represent these nodes with heterogeneity. The first example shows $User_1$ (u_1) is *typing* (a_{Ty}) with *phone in hand* (C_H). Thus, a hyperedge connecting three nodes $\{u_1, a_{Ty}, c_H\}$ is formed to represent the situation. Another example showcased scenarios where activities may co-occur: $User_2$ (u_2) is *sitting* (a_S) and *talking* (a_{Ta}) simultaneously with *phone on table* (c_T). In this case, a hyperedge connecting four nodes $\{u_2, (a_{Ta}, a_S), c_T\}$ is needed to well-represent the situation. A corresponding incidence matrix is also shown to represent the subgraph with only these seven nodes and two hyperedges.

hypergraph convolutional neural networks based approach and utilized separate linear projections to account for node-heterogeneity. We highlight two crucial ways in which their work differs from ours:

1) **The shallow-node-heterogeneity problem.** Using separate linear projections to map node representations into sub-spaces does not necessarily guarantee sufficient node-heterogeneity. The separate linear projections may still eventually lead to similar representations. A quantitive comparison in Sec. 5 shows its sub-optimal performance, and a qualitative visualization in Fig. 10b shows that it still has inter-mixing representations across different types of nodes.

2) **Missing labels that are common in in-the-wild CA-HAR data that result in missing hyperedges in our constructed graph, are not addressed.** We present evidence that this problem is not trivial by presenting hyperedge count statistics in Fig. 7 for two real-world datasets where over 9.5% $\{u, c\}$ hyperedges in the *WASH* dataset indicate a high activity missing rate, and over 42% $\{u, a\}$ hyperedges in the *Extrasensory* dataset indicates a high context missing rate. Motivated by the observation of missing labels, which commonly occur in CA-HAR data, resulting in missing hyperedges in our constructed graph, we designated this issue as an edge-heterogeneity problem that was accommodated by constructing corresponding sub-hypergraphs.

To address the problems mentioned above, we propose a novel framework **D**eep **H**eterogeneous **C**ontrastive **H**yper-**G**raph **L**earning (DHC-HGL) to address both node-heterogeneity and edge-heterogeneity in the heterogeneous hypergraph for the CA-HAR task with the following key pieces: 1) We address explicit node-heterogeneity with contrastive loss [17] where similar samples were pulled together, and dissimilar samples were pushed apart.

In our case, we defined similar nodes as same-type nodes (i.e., both are users or activities, or contexts), while dissimilar nodes as nodes with heterogeneity (i.e., user-context pairs, user-activity pairs, or context-activity pairs). The main reason for introducing contrastive loss is to ensure node-heterogeneity from a loss/objective regularization perspective, rather than only having shallow linear projections proposed by Ge *et al.*. 2) We designed three specific hypergraph convolutional networks for the sub-hypergraphs denoted as $G_{(u,c,a)}$, $G_{(u,c)}$, and $G_{(u,a)}$, where each sub-hypergraph consists of hyperedges connecting to unique types of nodes, such that optimal layer parameters can be learned independently. This is in contrast to the unified graph learning layer and one global hypergraph utilized by Ge *et al.*. To the best of our knowledge, our work is the first to introduce contrastive loss and edge-heterogeneity toward solving the CA-HAR task. In extensive evaluation, DHC-HGL consistently outperformed various state-of-the-art models on two real-life CA-HAR datasets. The further analysis illustrated how our contrastive loss better regularized the node embedding distributions, such that node-heterogeneity was well preserved and represented through Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) visualizations, thus enhancing model interpretability and explainability for human understanding.

This work makes the following key contributions:

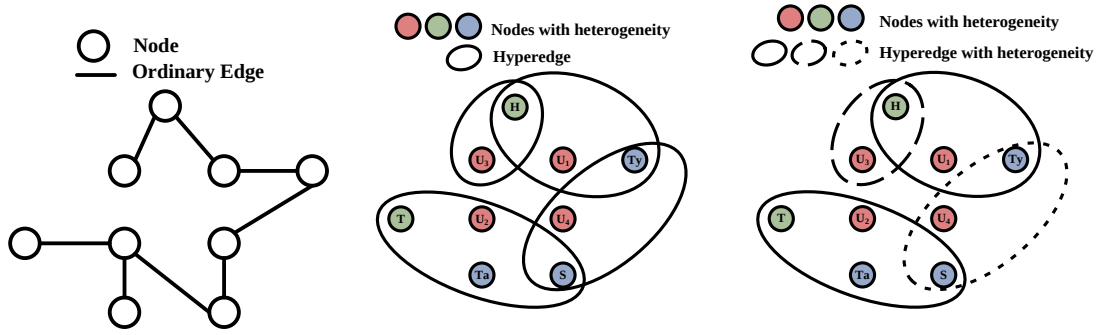
- We cast the CA-HAR task as a supervised message-passing task trained using graph neural networks. We transform CA-HAR data to a heterogeneous hypergraph with three sub-hypergraphs accounting for different label-occurrence scenarios. Our CA-HAR graph explicitly encodes the relationships between node entities, as user performance style and context (e.g., phone placement) can enhance the performance of HAR [23, 51].
- We address node-heterogeneity in the CA-HAR graph by introducing a contrastive loss, which forces the same-type node embeddings to be close together while pushing nodes from different categories further away.
- We address edge-heterogeneity caused by missing labels in the CA-HAR graph by varying edge types based on their connecting nodes and leveraging different corresponding hypergraph convolution layers with separate parameters for capturing their unique message passing patterns.
- We performed a rigorous evaluation of our novel DHC-HGL based on the proposed node-heterogeneity and edge-heterogeneity methods and achieved superior performance against various baselines, including non-graph methods, ordinary graph methods, and prior shallow heterogeneity hypergraph methods, with a large margin where the average activity recognition improvements ranged from 5.8% to 16.7% on Matthews Correlation Coefficient (MCC) and 3.0% to 8.4% on Macro F1 scores. Further ablation study verified the non-trivial contribution of each novel component of our framework.

2 PRELIMINARY

In this section, we first introduce the definitions around key concepts in graphs. We then show some illustrations around these definitions as well as real-world examples in the CA-HAR task.

Definition 1. *(Heterogeneous Hypergraph [28, 39]). A heterogeneous hypergraph $G = \{V, E, T_v, T_e, W\}$ contains vertices V and hyperedges E, whereas T_v, T_e represents the corresponding types of vertices and edges and W denotes the edge weights. Below are the related concepts:*

- *(Hyperedge in a hypergraph) Each hyperedge in a hypergraph can connect to more than two nodes, in contrast to ordinary graphs where each edge only connects to two nodes. Thus, hyperedges can be represented as set of connected nodes $e = \{v_k \mid v_k \in V\} \subseteq V$, for $e \in E$.*
- *(Graph Heterogeneity) The hypergraph is heterogeneous when $|T_v| + |T_e| > 2$. Node-Heterogeneity occurs when $T_v > 1$ and Edge-Heterogeneity occurs when $T_e > 1$.*
- *(Edge Weights) Typically, $W \in R^{|E|}$ is the matrix representing hyperedge weights.*

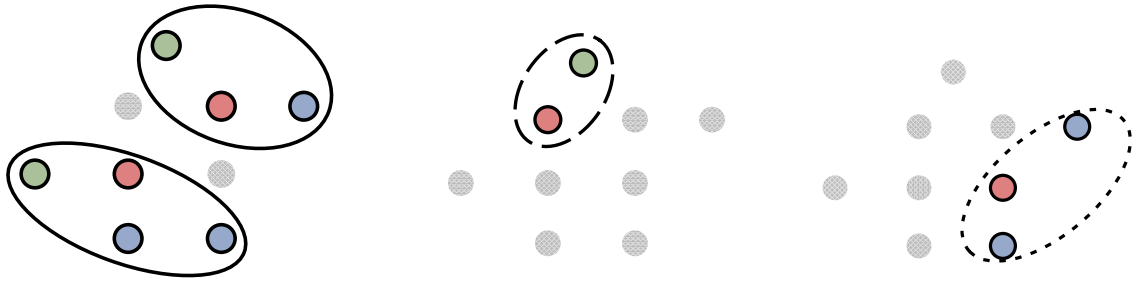


(a) Work on an ordinary graph. E.g., GCN [21].

(b) Work on hypergraph with node-heterogeneity. E.g., HHGNN [16].

(c) Work on an hypergraph with both node-heterogeneity and edge-heterogeneity (our approach).

Fig. 3. Prior work (Fig. 3a and Fig. 3b) constructs different CA-HAR graphs. Our approach (Fig. 3c) considers both node-heterogeneity and edge-heterogeneity. Intuitively, Fig. 3a is a simplified version of Fig. 3b, while Fig. 3c further generalizes beyond Fig. 3a and Fig. 3b. Fig. 4 presents an illustration with different sub-hypergraphs.



(a) Sub-hypergraph $G_{(u,c,a)}$ contains hyperedges connecting all three types of nodes (user, phone placement and activity). E.g., $User_1$ is *typing* with *phone in hand*, and $User_2$ is *sitting* and *talking* with *phone on table*.

(b) Sub-hypergraph $G_{(u,c)}$ contains a hyperedge connecting user and phone placement nodes while activity nodes can be missing. E.g., $User_1$ holds *phone in hand* and no activity label reported.

(c) Sub-hypergraph $G_{(u,a)}$ contains a hyperedge connecting user and activity nodes while phone placement nodes can be missing. E.g., $User_3$ is *typing* and *walking* simultaneously, and no context label is reported.

Fig. 4. The original heterogeneous hypergraph is further transformed into three sub-graphs that account for different types of hyperedges. In each sub-graph, node-heterogeneity is represented using different colors. However, only one type of hyperedge is incorporated. Thus, the sub-graphs capture graph information at a higher granularity.

- *(Incidence Matrix) The relationship between nodes and hyperedges can be represented by an incidence matrix* $Inc \in R^{|V|\times|E|}$ *with entries defined as:*

$$Inc(v,e) = \begin{cases} 1, if\, v \in e \\ 0, otherwise \end{cases} \tag{1}$$

We present real-world examples of the heterogeneous hypergraph properties of the CA-HAR task in Fig. 2, a comparison of graphs formed in Fig. 3, and a high-level abstraction of three sub-hypergraphs in Fig. 4. Intuitively, Fig. 3a is a simplified version of Fig. 3b, while Fig. 3c further generalizes beyond Fig. 3a and Fig. 3b. We found that real-world datasets contain edge-heterogeneity where edges may connect to different types of nodes, namely $G_{(u,c,a)}$ in Fig. 4a, $G_{(u,c)}$ in Fig. 4b, and $G_{(u,a)}$ in Fig. 4c. More specifically, 1) each hyperedge in $G_{(u,c,a)}$ connects to all three types of nodes simultaneously (i.e., user, context, and activity nodes); 2) the hyperedges in $G_{(u,c)}$ only connect to user and context nodes, and 3) $G_{(u,a)}$ contains hyperedges that are associated with only user and activity nodes. Directly applying pattern recognition models on Fig. 3b without factoring in the existence of edge-heterogeneity in Fig. 4a, 4b, and 4c can yield sub-optimal performance, as the message-passing patterns through these different hyperedges may vary. The key reason for addressing edge-heterogeneity is that it accounts for missing labels common in in-the-wild CA-HAR datasets by modeling hyperedges at a higher granularity. The inferior performance of the variant of our proposed model and HHGNN [16], which both applied a unified Graph Neural Network(GNN) without edge-heterogeneity, presented in Sections 5.5 and 6, support this claim.

3 RELATED WORK

Context-aware human activity recognition (CA-HAR) is an emerging task in academia and industry [14–16, 41], as CA-HAR is crucial in many real-world applications, such as healthcare [32], smart homes [5], and biometric authentication [52]. Prior works focused on recognizing human activities given various types of data, such as images [37] and videos [26], wearable sensors [19], smartwatches [41], and smartphone sensors [14–16, 41]. This work focuses on leveraging smartphone sensor signals for recognizing human activities, but one may extend our work to other data sources or modalities with minimum customization efforts.

More recently, many deep learning models were proposed for Context-Aware Human Activity Recognition (CA-HAR), and they mainly fall into the following categories:

Non-graph CA-HAR methods. Existing non-graph HAR methods mainly focused on feature engineering, utilizing predictive handcrafted features extracted from raw signals [16, 41], feature mining (i.e., directly learning from raw signals) [46], or combining both types of features [2, 14, 15]. In terms of deep learning frameworks, the most common methods are Multi-Layer perceptron (MLP) on handcrafted features [41], Convolutional Neural Networks (CNN) that capture spatial correlations [3, 14, 15], Recurrent Neural Networks (RNNs, LSTMs) on learning temporal dependencies [14, 15], and uncertainty measurements for mitigating data noise problems and missing label problems [14, 15, 46]. Despite exploring multiple perspectives in the HAR task, the common problem within non-graph HAR methods is they do not explicitly model the inter-entity (users, context, and activities) dependencies.

Feature/Sensor dependent graph HAR methods. Graph Neural Networks (GNNs) have been widely applied in HAR, particularly on videos in the Computer Vision domain, including learning Actor Relation Graph connections for group activity recognition [43] and utilizing spatial-temporal graphs for skeleton-based action recognition [37]. Another sensor-based HAR method proposed by Martin et al. [29] built two personalized mobility graphs (transition frequency and Euclidean distances as edge weights) using GPS sensors and applied two GCNs to recognize human activities. The derivation of the graphs depends on the availability of specific features/sensors, e.g., GPS sensors, which might violate user privacy and require users' permission. Unfortunately, less than 50% of users were willing to grant access to GPS sensors [11]. In summary, the feature/sensor-dependent graph method is a double-edged sword: they enabled models to better learn spatial and temporal correlations through a graph perspective; however, such a method also relies highly on the specific feature's existence, which limited their applicable scope in the HAR domain.

Feature independent graph methods. Unlike feature-dependent graph methods, feature-independent graph methods can construct graphs without requiring external information. Some prior work utilized fully connected

graphs [31, 49] for activities where edges are weighted through similarity kernels. Others formed graphs based on K-nearest-neighbours [33], i.e., they connect each node with the TopK nearest neighbors based on the feature embeddings [27]. Despite different designs for their downstream neural networks, they share drawbacks wherein their ordinary graphs do not contain hyperedges. Thus, they do not factor in the co-occurrence of more than two entities. Moreover, although node-heterogeneity can be incorporated into their frameworks, edge-heterogeneity was mostly overlooked.

The most relevant work to our research is from Ge *et al.* [16]. The authors transformed context-aware human activity visit patterns into a corresponding heterogeneous hypergraph.

To address the node-heterogeneity characteristic, the user, phone placement, and activity nodes were treated as different types of nodes with separate fully connected layers in-between multiple HyperGraph Convolution Layers (HGC) [4]. Despite their encouraging work, several problems remain unresolved: 1) *Shallow-node-heterogeneity*: Having separate linear projections for different types of nodes essentially sends their representation into subspaces, but there is no guarantee on the distribution of the representations. In fact, in the worst case scenario, separate linear projections can learn similar weights, thus behaving in the same way as a single unified fully connected layer without node-heterogeneity. 2) *Overlooking edge-heterogeneity*. As a direct result, the missing label problem that is common in in-the-wild CA-HAR datasets is not properly handled.

Prior Graph Neural Network (GNN) research:

Graph Neural Networks (GNNs) have attracted a lot of attention. The key idea of GNNs is to define graph nodes and egdes, that are used to generate node representations by aggregating information from the node's neighbors. For example, GCN [21] learns node embeddings from its first-order neighbors using the Kipf normalized aggregation. GraphSAGE [18] scales to large datasets by utilizing neighborhood sampling and neural networks such as Long Short Term Memory (LSTM) models to aggregate neighbor's information. GAT [42] introduces attention mechanisms that specify different importance weights to different neighbors. Recently, more GNNs have proposed various mechanisms that adapt GNNs to handle real-world applications better, including higher-order relationships between objects, and different types of nodes and edges. For instance, hyperConv [4] incorporates higher-order relationships by allowing information propagation between multiple nodes concurrently. HetGNN [47] was proposed for various graph mining tasks, including link prediction, recommendation, node classification, and clustering. It addressed node-heterogeneity by grouping different types of neighbors sampled using a Random Walk with Restart (RWR) algorithm and designing a specific module to extract heterogeneous content from each neighboring node. However, using the same restart probability for all nodes limits the expressiveness of the sampled neighbors, further limiting the expressiveness of the aggregated node representation. HERec [36], originally proposed for recommender systems, addresses data heterogeneity by using a meta-path based random walk strategy to convert a heterogeneous graph into a homogeneous graph. However, manually defining meta-paths to model these semantic relationships in heterogeneous graphs heavily relies on the quality of the designer's domain knowledge.

Novelty of our work in relation to prior work: Our work differs from prior works (especially the HHGNN model proposed by Ge *et al.* [16]) in at least two crucial ways: 1) we address node-heterogeneity with a contrastive loss objective, which pulls nodes from the same type closer by minimizing their distance and nodes from different types far apart by maximizing their distance. The key reason for using a contrastive loss function is that it explicitly regularizes the node representations, rather than implicitly projecting node representations into subspaces. Consequently, we are able to guarantee node-heterogeneity in a distance-based manner. Specifically, nodes of the same type have similar representations, while nodes of different types are distant from each other. It not only improved the model performance but also enhanced human-level understanding of the learned node representation distributions. and 2) we explicitly address edge-heterogeneity in our hypergraph framework design by distinguishing hyperedges via its connected nodes and designing separate hypergraph convolution layers to

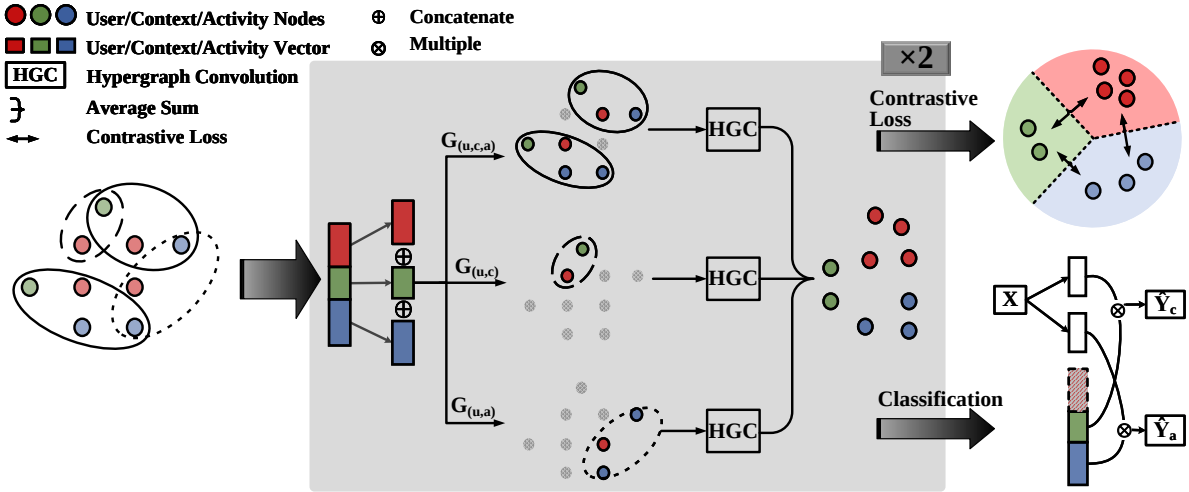


Fig. 5. Overview of our DHC-HGL Framework. It consists of two key components: the graph learning module for label encoding and the classification module for signal encoding. Given the heterogeneous hypergraph formulated (each node is defined as the user, context, and activity tuple), the model updates node representations using a GNN while factoring in node-heterogeneity (via separate projections and custom HGC layers), and edge-heterogeneity (via split subgraphs). During classification, the learned label encoding is utilized to infer connected nodes for the given signal. The objective loss function combines BCE loss for multi-label classification and contrastive loss that handles node-heterogeneity.

allow independent message propagation. The key rationale for the design is: by addressing edge-heterogeneity, we provide a solution that accounts for missing labels which are common in in-the-wild CA-HAR datasets.

4 PROPOSED FRAMEWORK

We formulate the CA-HAR problem in Section 4.1. Each component of our framework is then described in Section 4.2. A conceptual visualization of our proposed DHC-HGL is shown in Fig. 5.

4.1 Problem Formation

CA-HAR is a supervised multi-label classification task where a dataset D contains data instances $x \in X$, the corresponding data labels $y \in Y$, and the users involved $u \in U$:

$$D = \{(x, y, u) \mid x \in R^M, y \in [0, 1]^H, u \in [1, 2, ..., |U|]\} \tag{2}$$

where M is the dimension of the sensor signal, and H is the number of labels per instance, which is equivalent to the total number of contexts and activities.

$$H = |context| + |activity| \tag{3}$$

A CA-HAR model m learns an effective mapping between data instance X and data label Y:

$$m : X \rightarrow Y, m(x_i) = y_i, 0 \leq i < |X| \tag{4}$$

Under a heterogeneous hypergraph transformation, each sensor signal $x \in X$ can be associated with three types of elements: users, context, and activities, where the last two belong to Y. If each sensor signal is considered

as a hyperedge and the three types of elements are considered as graph nodes, the original dataset D will be transformed into a graph G.

$$\begin{aligned} V &= \{U\} + \{Y\} \\ E &= \{e_i\}_{0 \le i < |E|} \\ G &= (E, V) \end{aligned} \tag{5}$$

Each edge e can also be uniquely represented by their connecting nodes as a set:

$$e_i = \{v_{ik} \mid v_{ik} \in V, k = 0, 1, ...\} \tag{6}$$

Now that the new model m_G is given a hyperedge e as input, it tries to infer the set of nodes connecting to hyperedge v:

$$m_G(e) = v, e \in E, v \subseteq V \tag{7}$$

4.2 Network Design

Our network is composed of two key components: the graph learning module for label (Y) encoding, and the classification module for signal (X) encoding. At a high level, the graph learning module encodes the labels based on their co-occurrence relationship in a message-passing fashion, while the classification module encodes the data representations. During training, node representations are updated by comparing the label encodings with all training data/signal encodings. During inference, the final prediction is made via a dot product between label encodings and the test data/signal encodings, thus deriving the labels that are most similar to a given signal.

To the best of our knowledge, one of the many advantages of introducing graph learning into the CA-HAR domain is that we are able to transform simplified binary label encoding into more expressive vector representations. This goal is achieved with the help of aggregated signal information. Moreover, we are able to enhance the signal specificity of data encoding. Thus, both the macro-level and micro-level information from a given dataset can be obtained.

Our main contributions, however, are based on two key insights in improving the graph learning expressiveness that were overlooked in prior work. First, we address edge-heterogeneity to resolve the problem of missing labels as described in earlier sections. Secondly, we ensure node-heterogeneity using explicit, objective contrastive loss regularization, rather than simpler separate linear projections that may degrade to projected nodes that do not capture heterogeneity. We describe the details of our network design in the following subsections, and demonstrate their outstanding performance in Section 5.

4.2.1 Graph Learning. Given the formulated graph $G = (E, V)$, each node is initialized to the average of all instances connected to it:

$$Emb_v = mean(x_t), \text{where } y_t == 1, v \in V \tag{8}$$

The node embedding is arranged in the order of users (u), context (c), and activities (a)

$$V_g = \oplus[V_u, V_c, V_a] \tag{9}$$

where $\oplus$ represents the concatenation operation. The hyperedges are initialized to the average of sensor signal instances that have the same connecting nodes:

$$\begin{aligned} e &= \{v_k\}_{k=0,1,...}, \qquad \text{where } e \in E, v_k \in V, \{v_k\} \subseteq V \\ Emb_e &= mean(x_{kj}), \quad \text{for all } \cup \{u_{kj}, y_{kj}\}_{j=0,1,...} == \{v_k\} \end{aligned} \tag{10}$$

Given the graph G and initialized vector representations as inputs, we try to learn node representations using the graph network. To factor in node-heterogeneity, separate linear projections are then applied, followed by

non-linear activations and dropout on each type of node.

$$
\begin{aligned}
V_g^{'} &= \oplus[V_u^{'}, V_c^{'}, V_a^{'}] \\
V_t^{'} &= \delta \cdot \alpha \cdot l_t(V_t), t \in (u, c, a)
\end{aligned}
\tag{11}
$$

where δ represents a dropout operation, α is a non-linear activation function, and l_u, l_c, l_a are three linear projections corresponding to different node types. The hyperedges are then split into three groups based on the different types of nodes they are connected to, namely:

- $\{u, c, a\}$ edges: edges connect all three types of nodes
- $\{u, c\}$ edges: edges connect only user and context nodes, and no activity was provided
- $\{u, a\}$ edges: edges connect only user nodes and activity nodes, and no context was provided

The distribution of each type of hyperedge is shown in Fig. 7 for two real world datasets (i.e., *WASH Unscripted* and *Extrasensory*, with detailed descriptions in Section 5.1). Thus, there are three subgraphs $G_{(u,c,a)}$, $G_{(u,c)}$, and $G_{(u,a)}$. Each component goes through a different hyperConv layer [4] with non-linear activation and dropouts. Their results are aggregated into the whole graph through a summation function to form final node representations.

$$
\begin{aligned}
V_{gs}^{''} &= \delta \cdot \alpha \cdot HGC_s(V_{gs}^{'}) \\
V_g^{''} &= aggr(V_{gs}^{''}) \\
s &\in \{(u, a), (u, c, a), (u, c)\}
\end{aligned}
\tag{12}
$$

The combination of Eq. 9, 11 and 12 form one full layer of our heterogeneous hypergraph convolutional layer. In practice, several such layers can be stacked together for learning multi-hop neighborhood graph properties such that long-term dependencies may be better captured. Take the user node U_1 in Fig. 3b as an example: one heterogenous hypergraph convolution layer can aggregate information from its 1-hop neighbors (i.e., node H and Ty) while with two layers, we can aggregate additional information from node U_3, U_4 and S, where user node U_3 can be considered as a similar user of U_1, given the common context they share (i.e., node H). Thus the learned node embedding can capture richer information about the graph structure, which can further benefit the HAR task.

4.2.2 Classification. After learning effective node representations $V_g^{''}$, the model can infer connected nodes given hyperedge representations x (i.e., sensor feature). This procedure can be considered an edge-type classification problem. First, both node representations and the sensor signal edge are projected into the same dimensions. Then the dot product is computed for comparing vector similarities between a given sensor signal and nodes. As activities may co-occur, a sigmoid transformation on each dot product result is adopted, rather than the softmax function across results. The sigmoid results are treated as probabilities of each label as connecting nodes.

$$
\begin{aligned}
x_t^{'} &= \delta \cdot \alpha \cdot l_{xt}(x) \\
V_{gt}^{''} &= \delta \cdot \alpha \cdot l_{vt}(V_g^{''}) \\
\hat{Y} &= \oplus[\sigma \cdot V_{gt}^{''} \otimes X_t^{'}], t \in \{c, a\}
\end{aligned}
\tag{13}
$$

where $\otimes$ represents the dot product operation and σ represents the sigmoid function. l_{xt} and l_{vt} are two linear projections to map node and edge representations into the same dimension, such that a dot product can be applied to the two components.

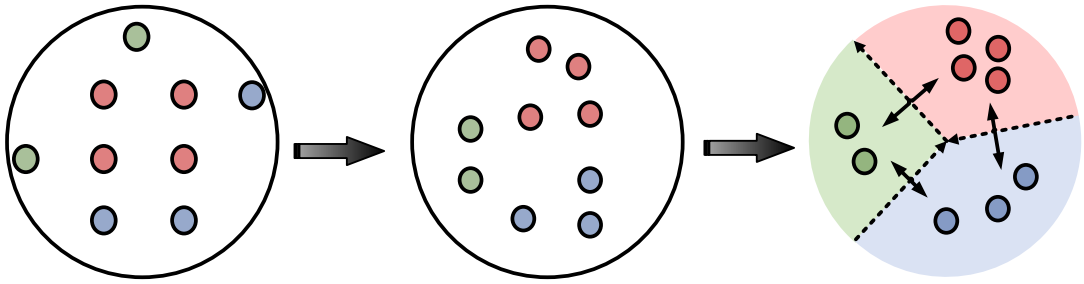


Fig. 6. Illustration of how contrastive loss improves learning of node embeddings. Left: initial node embedding formed as Eq. 8. Mid: learned node embedding w/o contrastive regularization. Right: learned node embedding w/ contrastive regularization.

4.3 Objective Loss Function

We now design a loss function that has two components: 1) A muti-label classification loss with label-wise Binary Cross Entropy (BCE) loss L_{bce} and 2) A node-heterogeneity contrastive loss L_c:

$$\begin{aligned} &\min_{\theta_g,\theta_c} \quad L = \min_{\theta_g,\theta_c} L_{BCE}(\hat{Y}, Y) + \min_{\theta_g} L_c(V_g^{''}) \\ &\text{where} \quad V_g^{''} = \theta_g(V_g), \hat{Y} = \theta_c(V_g^{''}, x) \end{aligned} \tag{14}$$

where BCE loss is commonly adopted as:

$$L_{BCE} = \frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C}[-\omega_{n,c}(y_{n,c} log \hat{y}_{n,c} + (1-y_{n,c}) log(1-\hat{y}_{n,c}))] \tag{15}$$

N is the number of instances within each batch and C is the number of classification labels.

To explicitly represent node-heterogeneity, we introduced a contrastive loss on node embeddings (Eq. 16), which essentially pulls nodes of the same type closer and pushes nodes of different types further away (Fig. 6). It calculates node embedding similarity pairwisely and pulls nodes of the same type closer by maximizing the similarity between them and vice versa. As there are a limited number of nodes (|U|+|C|+|A|), we adopted fully contrastive loss [17] that iterates over all node pairs rather than sampling-based loss [20, 34] which performs calculations based on sampled positive and negative pairs:

$$L_c = \frac{1}{H(H-1)} \sum_{V_i, V_j \in V_g^{''}, i \neq j} W(i,j) \cdot f(V_i, V_j) \tag{16}$$

where f is the distance function based on cosine similarity, and W is a weight function:

$$W(i,j) = \begin{cases} \lambda_1, & \text{if } I(i,j) == 1 \\ -\lambda_2, & \text{otherwise} \end{cases} \tag{17}$$

I is the node type identification function. It outputs 1 if two nodes have the same type. Otherwise, it outputs -1. λ_1, λ_2 are hyperparameters whose optimal values can be found using grid search.

Table 1. Context-aware Human Activity Datasets Information. The Accelerometer, Gyroscope, and Magnetometer are 3-axis sensors. If the sample rate is unspecified, the sensor is sampled once per example.

Dataset	#Instances	#Participants	#Features	Common Sensors	Unique Sensors
WASH	7,773,479	108	139	Accelerometer(40Hz), Gyroscope(40Hz) Location, Magnet(40Hz) Env. Measure, Phone State	Response
Extrasensory	6,355,350	60	170		Gravity Audio(46Hz)

Table 2. Context-aware Human Activity Labels.

Dataset	Common Context Label	Unique Context Label	#Context Label	Common Activity Label	Unique Activity Label	#Activity Label
WASH	In Pocket In Hand In Bag	On Table-Face Down On Table-Face Up	5	Lying Down, Sitting Walking, Sleeping Standing, Running Stairs-GoingDown Stairs-Going Up Exercising	Talking On Phone Bathroom Jogging, Typing	12
Extrasensory		On Table	4		Talking Bath-Shower Toilet	13

5 EXPERIMENTS

5.1 Context-aware Human Activity Recognition Datasets

We evaluate DHC-HGL on two unscripted context-aware human activity recognition datasets, namely *WASH*[2] and *Extrasensory* [40]. Unscripted datasets were collected in-the-wild, with participants running a data-gathering app on their smartphones as they lived their lives and provided context-aware activity labels periodically. Unscripted datasets are realistic and can provide better insights into participants' real-world behavior patterns. However, such datasets are quite noisy with user-provided labels that may be missing, wrong or conflicting with each other. Additionally, on different phone types, not all sensors are always available. Sensor readings may also be missing for several reasons, including weak signals and the fact that participants sometimes turn off their phones to save power or not give permission to collect data from certain sensors (e.g., GPS for privacy reasons). We describe pre-processing methods on the two datasets in Section 5.1.1. Detailed statistics of these datasets are presented in Table 1. The *Extrasensory* [40] dataset collected 20 seconds of sensor measurement per minute from smartphones and smartwatches. 60 participants from diverse ethnic backgrounds, including 34 females and 26 males, were recruited in the study for approximately one week. *WASH* followed a similar data collection and labeling methodology as *Extrasensory*, 108 users participated in a data collection study for about two weeks. Context and activity labels collected by both datasets' are listed in Table 2. We selected 17/51 labels from *Extrasensory* that are similar to labels in the *WASH* dataset to ensure a fair comparison.

5.1.1 Dataset Pre-processing. First, we resolved label conflicts by removing problematic data instances falling into the following cases: 1) Users provided more than one phone placement label resulting in co-occurring phone placement. Phone placements are mutually exclusive in the real world as a phone can be carried in only one position (e.g. in hand, back or coat pocket) at a time. 2) Multiple conflicting activity labels that cannot be performed simultaneously were provided by the study participant (co-occurred). For example, a smartphone user cannot sleep and run at the same time.

[2] https://tinyurl.com/8wvrhr7k

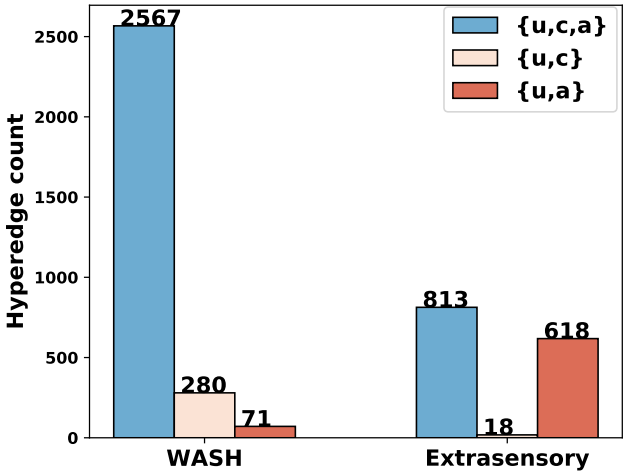


Fig. 7. Hyperedge distribution on two datasets. Specifically, $\{u, c, a\}$ represents hyperedges simultaneously connecting user, context and activity nodes, $\{u, c\}$ represents hyperedges connecting only user and context node, and $\{u, a\}$ represents hyperedges connecting only user and activity node.

Table 3. Comparison of existing CA-HAR methods.

Category	Model	Handcraft Feature	Methods	Hypergraph	Heterogeneity	
					Node	Edge
Non Graph-based method	**CRUFT**	✓	Deep Learning	-	-	-
	LightGBM	✓	Machine Learning	-	-	-
	ExtraMLP	✓	Deep Learning	-	-	-
Graph-based method	**GCN**	✓	Deep Learning			
	HHGNN	✓	Deep Learning	✓	✓	
	DHC-HGL	✓	Deep Learning	✓	✓	✓

Next, the raw multi-sensor signals were segmented into equal-sized time windows with a duration of 3 secs and a 1.5 sec step size, yielding 7,773,479 and 6,355,350 instances (Tab. 1), respectively. Next, the features extracted in many prior similar works were extracted [14–16, 41] subject to sensor availability in both datasets. Finally, we generated 139 and 170 handcrafted features after removing identical features, respectively.

Lastly, we normalized features in the train set into a range of 0 to 1 using Eq. 18 and then applied it to the test and validation sets.

$$z = (x - \mu)/s \tag{18}$$

where μ and s are the mean and standard deviation of features in the training set. x and z are the original and transformed features, respectively. As previously mentioned, the normalized features were used to initialize our graph with the aggregated mean function. The missing values in the initialized graph were filled with zeros. The number of unique hyperedges is reported in Fig. 7. It is instructive to note that in prior works, hyperedges were treated as being similar (no heterogeneity). In contrast, edge-heterogeneity was incorporated into our framework.

5.2 Baseline HAR Models

As part of our rigorous evaluation, we compared our proposed DHC-HGL performance to several state-of-the-art baselines, including non-graph based models, ordinary graph-based models, and heterogeneous hypergraph models. Table 3 summarizes the key differences between them. We provide brief descriptions of models with rationale on why they were selected in the following content.

- **CRUFT [14]:** is a state-of-the-art non-graph-based framework that achieved some of the best results till date. It jointly learns using two branches: one MLP branch analyzing handcrafted features and one CNN-BiLSTM branch analyzing raw accelerometer and gyroscope data. It exploited temporal correlation among instances by learning multiple consecutive samples together. CRUFT also incorporated an uncertainty estimation module to deal with noisy CA-HAR data collected in-the-wild. The random splitting of our data may undermine CRUFT performance as temporal correlation might not be fully revealed.
- **LightGBM [1]:** is a widely applied variant of the Gradient Boosted Machines Decision Tree (GBDT) classifier. To compute the information gain of possible split points and reduce the number of features, mutually exclusive features are bundled, and it samples data with large gradients, facilitating fast computation and high performance. We trained separate models for each context label and then combined the results. This is a non-graph-based method, which achieved good performance across many labels in prior works [13, 16].
- **ExtraMLP [41]:** was a state-of-the-art CA-HAR deep learning model, an MLP-based model that analyzed fine-grained handcrafted features extracted from both smartphone and smartwatch, and outperformed other models on the *Extrasensory* dataset in [40]. Using a multi-label formulation, it can recognize multiple co-occurring context labels and mitigate imbalanced context labels.
- **GCN [21]:** is a classic graph convolutional neural network that is essentially a simplified version of our proposed method. Two GCNs were stacked on the ordinary homogeneous graph in order to learn center node representations from 2-hop neighbors. It was included as a baseline because it previously demonstrated its ability to predict the purpose of a user's visit from geographic information (GPS-based mobility data) [29]. Unlike DHC-HGL, it considers all edges homogenous/same and does not consider the hyperedges or node/edge heterogeneity. Including it evaluated the utility of the hyperedges and heterogeneity properties of DHC-HGL.
- **HHGNN [16]:** built the same graph as we used in this paper. Node-heterogeneity and hypergraph properties were addressed by mapping different nodes with corresponding linear projection functions and applying Hypergraph Convolution Layers (HGC). However, a shared HGC was used for all types of hypergraphs, which overlooked and did not take maximal advantage of the edge heterogeneity property. Moreover, assigning specific linear functions for different kinds of nodes is an implicit and plain solution to the node-heterogeneity problem. In contrast, our DHC-HGL addressed node-heterogeneity using explicit contrastive loss in the node embedding latent space.
- **DHC-HGL**: is our proposed framework. In contrast to the above-mentioned models, it not only addressed implicit node-heterogeneity using separate linear projections, but also explicitly addressed node-heterogeneity using a contrastive loss on nodes in embedding space. Furthermore, DHC-HGL addressed edge-heterogeneity in a hypergraph by introducing separate hypergraph convolutional layers on different hyperedges and their connecting nodes.

5.3 Experimental Setup

We randomly split each dataset into 60% for training, 20% for validation, and 20% for hold-out testing. Grid search was used to determine optimal hyperparameter values for all models in our experiment. For our proposed model, the RAdam [25] optimizer was utilized. The training batch size is fixed as 1024 and the epoch size is set to 300.

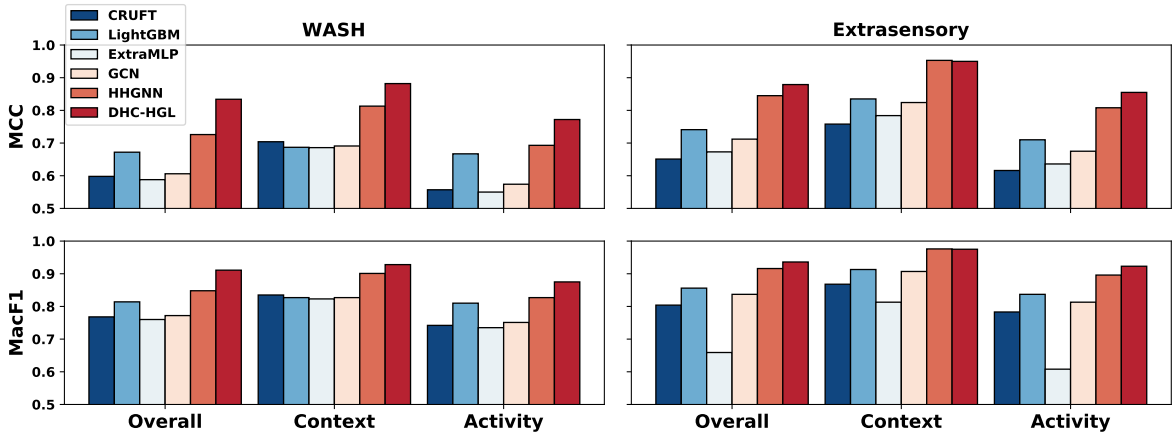


Fig. 8. Average performance of all models across all labels (Overall) and on each category (Context, Activity). Our proposed DHC-HGL is shown in red.

We had learning rate of 8e-4 on the *Extrasensory* dataset and 1e-3 on the *WASH* dataset The optimal values of λ_1 and λ_2 in the loss function were <0.03, 0.01> and <0.3, 0.1> for the *WASH* and *Extrasensory* datasets respectively.

5.4 Evaluation Metrics

Due to the extremely imbalanced nature of the CA-HAR datasets, Matthews Correlation Coefficient (MCC), Eq. 19) and Macro F1 Score (MacF1, Eq. 20) were our main evaluation metrics. MCC is a statistical tool used for model evaluation. Its job is to gauge or measure the difference between the predicted values and actual values. In practice, MCC ranges from -1 to +1, and takes all elements in the confusion matrix (True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN)) into account, making it a reliable statistical rate even for imbalanced datasets [10]. F1-score is the harmonic mean of precision and recall. It is one of the most popular adopted evaluation metrics in classification tasks, and calculating the Macro F1 Score helped us to characterize the overall performance of the proposed model.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{19}$$

$$MacF1 = \frac{1}{C} \sum_{C_i=1}^{C} 2 * \frac{pre_{c_i} * rec_{c_i}}{pre_{c_i} + rec_{c_i}} \tag{20}$$

where C stands for the number of classes. “pre” and “rec” refer to precision and recall. Resultant values of metrics on specific labels are reported as well as averages of their category on the hold-out test set.

5.5 Experiment Results

We show the overall and average performance of all models on each category in Fig. 8. Detailed results for specific labels on two real-world CA-HAR datasets are reported in Tables 4 and 5. Our findings derived from these experimental results are described in this section.

Overall performance: As shown in Fig. 8, DHC-HGL consistently achieves significantly better recognition over baseline models across different datasets, highlighting the effectiveness and generalizability of our proposed

Table 4. Detailed Results on the *WASH* CA-HAR Dataset. For each label, the best results are marked in gray, and the second best results are underlined. DHC-HGL achieved the best performance on most labels.

Category	Label	CRUFT		LightGBM		ExtraMLP		GCN		HHGNN		**DHC-HGL**			
		MCC	MacF1	MCC	MacF1	MCC	MacF1	MCC	MacF1	MCC	MacF1	MCC	Impv. ↑	MacF1	Impv. ↑
Context	In Pocket	0.648	0.801	0.640	0.799	0.629	0.789	0.644	0.803	0.771	0.878	0.849	(13.3%)	0.921	(6.6%)
	In Hand	0.522	0.716	0.524	0.722	0.514	0.713	0.511	0.707	0.718	0.846	0.781	(11.1%)	0.883	(5.8%)
	In Bag	0.756	0.867	0.706	0.837	0.683	0.821	0.701	0.833	0.838	0.914	0.908	(11.3%)	0.953	(5.7%)
	On Table-Face Down	0.803	0.896	0.802	0.896	0.805	0.897	0.819	0.904	0.867	0.931	0.942	(11.1%)	0.971	(5.5%)
	On Table-Face Up	0.793	0.894	0.862	0.880	0.800	0.898	0.781	0.890	0.872	0.936	0.928	(8.1%)	0.964	(3.8%)
	Context Avg	0.704	0.835	0.687	0.827	0.686	0.823	0.691	0.827	0.813	0.901	0.902	(10.9%)	0.950	(5.4%)
Activity	Lying Down	0.820	0.907	0.783	0.887	0.831	0.912	0.848	0.921	0.893	0.945	0.943	(8.3%)	0.971	(4.0%)
	Sitting	0.758	0.876	0.735	0.864	0.741	0.867	0.743	0.686	0.816	0.907	0.898	(12.7%)	0.949	(5.8%)
	Walking	0.510	0.721	0.536	0.741	0.523	0.731	0.519	0.732	0.623	0.794	0.738	(23.9%)	0.860	(10.7%)
	Sleeping	0.915	0.957	0.922	0.961	0.921	0.960	0.935	0.967	0.954	0.977	0.947	(3.1%)	0.987	(1.5%)
	Talking On Phone	0.417	0.645	0.545	0.730	0.436	0.656	0.465	0.675	0.657	0.806	0.760	(12.9%)	0.869	(6.5%)
	Bathroom	0.425	0.660	0.499	0.704	0.427	0.659	0.406	0.641	0.615	0.780	0.718	(19.8%)	0.846	(10.0%)
	Standing	0.457	0.681	0.486	0.703	0.463	0.686	0.489	0.705	0.605	0.777	0.716	(27.4%)	0.845	(12.9%)
	Jogging	0.551	0.737	0.964	0.982	0.520	0.712	0.599	0.765	0.712	0.837	0.717	(-11.2%)	0.841	(-5.9%)
	Running	0.479	0.692	0.946	0.973	0.420	0.647	0.489	0.695	0.604	0.769	0.680	(-22.7%)	0.819	(-12.5%)
	Stairs-Going Down	0.382	0.636	0.488	0.697	0.376	0.626	0.374	0.624	0.529	0.726	0.623	(23.3%)	0.789	(11.1%)
	Stairs-Going Up	0.397	0.645	0.469	0.686	0.388	0.634	0.399	0.640	0.537	0.731	0.638	(27.7%)	0.799	(13.4%)
	Typing	0.636	0.793	0.666	0.814	0.636	0.794	0.672	0.817	0.770	0.876	0.865	(16.2%)	0.930	(8.0%)
	Exercising	0.493	0.699	0.626	0.783	0.463	0.672	0.523	0.715	0.692	0.826	0.769	(14.9%)	0.874	(7.7%)
	Activity Avg	0.557	0.742	0.667	0.810	0.550	0.735	0.574	0.751	0.693	0.827	0.808	(16.7%)	0.897	(8.4%)
Overall Avg		0.598	0.768	0.672	0.814	0.588	0.760	0.606	0.772	0.726	0.848	0.834	(14.9%)	0.911	(7.5%)

model. The success of both HHGNN and DHC-HGL demonstrates that explicitly encoding the user and context in the CA-HAR graph can capture correlations between them and facilitate learning better representations. Based on the same initial heterogeneous hypergraph, the better performance of DHC-HGL can prove that our deep heterogeneity design further benefits CA-HAR tasks. To be specific, DHC-HGL outperform the best baseline, HHGNN, with an overall improvement of 14.9% and 4.1% in MCC and 7.5% and 2.2% in Macro F1 on the WASH and ExtraSensory datasets, respectively.

Model performance on each label category: On the *WASH* dataset, MCC and Macro F1 improved by 10.9% and 5.4% for context (phone placement) and improved by 16.7% and 8.4% for activity recognition. On the *Extrasensory* dataset, MCC and Macro F1 results improved by 5.8% and 3.0% for activity recognition and had performance on par with the best baseline for context category (i.e., the performance difference is less than 0.3%). As most CA-HAR tasks consider activities as the most important category to recognize, the observation of higher performance gains in the activity category is encouraging.

Detailed results for each specific label on the *WASH* CA-HAR dataset: Detailed results for all models on the *WASH* dataset are listed in Table 4. Among the baseline models, first, it can be observed that HHGNN outperformed all other models, which strongly demonstrates the advantage of modeling CA-HAR tasks as a heterogeneous hypergraph. Additionally, DHC-HGL outperforms all baselines on most labels by a large margin, except for the *Jogging* and *Running* labels, which have very small positive support instances (are scarce) (with only 3804 and 2422 positive instances, separately) compared to other labels with larger positive instance support. For instance, *Sleeping* has 2,935,264 positive instances. On the other hand, *Jogging* and *Running* produce similar sensor signals, making them challenging to discriminate [6]. LightGBM overcomes this confusion by training a separate model for each label. Other baselines that train a single model for all labels achieved are less able to discriminate *Jogging* and *Running*. Another possible explanation is that as labels were self-reported, many

Table 5. Detailed Results on the *Extrasensory* CA-HAR Dataset. For each label, the best results are in gray, and the second best results are underlined. DHC-HGL achieved the best performance on all activities and performance mostly on-par with or better than the best baselines.

Category	Label	CRUFT		LightGBM		ExtraMLP		GCN		HHGNN		**DHC-HGL**			
		MCC	MacF1	MCC	MacF1	MCC	MacF1	MCC	MacF1	MCC	MacF1	MCC	Impv. ↑	MacF1	Impv. ↑
Context	In Pocket	0.782	0.884	0.839	0.916	0.801	0.825	0.809	0.898	0.950	0.974	0.952	(0.2%)	0.976	(0.1%)
	In Hand	0.638	0.797	0.736	0.855	0.682	0.686	0.747	0.864	0.931	0.965	0.925	(-0.7%)	0.962	(-0.4%)
	In Bag	0.775	0.878	0.895	0.945	0.788	0.786	0.862	0.928	0.964	0.982	0.954	(-1.1%)	0.976	(-0.6%)
	On Table	0.835	0.915	0.872	0.935	0.864	0.956	0.876	0.937	0.966	0.983	0.969	(0.3%)	0.984	(0.1%)
	Context Avg	0.758	0.868	0.835	0.913	0.784	0.813	0.824	0.907	0.953	0.976	0.950	(-0.3%)	0.975	(-0.2%)
Activity	Lying Down	0.918	0.959	0.911	0.955	0.936	0.958	0.941	0.971	0.962	0.981	0.977	(1.6%)	0.989	(0.8%)
	Sitting	0.772	0.885	0.755	0.876	0.797	0.889	0.781	0.888	0.870	0.935	0.907	(4.3%)	0.953	(2.0%)
	Walking	0.535	0.730	0.560	0.746	0.554	0.539	0.580	0.758	0.718	0.846	0.772	(7.5%)	0.878	(3.8%)
	Sleeping	0.934	0.967	0.934	0.967	0.950	0.965	0.952	0.976	0.979	0.989	0.985	(0.6%)	0.993	(0.3%)
	Talking	0.635	0.794	0.697	0.833	0.681	0.703	0.740	0.859	0.858	0.927	0.895	(4.3%)	0.946	(2.1%)
	Bath-Shower	0.495	0.703	0.729	0.848	0.496	0.404	0.648	0.798	0.780	0.880	0.827	(5.9%)	0.908	(3.1%)
	Toilet	0.416	0.661	0.524	0.716	0.429	0.326	0.446	0.666	0.670	0.813	0.743	(10.8%)	0.859	(5.7%)
	Standing	0.579	0.762	0.573	0.760	0.621	0.648	0.639	0.802	0.766	0.875	0.828	(8.1%)	0.911	(4.0%)
	Running	0.609	0.782	0.656	0.802	0.616	0.555	0.724	0.845	0.866	0.930	0.883	(1.9%)	0.938	(0.9%)
	Stairs-Going Down	0.431	0.674	0.791	0.886	0.441	0.334	0.462	0.676	0.695	0.831	0.807	(2.1%)	0.898	(1.3%)
	Stairs-Going Up	0.429	0.674	0.654	0.801	0.446	0.341	0.513	0.710	0.706	0.837	0.770	(9.1%)	0.876	(4.6%)
	Exercising	0.642	0.800	0.735	0.852	0.665	0.636	0.673	0.812	0.803	0.909	0.868	(4.5%)	0.930	(2.3%)
	Activity Avg	0.616	0.783	0.710	0.837	0.636	0.608	0.675	0.813	0.808	0.896	0.855	(5.8%)	0.923	(3.0%)
Overall Avg		0.651	0.804	0.741	0.856	0.673	0.659	0.712	0.837	0.845	0.916	0.879	(4.1%)	0.936	(2.2%)

participants may have confused *Jogging* and *Running*. It is instructive to note that when compared to the best overall baseline HHGNN, DHC-HGL managed to improve on the *Jogging* and *Running* labels.

Detailed results for each specific label on the *Extrasensory* CA-HAR dataset: Detailed experimental results of all models for all labels are reported in Table 5. It can be observed that slightly different from *WASH* dataset, DHC-HGL outperforms baselines on all activity labels, even including labels where HHGNN could not outperform LightGBM (e.g., Stairs-Going Down). Meanwhile, DHC-HGL did not achieve as significant improvements on context labels on *Extrasensory* as it did on the *WASH* dataset. It is still worth noting that results for DHC-HGL on context labels are on-par with the best baselines, with a difference of no more than 1.1%. The lack of improvement on context labels might be due to the limited amount of $\{u, c\}$ edges in *WASH* as reported in Fig. 7, where only 1.2% of total hyperedges are $\{u, c\}$ in *Extrasensory*, compared to 9.6% in *WASH*.

Nevertheless, most of the results of DHC-HGL on the *Extrasensory* dataset are consistent with results achieved on the *WASH* dataset.

6 ANALYSIS

In this section, we attempt to dive deeper into our results and derive additional insights on the performance of our proposed DHC-HGL in a Q&A fashion.

RQ1: How much does each proposed novel component contribute? In order to deal with the heterogeneous nature of CA-HAR data effectively, our proposed DHC-HGL framework integrates several key components. A key question that arises is the utility of the various components of DHC-HGL. To answer this question, an ablation study was conducted on both CA-HAR datasets in order to understand the contributions made by the contrastive loss and edge heterogeneity aspects of the DHC-HGL design to its improved performance for the CA-HAR task. Results are shown in Fig. 9. The models explored are described below:

- **DHC-HGL**: Proposed approach using optimal hyperparameter values.

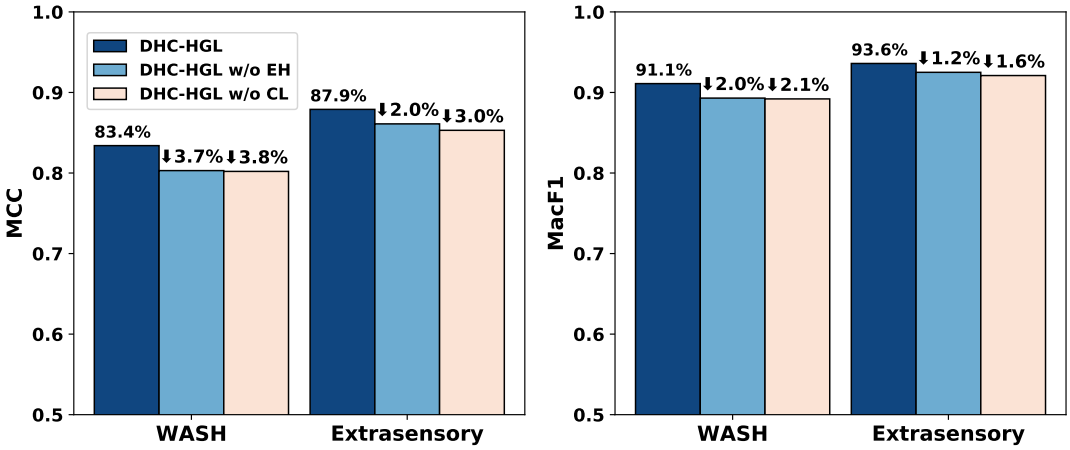
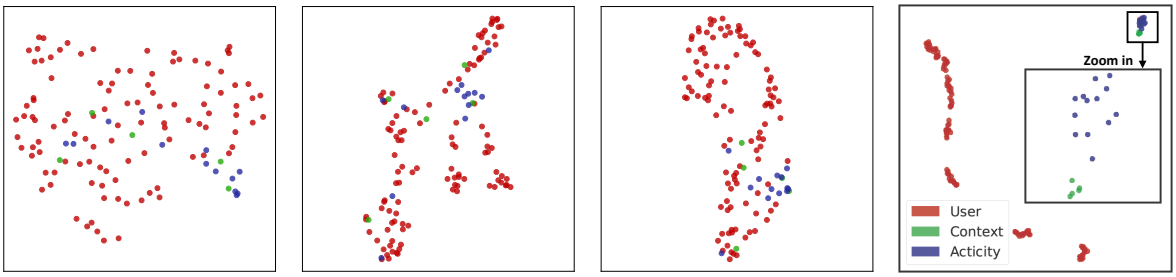


Fig. 9. Ablation study compares DHC-HGL with its variants. EH: designs addressing edge-heterogeneity. CL: designs introducing contrastive loss for node-heterogeneity.



(a) Original nodes (b) HHGNN learned nodes (c) DHC-HGL w/o CL (d) DHC-HGL w/ CL

Fig. 10. UMAP visualization of 10a) the original *WASH* initialized node, 10b) HHGNN learned node embedding, 10c) node embedding learned by DHC-HGL without contrastive loss, and 10d) node embedding learned by DHC-HGL with contrastive loss (full model). The red, green, and blue dots in the graph represent user, context, and activity nodes, respectively.

- **DHC-HGL w/o edge heterogeneity(EH)**: The hyperedge splitting procedure was removed, which yielded an integrated graph that is passed through a common, single hyperConv layer.
- **DHC-HGL w/o contrastive loss (CL)**: The contrastive loss was removed from DHC-HGL. In this way, the graph learned is only used for classification. It is included in order to evaluate the contribution of the contrastive loss to both model performance and interpretability aspects.

Unsurprisingly, performance in terms of MCC/MacF1 decreased by -3.7%/-2.0% and -2.0%/-1.2% when the contrastive loss was removed; and decreased by -3.8%/-2.1% and -3.0%/-1.6% when the same hypergraph convolution layer was used without distinguishing the type of hyperedges that occur in the *WASH* and *Extrasensory* datasets, respectively. In summary, 1) both the innovative contrastive loss and passing different types of hypergraphs through different hypergraph convolution layers had non-trivial contributions to the overall performance improvement of DHC-HGL, and 2) using different hypergraph convolutional layers to address edge-heterogeneity

appeared to have a larger influence than the designed contrastive loss. This is understandable as no other component addresses edge-heterogeneity. In contrast, the contrastive loss function addressed node-heterogeneity, which is partly resolved by separate linear projections as proposed in HHGNN and leveraged in DHC-HGL.

RQ2: Does DHC-HGL accurately capture the relationships between various nodes? We previously suggested that DHC-HGL captured predictive relationships between various entities within context labels. In order to validate this claim, here, in Fig. 10, we present UMAP [30] visualizations of node representations generated from the *WASH* dataset when various graph-based models were applied. Visualizations on the *Extrasensory* dataset followed a similar pattern.

UMAP projects representations from high-dimensional to two-dimensional space while preserving both local properties (within each cluster) and global properties (among clusters). Essentially, nodes that are close in high dimensions remain close after UMAP projection. The visualization on the left (Fig 10a) shows that the initialized points are close to being uniformly distributed. HHGNN tried to deal with node heterogeneity by assigning specific linear functions to different types of nodes, but the learned node embedding (Fig. 10b) is not able to distinguish different types of nodes well, which is consistent with our previous claim that separate linear functions are not adequate to handle node heterogeneity. By applying DHC-HGL, nodes form clusters with clear boundaries (Fig. 10d). Additionally, user nodes formed different groups, suggesting the possibility of building customized models. A comparison of Figs 10c and 10d further demonstrates the contribution of contrastive loss not only quantitatively but also its improvement of interpretability.

RQ3: How does model structure influence recognition performance? We explore the robustness of DHC-HGL by investigating the impact of two key hyperparameters: 1) node embedding size and 2) the number of graph learning layers. A performance visualization is shown in Fig. 11 with four subplots. The best-performing baseline HHGNN with a node embedding dimension of 1536[3] is included as a yellow line in all subplots.

Impact of node embedding dimension in Fig. 11a and Fig. 11b:

(1) DHC-HGL achieved its best performance on the *WASH* dataset with 6144 dimensions and on the *Extrasensory* dataset with 3072 dimensions. Overall, a performance gain can be observed as the embedding dimension increases, where a higher dimension enables the model to capture more information from the graph. However, there is also a saturation point at which the performance stops increasing after the embedding dimension is larger than the optimal value.
(2) DHC-HGL consistently outperformed HHGNN in most cases in the regions of the graphs around the optimal node embedding dimensions, which indicates its robustness. A larger gain is achieved on the *WASH* dataset compared to the *Extrasensory* dataset.

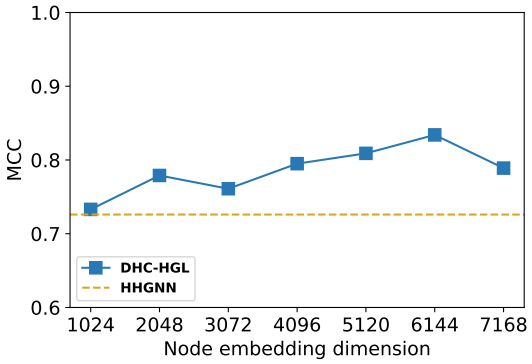
Impact of number of graph learning layer in Fig. 11c and Fig. 11d:

(1) DHC-HGL achieved its best performance with two graph learning layers, on both *WASH* and *Extrasensory* datasets. The performance was reduced when more than two graph learning layers were used. One possible reason is that deeper graph learning layers make it more difficult for the model to generalize and are also prone to overfitting. Another possible reason is that deeper graph learning layers might cause over-smoothing [22, 50], where learned node embeddings become similar and indistinguishable.
(2) DHC-HGL yields better performance than HHGNN under most cases, indicating DHC-HGL's robustness w.r.t the number of graph learning layers.

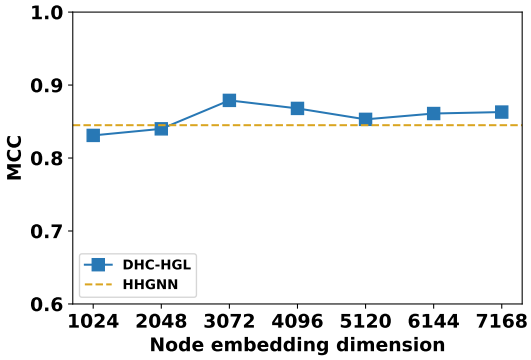
7 LIMITATIONS

Our work has a few limitations we now mention. First, our definition of context as phone placement in the CA-HAR problem statement followed and was inspired by prior work. Interested researchers may extend our

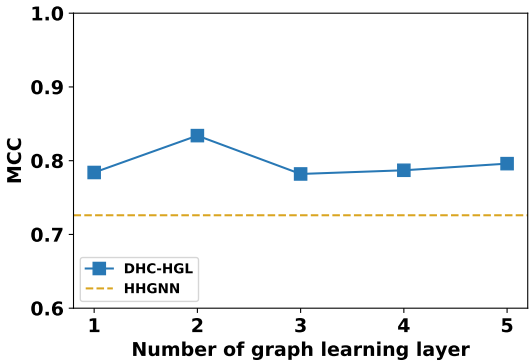
[3]These are the best hyperparameters found through grid search.



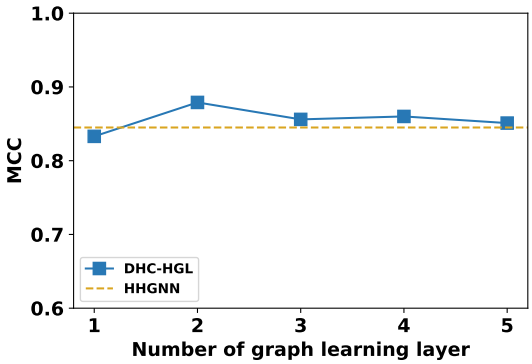
(a) The overall performance (MCC) of DHC-HGL with different node embedding sizes on the *WASH* dataset.



(b) The overall performance (MCC) of DHC-HGL with different node embedding size on the*Extrasensory* dataset.



(c) The overall performance (MCC) of DHC-HGL with different numbers of graph learning layers on the *WASH* dataset.



(d) The overall performance (MCC) of DHC-HGL with different numbers of graph learning layers on the *Extrasensory* dataset.

Fig. 11. Hyperparameter evaluation.

work to other context definitions subject to availability of data on other relevant contextual information such as device type. Secondly, while we formed the heterogeneous hypergraph on the training set in a user-aware mode, the inference on the test set was user-agnostic / user-implicit (i.e., we did not use user identity as an input while inferencing, nor did we infer user nodes given input sensor signal). We leave user-explicit inference and inferring user identity as future work as it requires minimum adaptation effort. Admittedly, we evaluated the effectiveness of our framework mainly on the hypergraph convolutional neural network backbone. We believe that our design is backbone-agnostic and that our novel contributions can serve as plug-ins for various hypergraph learning backbones. Lastly, casting the CA-HAR task into a graph learning task is only one of the possible approaches for resolving the problem. We may consider further combining other modalities with the message-passing design to improve the model performance.

8 CONCLUSION AND FUTURE WORK

Context-aware Human Activity Recognition is an emerging but challenging problem for academia and industry domains. Prior work mainly researched non-graph-based, feature-dependent ordinary-graph-based, or hypergraph-based methods and models that implicitly addressed node-heterogeneity. In this work, we introduced a novel feature-independent hypergraph-based neural networks approach DHC-HGL, which addresses both edge-heterogeneity and node-heterogeneity in a CA-HAR data-transformed hypergraph. More specifically, 1) for handling edge-heterogeneity, we leverage different hypergraph convolutional layers for various edge types in contrast to unified hypergraph convolutional layers with shared parameters for all hyperedges. 2) To enforce explicit node-heterogeneity, we designed a contrastive loss applied to the node embedding latent space. Such a contrastive loss enlarges the distance between different types of nodes while pulling nodes of the same type closer. In an extensive experimental study, DHC-HGL yielded a superior performance boost on two large datasets and provided better UMAP visualizations of label representation distributions, which enhances model interpretability.

In future work, we plan to evaluate our design on more neural network backbones other than GCN-based models as our model is intuitively backbone-agnostic. We adopted the straightforward summation function as our aggregation method of hypergraph neural networks. In the future, we will explore other possibilities, such as mean pooling and attention-based pooling. Following prior works, our definition of context was restricted to the device placement in this work due to the limited availability of other contextual factors (e.g., device type) in experimental CA-HAR datasets. However, in the future, it would be interesting to explore whether exploiting other context labels/features can help to further improve the model performance. In any case, our novel designed components can easily be adapted to incorporate other contextual factors with minimum effort. We would also like to point out the possibility of extending this work to handle both user-explicit and user-implicit CA-HAR tasks by introducing user identity as part of the input. This will facilitate user-explicit settings, introducing unknown user nodes for user-implicit settings, or inferring not only contexts and activities but also user identities. Although this work focused on CA-HAR tasks, DHC-HGL also has the potential applicability to other general time series problems and link prediction problems, including diagnosing cardiovascular disease (CVD) using Electrocardiogram (ECG) signals and document recommendation.

ACKNOWLEDGMENT

The research was sponsored by DARPA grant HR00111780032-WASH-FP-031. We thank the WPI WASH team for gathering the WASH dataset analyzed in this paper.

REFERENCES

[1] 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Proc. NIPS* 30 (2017).

[2] Abdulaziz Alajaji, Walter Gerych, Kavin Chandrasekaran, Luke Buquicchio, Emmanuel Agu, and Elke Rundensteiner. 2020. Deepcontext: Parameterized compatibility-based attention cnn for human context recognition. In *2020 IEEE 14th International Conference on Semantic Computing (ICSC)*. IEEE, 53–60.

[3] Lei Bai, Lina Yao, Xianzhi Wang, Salil S Kanhere, Bin Guo, and Zhiwen Yu. 2020. Adversarial multi-view networks for activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (2020), 1–22.

[4] Song Bai, Feihu Zhang, and Philip HS Torr. 2021. Hypergraph convolution and hypergraph attention. *Pattern Recognition* 110 (2021), 107637.

[5] Valentina Bianchi, Marco Bassoli, Gianfranco Lombardo, Paolo Fornacciari, Monica Mordonini, and Ilaria De Munari. 2019. IoT wearable sensor and deep learning: An integrated approach for personalized human activity recognition in a smart home environment. *IEEE Internet of Things Journal* 6, 5 (2019), 8553–8562.

[6] Bhaskar Chakraborty, Ognjen Rudovic, and Jordi Gonzalez. 2008. View-invariant human-body detection with extension to human action recognition using component-wise HMM of body parts. In *Int'l Conf. Automatic Face & Gesture Rec.* IEEE, 1–6.

[7] Youngjae Chang, Akhil Mathur, Anton Isopoussu, Junehwa Song, and Fahim Kawsar. 2020. A systematic study of unsupervised domain adaptation for robust human-activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–30.

[8] Kuang-Hsuan Chen, Yu-Wei Hsu, Jing-Jung Yang, and Fu-Shan Jaw. 2018. Evaluating the specifications of built-in accelerometers in smartphones on fall detection performance. *Inst. Sci. & Tech.* 46, 2 (2018), 194–206.

[9] Ling Chen, Yi Zhang, and Liangying Peng. 2020. METIER: A deep multi-task learning based activity and user recognition model using wearable sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–18.

[10] Davide Chicco and Giuseppe Jurman. 2020. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC genomics* 21 (2020), 1–13.

[11] Ada Dogrucu, Alex Perucic, Anabella Isaro, Damon Ball, Ermal Toto, Elke A Rundensteiner, Emmanuel Agu, Rachel Davis-Martin, and Edwin Boudreaux. 2020. Moodable: On feasibility of instantaneous depreswension assessment using machine learning on voice samples with retrospectively harvested smartphone and social media data. *Smart Health* 17 (2020), 100118.

[12] Haoyi Fan, Fengbin Zhang, Yuxuan Wei, Zuoyong Li, Changqing Zou, Yue Gao, and Qionghai Dai. 2021. Heterogeneous hypergraph variational autoencoder for link prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 8 (2021), 4125–4138.

[13] Xile Gao, Haiyong Luo, Qu Wang, Fang Zhao, Langlang Ye, and Yuexia Zhang. 2019. A human activity recognition algorithm based on stacking denoising autoencoder and lightGBM. *Sensors* 19, 4 (2019), 947.

[14] Wen Ge and Emmanuel Agu. 2020. CRUFT: Context recog. under uncertainty using fusion and temporal learning. In *Proc. ICMLA*. IEEE, 747–52.

[15] Wen Ge and Emmanuel O Agu. 2022. QCRUFT: Quaternion Context Recog. under Uncertainty using Fusion & Temporal Learning. In *Proc. ICSC*. IEEE, 41–50.

[16] Wen Ge, Guanyi Mou, Emmanuel O Agu, and Kyumin Lee. 2023. Heterogeneous Hyper-Graph Neural Networks for Context-aware Human Activity Recognition. In *PerCom*.

[17] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. 2. IEEE, 1735–1742.

[18] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Proc. NIPS* 30 (2017).

[19] Hua Kang, Qianyi Huang, and Qian Zhang. 2022. Augmented Adversarial Learning for Human Activity Recognition with Partial Sensor Sets. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 3 (2022), 1–30.

[20] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems* 33 (2020), 18661–18673.

[21] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[22] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[23] Yinfeng Li, Chen Gao, Quanming Yao, Tong Li, Depeng Jin, and Yong Li. 2022. DisenHCN: Disentangled Hypergraph Convolutional Networks for Spatiotemporal Activity Prediction. *arXiv preprint arXiv:2208.06794* (2022).

[24] Janne Lindqvist and Jason Hong. 2011. Undistracted driving: A mobile phone that doesn't distract. In *Proc. Hotmobile*. 70–75.

[25] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265* (2019).

[26] Shinan Liu, Tarun Mangla, Ted Shaowang, Jinjin Zhao, John Paparrizos, Sanjay Krishnan, and Nick Feamster. 2023. AMIR: Active Multimodal Interaction Recognition from Video and Network Traffic in Connected Environments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 1 (2023), 1–26.

[27] Weifeng Liu, Sichao Fu, Yicong Zhou, Zheng-Jun Zha, and Liqiang Nie. 2021. Human activity recognition by manifold regularization based dynamic graph convolutional networks. *Neurocomputing* 444 (2021), 217–225.

[28] Jing Ma, Mengting Wan, Longqi Yang, Jundong Li, Brent Hecht, and Jaime Teevan. 2022. Learning causal effects on hypergraphs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1202–1212.

[29] Henry Martin, Dominik Bucher, Esra Suel, Pengxiang Zhao, Fernando Perez-Cruz, and Martin Raubal. 2018. Graph convolutional neural networks for human activity purpose imputation. In *Spatiotemporal workshop co-located with NIPS*.

[30] Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

[31] Abduallah Mohamed, Fernando Lejarza, Stephanie Cahail, Christian Claudel, and Edison Thomaz. 2022. HAR-GCNN: Deep Graph CNNs for Human Activity Recog. From Highly Unlabeled Mobile Sensor Data. In *Proc. PerCom*. IEEE, 335–40.

[32] Kazushige Ouchi and Miwako Doi. 2013. Smartphone-based monitoring system for activities of daily living for elderly people and their relatives etc.. In *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*. 103–106.

[33] Leif E Peterson. 2009. K-nearest neighbor. *Scholarpedia* 4, 2 (2009), 1883.

[34] Sreenivasan Ramasamy Ramamurthy, Soumyajit Chatterjee, Elizabeth Galik, Aryya Gangopadhyay, Nirmalya Roy, Bivas Mitra, and Sandip Chakraborty. 2022. CogAx: Early Assessment of Cognitive and Functional Impairment from Accelerometry. In *PerCom*. IEEE, 66–76.

[35] Tifenn Rault, Abdelmadjid Bouabdallah, Yacine Challal, and Frédéric Marin. 2017. A survey of energy-efficient context recognition systems using wearable sensors for healthcare applications. *Perv. & Mob. Comp.* 37 (2017), 23–44.

[36] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S Yu Philip. 2018. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering* 31, 2 (2018), 357–370.

[37] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. 2020. Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. *IEEE Transactions on Image Processing* 29 (2020), 9532–9545.

[38] Guillaume St-Onge, Iacopo Iacopini, Vito Latora, Alain Barrat, Giovanni Petri, Antoine Allard, and Laurent Hébert-Dufresne. 2022. Influential groups for seeding and sustaining nonlinear contagion in heterogeneous hypergraphs. *Communications Physics* 5, 1 (2022), 25.

[39] Xiangguo Sun, Hongzhi Yin, Bo Liu, Hongxu Chen, Jiuxin Cao, Yingxia Shao, and Nguyen Quoc Viet Hung. 2021. Heterogeneous hypergraph embedding for graph classification. In *Proceedings of the 14th ACM international conference on web search and data mining*. 725–733.

[40] Yonatan Vaizman, Katherine Ellis, and Gert Lanckriet. 2017. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE Pervasive Computing* 16, 4 (2017), 62–74.

[41] Yonatan Vaizman, Nadir Weibel, and Gert Lanckriet. 2018. Context recognition in-the-wild: Unified model for multi-modal sensors and multi-label classification. *ACM IMWUT* 1, 4 (2018), 1–22.

[42] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[43] Jianchao Wu, Limin Wang, Li Wang, Jie Guo, and Gangshan Wu. 2019. Learning actor relation graphs for group activity recognition. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*. 9964–9974.

[44] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudre-Mauroux. 2019. Revisiting user mobility and social relationships in lbsns: a hypergraph embedding approach. In *The WWW Conf.* 2147–2157.

[45] Dingqi Yang, Bingqing Qu, Jie Yang, and Philippe Cudré-Mauroux. 2020. Lbsn2vec++: Heterogeneous hypergraph embedding for location-based social networks. *Trans. Knowledge and Data Engineering* (2020).

[46] Shuochao Yao, Yiran Zhao, Huajie Shao, Aston Zhang, Chao Zhang, Shen Li, and Tarek Abdelzaher. 2018. Rdeepsense: Reliable deep mobile computing models with uncertainty estimations. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 1–26.

[47] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proc. ACM SIGKDD*. 793–803.

[48] Ye Zhang, Longguang Wang, Huiling Chen, Aosheng Tian, Shilin Zhou, and Yulan Guo. 2022. IF-ConvTransformer: A framework for human activity recognition using IMU fusion and ConvTransformer. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (2022), 1–26.

[49] Wenbo Zheng, Lan Yan, Chao Gou, and Fei-Yue Wang. 2022. Meta-learning meets the Internet of Things: Graph prototypical models for sensor-based human activity recognition. *Information Fusion* 80 (2022), 1–22.

[50] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI open* 1 (2020), 57–81.

[51] Yu Zhu, Ziyu Guan, Shulong Tan, Haifeng Liu, Deng Cai, and Xiaofei He. 2016. Heterogeneous hypergraph embedding for document recommendation. *Neurocomputing* 216 (2016), 150–162.

[52] Qin Zou, Yanling Wang, Qian Wang, Yi Zhao, and Qingquan Li. 2020. Deep learning-based gait recognition using smartphones in the wild. *IEEE Transactions on Information Forensics and Security* 15 (2020), 3197–3212.