This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945

Negative Selection by Clustering for Contrastive Learning in Human Activity Recognition

Jinqiang Wang, Tao Zhu, Liming Chen, *Senior Member, IEEE,* Huansheng Ning, *Senior Member, IEEE,* and Yaping Wan

***Abstract*—Contrastive learning is an emerging and important self-supervised learning paradigm that has been successfully applied to sensor-based human activity recognition (HAR) because it can achieve competitive performance relative to supervised learning. Contrastive learning methods generally involve instance discrimination, which means that the instances are regarded as negatives of each other, and thus their representations are pulled away from each other during the training process. However, instance discrimination could cause overclustering, meaning that the representations of instances from the same class could be overly separated. To alleviate this overclustering phenomenon, we propose a new contrastive learning framework to select negatives by clustering in HAR, which is named ClusterCLHAR. First, ClusterCLHAR clusters the instance representations, and for each instance, only those from different clusters are regarded as negatives. Second, a new contrastive loss function is proposed to mask the same-cluster instances from the negative pairs. We evaluate ClusterCLHAR on three popular benchmark datasets, USC-HAD, MotionSense, and UCI-HAR, using the mean F1-score as an evaluation metric for downstream tasks. The experimental results show that ClusterCLHAR outperforms all the state-of-the-art methods applied to HAR in self-supervised learning and semi-supervised learning.**

***Index Terms*—Human Activity Recognition, Sensor Data, Contrastive Learning, Negative Selection, Clustering.**

I. Introduction

HUMAN activity recognition (HAR) based on the Internet of Things and wearable sensing (accelerometers and gyroscopes) had a crucial role in emerging user-centered smart applications, such as smart homes [1], [2] fall detection [3], [4] and healthcare rehabilitation [5], [6]. Recent applications of deep learning techniques [7], [8], [9], [10] have significantly improved activity recognition accuracy. However, deep learning methods usually require a large number of labeled data sets to train an activity recognition model. Manual labeling of sensor data is time-consuming and tedious, especially in the healthcare field, where the collection of labeled data is more challenging. Moreover, the labels are affected by various noise sources, such as sensor noise, segmentation problems, and differences in the activities of different people, rendering the annotation process error-prone [11]. Therefore, insufficient data annotation becomes a major challenge for HAR.

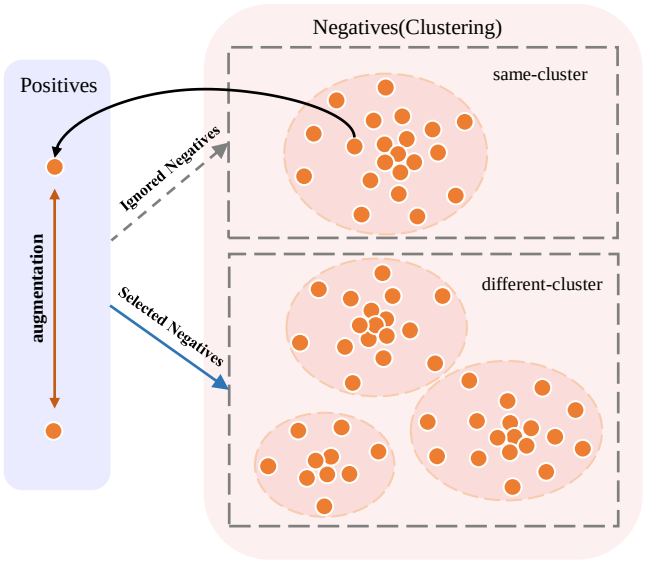


Fig. 1. Negative Selection by Clustering. An elliptical box represents a cluster. Previous negative example selection methods, such as the instance discrimination, consider all samples inside the right rounded rectangle as negative examples except for itself and the corresponding augmented samples. However, our proposed negative example selection method will not consider samples from the same cluster as negative examples.

To alleviate the issue of insufficient data annotation, contrastive learning, which is a paradigm of self-supervised learning, has been developed and indeed achieved excellent performance in computer vision [12]. The pretraining process for contrastive learning is to generate pseudolabels using data augmentation on a large amount of unlabeled data, enabling the model to distinguish which augmented instances are positive pairs and which augmented instances are negative pairs [13]. After pretraining, the model is fine-tuned in downstream tasks using a small amount of labeled data and can achieve performance comparable to supervised learning [14], [15].

There are many types of pretraining tasks for contrastive learning, such as MoCo [16], [17] and SimCLR [18], [19], which use instance discrimination [20] as the task. NNCLR [21], MSF [22], TTL [23] and HardCL [24] were proposed to redefine positive and negative pairs based on the instance

This work is partly supported by the National Natural Science Foundation of China (62006110, 62071213), the Natural Science Foundation of Hunan Province (2021JJ30574), and the Research Foundation of Education Bureau of Hunan Province (21C0311, 21B0424). *(Corresponding author: Tao Zhu.)*

Jinqiang Wang, Tao Zhu and Yaping Wan are with the School of Computer Science, University of South China, 421001 China. e-mail: tzhu@usc.edu.cn.

Liming Chen is with the Ulster University, Northern Ireland, UK. e-mail: l.chen@ulster.ac.uk.

Huansheng Ning is with the School of Computer & Communication Engineering, University of Science and Technology Beijing, 100083 China. e-mail: ninghuansheng@ustb.edu.cn.

Codes available at https://github.com/diheal/negative.

discrimination task. Generally, the pretraining task of the aforementioned studies is to enable the model to generate close representations for positive pairs and distant representations for negative pairs in the latent space. However, such a pretraining task tends to force the model to overcluster instances of the same class into different clusters.

Pretraining tasks of contrastive learning applied to HAR also involve instance discrimination [20], [25] such as SimCLRHAR [26] and CSSHAR [27] built on an improved SimCLR [18], and MoCoHAR [28] built on an improved MoCo [17]. These works generally optimize augmentation methods or backbone networks rather than pretraining tasks. Contrastive learning models that use instance discrimination as a pretraining task tend to fall into overclustering [23] during training. The normalized temperature-scaled cross-entropy loss (NT-Xent) is the contrastive loss function of SimCLR, which follows the instance discrimination task and easily causes overclustering. The pretraining effect of NT-Xent is shown in Fig. 2(a). Each activity instance is separated under the latent space, even if they belong to the same class. This phenomenon occurs because the instance discrimination task requires pulling instance representations away from each other. As shown in Fig. 2(b), it is more ideal for HAR that the representations of activities of the same class are close to each other in the latent space. Such representations are more conducive to downstream classification tasks. Therefore, it is challenging to avoid overclustering during contrastive learning training. The challenge is how to avoid regarding the same-class samples as negative examples during the instance discrimination task.

To address this challenge, we extend SimCLR (Fig. 3(a) [18]) to propose a new contrastive learning framework named ClusterCLHAR to select negatives by clustering in HAR. ClusterCLHAR is shown in Fig. 3(b). First, ClusterCLHAR clusters the instance representations, and for each instance, only those from different clusters are regarded as negatives. Second, a new contrastive loss function Cluster-NT-Xent, which is built on an improved NT-Xent, is proposed to mask instances of the same cluster out of the negative pairs. Therefore, this function could reduce the probability of pulling away instance representations of the same class from each other. A graphical representation of the improvement of the contrastive loss function is shown in Fig. 1.

ClusterCLHAR is evaluated in the following experimental setup. TPN [29] is employed as the backbone network of ClusterCLHAR, which is fast in inference and outperforms DeepConvLSTM [30] in supervised learning. Three benchmark datasets, USC-HAD [31], MotionSense [32], and UCI-HAR [33], were employed for the experiment. First, we compare ClusterCLHAR with previous work under self-supervised learning (downstream tasks using all the data labels). The initial analysis shows that ClusterCLHAR outperforms all state-of-the-art self-supervised learning work on three datasets. Second, we evaluate ClusterCLHAR under semi-supervised learning (downstream tasks using a small fraction of the data labels). This part of the experiment focuses on comparing the performance of supervised learning and previous contrastive learning methods. The pretrained model was fine-tuned using 1% or 10% of the data labels in the downstream task, and the remaining data were utilized as the test set. The semi-supervised experimental results show that ClusterCLHAR outperforms all state-of-the-art contrastive learning methods on three datasets. Last, we visualize the output of ClusterCLHAR and SimCLR using the t-SEN [34], [35] dimensionality reduction method. The experimental results indicated that ClusterCLHAR outputs more similar representations for the same-class samples relative to SimCLR. This demonstrates that ClusterCLHAR can effectively alleviate overclustering and explains why ClusterCLHAR works.

The contributions of this paper are presented as follows: 1. A new contrastive learning framework, ClusterCLHAR, is proposed for HAR, which applies clustering methods to exclude same-cluster samples from negative pairs to effectively alleviate the overclustering phenomenon. 2. We propose a new contrastive loss function, Cluster-NT-Xent, to verify the effect of clustering confidence on CluserCLHAR. 3. On three popular benchmark datasets, we obtained SOTA results in both self-supervised learning settings and semi-supervised learning settings.

The remainder of this paper is structured as follows. Section II reviews the work of contrastive learning on negative example selection and HAR. Section III describes the ClusterCLHAR framework in detail. Section IV designs self-supervised learning and semi-supervised learning experimental protocols to evaluate the performance of our proposed framework. Section V presents and discusses the experimental results of self-supervised learning and semi-supervised learning. Section VI further discusses the details of our proposed framework by ablation studies. Section VII summarizes this paper and proposes future work based on the observed shortcomings.

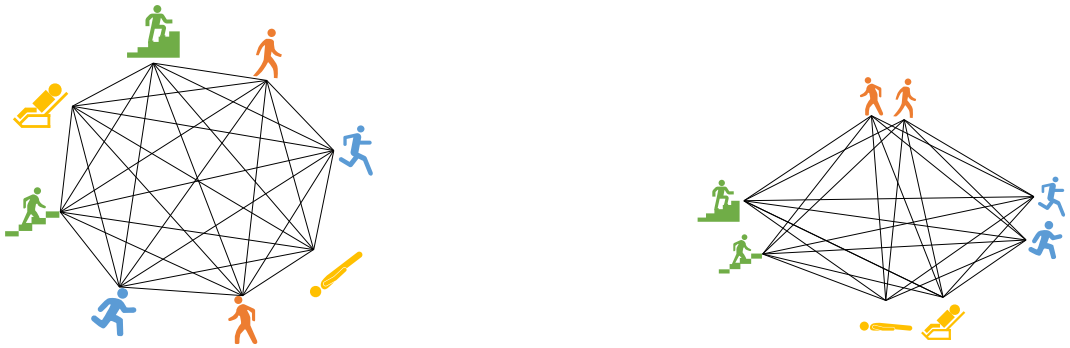
II. Related Works

A. Selection of negative examples in contrastive learning

Contrastive learning is a paradigm of self-supervised learning. In contrastive learning, data augmentation is performed to generate pseudolabels that enable the model to distinguish between positive pairs and negative pairs in augmented samples [13]. Contrastive learning consists of three steps [14]. The first step is augmentation, which determines the quality of pseudolabel generation and has a significant impact on the final performance of the model. The second step is the encoder, which encodes the augmented samples and affects the quality of the generated representation in the latent space. The third step is the loss function (pretraining task), which defines the positive and negative samples that determine the direction the model will learn.

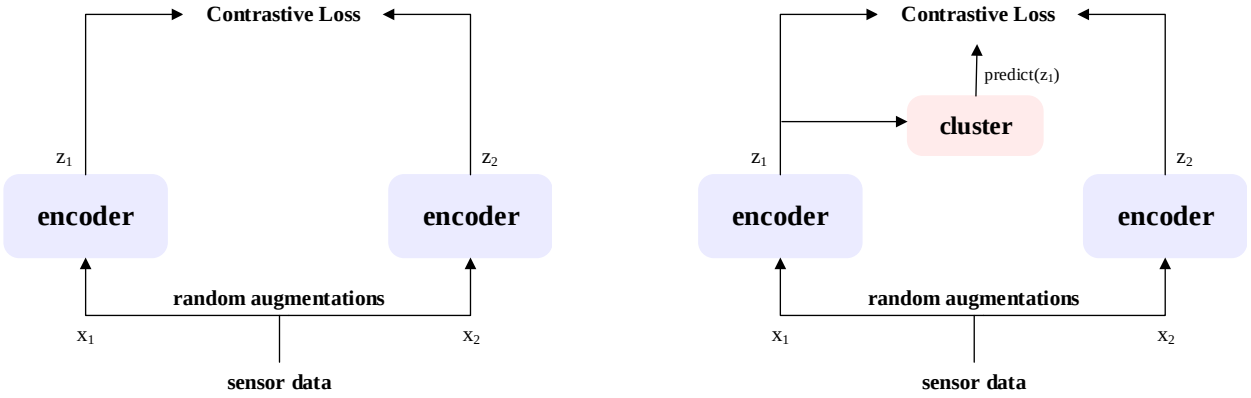
The definition of negative examples has a crucial role in the final performance of contrastive learning, and many studies on contrastive learning have been conducted on this issue. MoCo [16] uses queues to expand negative examples and achieves excellent performance with a small batch size. SimCLR [18] uses a larger batch size to expand the number of negative examples, which improves the performance of contrastive learning in a simple and efficient way. NNCLR [21] compares with SimCLR to select the most similar sample representation from a queue by the nearest neighbor method instead of the

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945



(a) over-clustering (b) "ideal" result

Fig. 2. Activity Representations for Classification in Latent Space. The same color indicates the same activity class. Shorter distances indicate more similar representations.



(a) SimCLR (b) ClusterCLHAR

Fig. 3. Outline of Contrastive Learning Frameworks. Compared to SimCLR, ClusterCLHAR clusters the representations of a branch and assigns cluster labels. Sample representations with the same cluster labels will no longer be considered as negative pairs when calculating the contrastive loss.

original representation to calculate the contrastive loss, thus improving the model performance by increasing the training complexity. MSF [22] calculates the contrastive loss by selecting the mean of the K most similar sample representations from a queue instead of the original sample representation. TTL [23] and HardCL [24] define near positive examples as negative examples for the purpose of separating similar sample representations in the latent space.

The above work tends to cause overclustering, assigning the same class of samples to different clusters.

It is expected that the same-class sample representations remain close in the latent space while the different classes are pushed apart. For this reason, when designing the contrastive loss function, the possibility of pushing the same-class samples apart is reduced if we can avoid regarding the same-class samples as negative examples. The resulting encoder representation can approach the effect of Fig. 2(b), which will be easier and more efficient for training of downstream classification tasks.

B. Contrastive learning for HAR

Self-supervised learning has been applied to human activity recognition based on sensor data, such as Multi-task SSL [29], CAE [36], Masked Reconstruction [37], and CPCHAR [38]. These studies use unlabeled data to generate pseudolabels and to set pretext tasks based on the pseudolabels, enabling the encoder to learn excellent representations by completing the pretext tasks. As a specific paradigm of self-supervised learning, contrastive learning has been applied to human activity recognition. SimCLRHAR [26] first applies contrastive learning to HAR, using SimCLR's architecture, and then achieves a slight improvement in activity recognition accuracy relative to supervised learning. CSSHAR [27] replaces the backbone network with a custom transformer. Although the overall accuracy is improved, the performance improvement relative to the supervised learning of the backbone network is not significant. MoCoHAR [28] uses MoCo's architecture and proposes a resampling sensor data augmentation method, which significantly improves the accuracy with a small amount of data labels compared with supervised learning. However, the contrastive loss function of the above work is InfoNCE [39] or NT-Xent [40]; both are essentially the instance discrimination task that will regard the same-class samples as negative examples. Such a pretraining task tends to cause overclustering so that the same class of activity representation is pulled apart.

To alleviate this issue, we improve the NT-Xent contrastive

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945

loss function by introducing an unsupervised clustering technique to mask the same-cluster samples. This approach avoids the same-cluster sample representation as negative examples to the extent that the same-cluster samples in negative examples are eliminated in calculating contrastive losses. This operation will enable the representation of the same-class activities to be closer in the latent space.

III. Methods

A. Proposed ClusterCLHAR Framework

The framework of the proposed ClusterCLHAR is shown in Fig. 4. The steps in order are data augmentation, encoder, projection head, negative selection, and contrastive loss function. Our work focused on improving the last two components.

Negative Selection: As shown in the dashed box in Fig. 4, we optimize the negative selection method defined by the instance discrimination task. For the first branch, each sample representation was assigned a cluster label by using unsupervised clustering methods. Then representations with the same-cluster label will no longer constitute negative pairs. The similarity matrix of representations of the two branches is shown in the right part of Fig. 4. For each row of this matrix, the previous contrastive learning pretext task, such as SimCLR, is to maximize the similarity of positive pairs. Our proposed framework inherits this idea, but we do not define representations of the same-cluster as negative pairs. This change will potentially avoid the same-class sample representations being pulled apart under the latent space and effectively alleviate overclustering.

Contrastive Loss Function: Due to the change in the negative selection method, the corresponding contrastive loss function needs to be improved. The original contrastive loss function and the improvement process are detailed below.

The instance discrimination task of SimCLR, using NT-Xent [40] (normalized temperature-scaled cross-entropy loss, in Eq. 1) as the contrastive loss function, could cause overclustering and distance the representation of the same-class instances for the downstream classification task.

In Eq. (1), (i, j) is a positive pair; $\boldsymbol{z}_i$ denotes the representation output by the projection head; N denotes the mini-batch length; $I_{[\cdot]}$ is an indicator function, which is equal to 1 when the expression in $[\cdot]$ is true and 0 otherwise; and τ is the temperature coefficient. The formula shows that sample i constitutes negative pairs for all samples except sample j. This finding means that sample i will regard samples of the same class as negative examples, which tends to cause overclustering of pretrained representations.

$$l_{i,j}^{NT-Xent} = -\log \frac{exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_j/\tau)}{\sum_{k=1}^{2N} I_{[k\neq i]} exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_k/\tau)} \tag{1}$$

To alleviate the phenomenon of overclustering and to avoid pulling away the same-class representations, a new contrastive loss function Cluster-NT-Xent is proposed, as shown in Eq. (2), where $cluster(i)$ denotes the set of clusters in which sample i is located. The batch calculation is shown in Eq. (3). We cluster the representations generated by the first branch and assign a cluster label to each representation. Representations with the same cluster label will no longer constitute negative pairs when computing the contrastive loss. The difference between Cluster-NT-Xent and NT-Xent is visually illustrated on the right part of Fig. 4. It can be seen that the sample representations with the same label no longer constitute negative pairs.

Algorithm 1: ClusterCLHAR Pseudocode, PyTorch-like.

```
# f, g: encoder and projection head
# N: batch size
# t: temperature
# cluster: clustering methods, such as K-means

LARGE_NUM=1e9
for x in loader: # load a minibatch x with N samples
   x1, x2 = aug(x), aug(x) # random augmentation

   z1, z2 = g(f(x1)), g(f(x2)) #forward

   p1 = normalize(z1)
   p2 = normalize(z2)

   labels = range(N)
   loss_a = CrossEntropyLoss(sim(p1,p2), labels)
   loss_b = CrossEntropyLoss(sim(p2,p1), labels)
   loss = loss_a + loss_b

   loss.backward() # back-propagate
   update([f.params, g.params]) # Adam update

# similarity of positive and negative pairs
def sim(p1,p2):
   logits_ab = matmul(p1, p2.T)/t #(N,N)
   logits_aa = matmul(p1, p1.T)/t #(N,N)

   # mask generation by clustering
   p1_clu = cluster.fit_predict(p1.cpu()) #(N,)

   masks_aa = [i == p1_clu for i in p1_clu] #(N,N)
   logits_aa = logits_aa - masks_aa * LARGE_NUM #(N,N)

   masks_ab = masks_aa - eye(N) #(N,N)
   logits_ab = logits_ab - masks_ab * LARGE_NUM #(N,N)

   return concat([logits_ab, logits_aa], axis=1)
```

$$l_{i,j}^{Cluster} = -\log \frac{exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_j/\tau)}{\sum_{k=1}^{2N} I_{[k\notin cluster(i)]} exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_k/\tau)} \tag{2}$$

$$\mathcal{L}^{Cluster} = \frac{1}{2N} \sum_{k=1}^{N} [l_{2k-1,2k}^{Cluster} + l_{2k,2k-1}^{Cluster}] \tag{3}$$

The ClusterCLHAR pretraining pseudocode is shown in Algorithm 1. After pretraining, the encoder is used for downstream tasks.

B. Influence of the hypeparameter α

The performance of clustering methods has a significant impact on model performance. If the clustering results differ significantly from the true classification, then many true negative pairs will be eliminated, seriously affecting the pretraining performance. In clustering methods, samples close to the cluster centroids are more likely belong to the same class, than those close to the cluster boundaries [41]. Those samples at cluster boundaries are not suitable for our proposed negative selection strategy (samples of the same cluster do not constitute negative pairs) because they do not have a high probability of belonging to the same class. Therefore, we decided to reduce the proportion of samples using our

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945

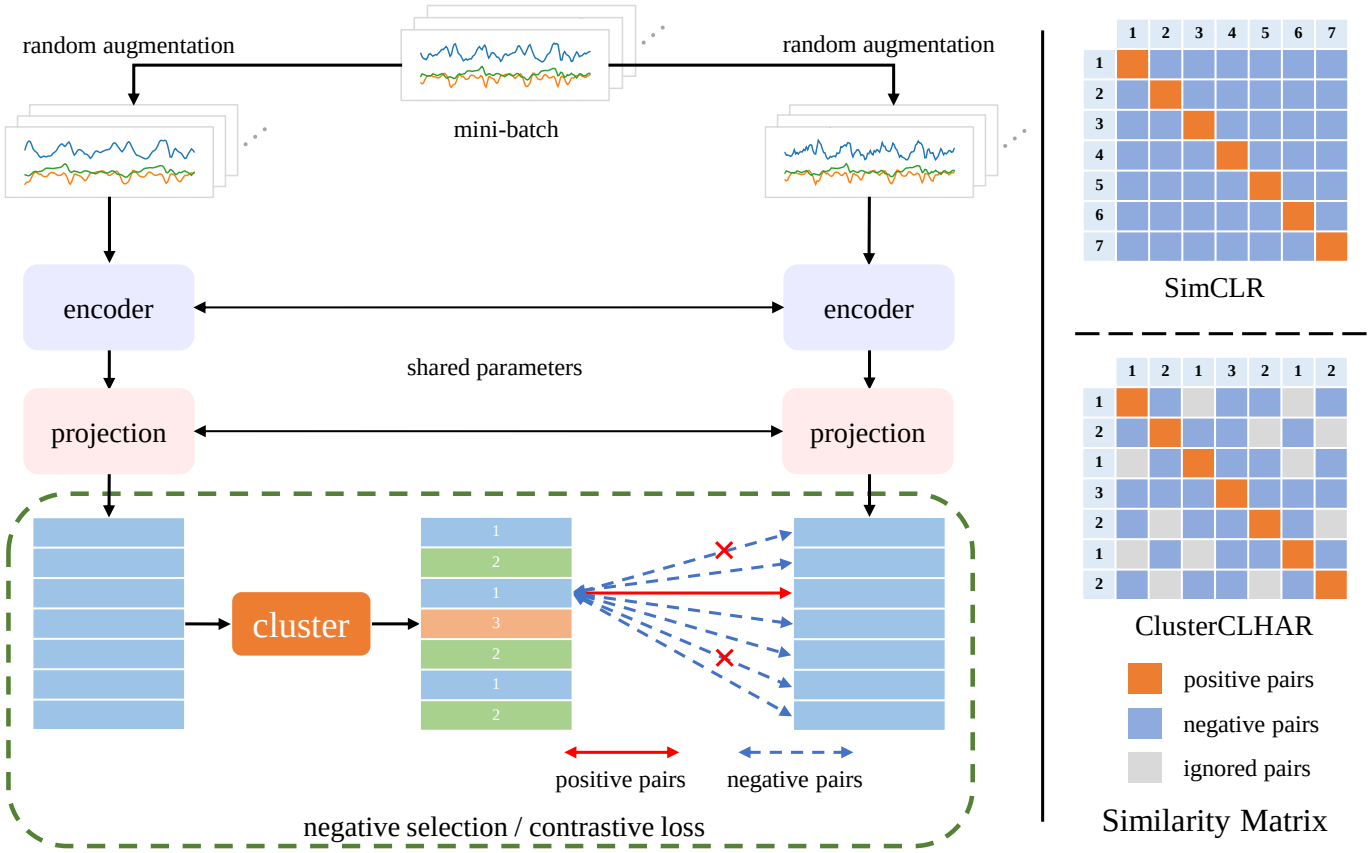


Fig. 4. Negative Selection by Clustering for Contrastive Learning in Human Activity Recognition (ClusterCLHAR). The left part of the figure shows the forward propagation and the selection of negative examples for the pretraining process of ClusterCLHAR. The right part of the figure shows the similarity matrix for the calculation of the contrastive loss, where the row and column numbers are the cluster labels.

proposed negative selection strategy to reduce the negative impact of the difference between the clustering results and the true classification. We set a hyperparameter α indicating the proportion of samples close to the cluster center, in which the proportion of samples uses our proposed negative selection strategy and the rest of the samples use SimCLR's strategy (instance discriminant). Fig. 5 shows the sample distribution using our strategy and SimCLR's strategy at different α. Here, the samples with our proposed strategy use Cluster-NT-Xent as the loss function and the samples with SimCLR's strategy use NT-Xent as the loss function.

$$l_{i,j}^{\alpha} = -\log \frac{exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_j/\tau)}{\sum_{k=1}^{2N} I_{[k \notin cluster(i) \cup \neg thr(\alpha,i)]} exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_k/\tau)} \tag{4}$$

The contrastive loss function Cluster-NT-Xent-α (normalized temperature-scaled cross-entropy loss for selecting negatives by the hypeparameter α) compatible with the above two strategies is shown in Eq. (4), where $thr(\alpha, i)$ indicates that the value is 1 if sample i belongs to the top $\alpha\%$ of the sample representation closest to the cluster center and is 0 otherwise. Here, $\alpha \in [0, 100]$ denotes the proportion of samples that are close to the cluster center, and these samples use our negative selection strategy. When α decreases, the number of samples using our strategy decreases, and the number of negative pairs increases. Eq. (4) not only sets the samples that do not belong to the cluster in which sample i is located as negative examples but also considers samples that belong to the cluster in which sample i is located but do not belong to the top $\alpha\%$ near the cluster center as negative examples. In simple terms, the top $\alpha\%$ of samples near the cluster center will calculate the contrastive loss according to Cluster-NT-Xent, while samples that do not belong to the above range will calculate the loss according to NT-Xent. When α equals 100, the mathematical model is converted to Cluster-NT-Xent, and when α equals 0, the equation actually becomes NT-Xent.

IV. Experiments

A. Datasets

The USC-HAD [31] dataset was collected on the MotionNode sensing platform and contained accelerometer and gyroscope data. This dataset consists of data from 14 subjects recording 12 activities, including walking forward, walking left, walking right, going upstairs, going downstairs, running forward, jumping, sitting, standing, sleeping, and riding the elevator up and down. All data were collected at a 100 Hz sampling rate.

The MotionSense [32] dataset consists of time-series data generated by accelerometer and gyroscope sensors. An iPhone 6s was placed in the participant's front pocket, and information

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945

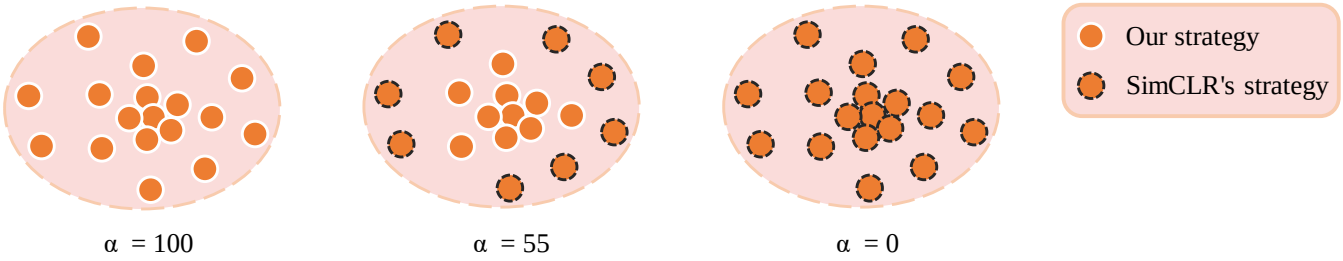


Fig. 5. Distribution of samples with two strategies at different α. The ellipse represents the range of a cluster. α denotes the proportion of samples that are close to the cluster center, and these samples use our negative selection strategy.

was collected from the core motion framework on the IOS device using SensingKit. All data were collected at a 50 Hz sampling rate. A total of 24 participants of different genders, ages, weights, and heights performed six activities: going downstairs, going upstairs, walking, jogging, sitting, and standing in 15 trials in the same environment and under the same conditions.

The UCI-HAR [33] activity recognition dataset was collected from 30 subjects who performed basic activities and postural transitions while carrying a waist-mounted smartphone with embedded inertial sensors. Six basic activities were included: standing, sitting, lying, walking, going upstairs and going downstairs. Experiments captured 3-axis linear acceleration and 3-axis angular velocity at a constant 50 Hz rate using the device's built-in accelerometer and gyroscope.

The experiments in this paper will use the accelerometer and gyroscope data from the above datasets.

B. Self-supervised experimental protocol

According to work [29], [26], the USC-HAD and MotionSense datasets were segmented with 400 sample points as a sliding window with 50% overlap between two windows. According to work [33], [28], the UCI-HAR is segmented with 128 sample points as a sliding window with 50% overlap between two windows. Based on experimental protocols from previous work [36], [29], USC-HAD sensor data from subjects 11 and 12 were selected for validation, while data from subjects 13 and 14 comprised the test set. According to the splitting protocol of MotionSense and UCI-HAR, 20% of subjects are selected for the test set. Out of the remaining subjects, 20% are selected for the validation set, and the rest comprised the training set. Five-fold cross validation was used to accomplish the aforementioned examination, and the average value was used as the experimental result.

For pretraining, ClusterCLHAR uses resampling [28] as the augmentation method and Transformation Prediction Network (TPN) [29] as the encoder, where the resampling augmentation method obtains samples at a new sampling frequency using upsampling and downsampling operations. The projection header uses three layers of MLP, all with 96 neurons, and ReLU as the activation function between two layers. The optimizer uses Adam [42] with an initial learning rate of 1e-3. The temperature coefficient is 0.1 and the model is trained for 200 epochs. The batch size is 1024 for the USC-HAD dataset and 256 for the MotionSense and UCI-HAR datasets. All deep learning codes are built on the TensorFlow [43] platform. An NVIDIA GeForce RTX 3090 GPU was utilized to accelerate the training process. The contrastive loss function is Cluster-NT-Xent. The clustering method applied to USC-HAD and MotionSense is K-means [44], [45], and that applied to UCI-HAR is BIRCH [46]. The number of clustering centers defaults to the true number of classifications in the dataset. The pretraining task uses the unlabeled data from the training set, and the network parameters are initialized using randomization. After pretraining, the model is thrown off the projection header, keeping only the encoder and freezing all layers, with a trainable linear classification layer added at the end of the model. In the downstream task, the optimizer uses Adam with an initial learning rate of 10. The model is trained for 200 epochs, using the mean F1-score [47] as the evaluation metric. All experiments were trained ten times, and the results were averaged. Note that the pretraining task uses the methods of Section III-A by default, and the methods of Section III-B will be shown in Section VI-C.

C. Semi-supervised experimental protocol

This part of the experiment is designed to evaluate the performance of our proposed framework under inadequate data labels. In the data preprocessing, we narrowed the sliding window to be more relevant to the situation of insufficient labels. Based on previous work [28], the USC-HAD and MotionSense datasets were segmented using a sliding window of 200 samples, with 25% overlap for USC-HAD and 12.5% overlap for MotionSense. The UCI-HAR dataset maintains the settings of the self-supervised experimental protocol. To simulate the situation of insufficient labels, this paper uses two training set proportion settings of 1% and 10% in the downstream task. The training set data is selected by using different random seeds.

Pretraining uses all unlabeled data from a single dataset with a training batch size of 1024. The other pretraining settings are the same as the self-supervised learning experimental protocol. To more comprehensively evaluate the pretraining performance of our proposed framework, two evaluation protocols are used in the downstream task.

Linear evaluation: Freeze all layers of the encoder and add a trainable linear classification layer at the end of the model. The optimizer uses Adam with an initial learning rate of 1e-1.

Fine-tuning: Unfreeze the last two layers of the encoder and add a trainable linear classification layer to the end of the

model. The optimizer uses Adam with an initial learning rate of 1e-2.

The loss function for downstream tasks uses cross-entropy. The batch size is 50 and 500 according to the proportion of 1% and 10%, respectively, of different training sets. The model is trained for 200 epochs, and the mean F1-score is selected as the evaluation metric. All experiments were trained ten times, and the results were averaged.

V. Experimental Results

A. Self-supervised learning

In this subsection, we evaluate the performance of ClusterCLHAR in the activity classification task. We compare state-of-the-art self-supervised learning work, where the training and test sets are divided in the same way. Furthermore, we quantify the performance of the self-supervised learning framework's backbone networks on supervised learning. They are 1D Conv, Transformer, and TPN (the backbone networks of CPCHAR, CSSHAR, and ClusterCLHAR, respectively). This comparison shows the difference between supervised and self-supervised learning under the same backbone network with a large amount of labeled data. The experimental results are shown in Table I, and the results of the state-of-the-art work are obtained from [38], [27].

The experimental results show that our proposed ClusterCLHAR outperforms all state-of-the-art self-supervised learning methods for three benchmark datasets. This finding demonstrates that our proposed framework can better extract data representations with unlabeled data and is effective in downstream activity recognition tasks. ClusterCLHAR outperforms supervised learning of the corresponding backbone network on the USC-HAD dataset and is inferior to it on the MotionSense and UCI-HAR datasets. We speculate that this is because supervised learning performs significantly worse on the USC-HAD dataset than on the other two datasets, which gives our proposed framework more space for improvement to outperform supervised learning on the USC-HAD dataset. For the MotionSense and UCI-HAR datasets, we believe this is because supervised learning with a clear goal can better utilize a large amount of labeled data relative to contrastive learning with the goal of learning robust data representations. Finally, it is worth noting that while the backbone network TPN of ClusterCLHAR is inferior to the backbone network Transformer of CSSHAR in supervised learning, ClusterCLHAR outperforms CSSHAR in self-supervised learning, demonstrating the effectiveness of our suggested negative selection strategy.

B. Semi-supervised learning

In this subsection, we evaluate the performance of our proposed method in the context of contrastive learning and a small amount of data labels. We use the TPN as a benchmark for supervised learning and compare our proposed framework with three previous contrastive learning works applied to human activity recognition, SimCLRHAR [26], MoCoHAR [28], and NNCLR [25] [21]. For a fair comparison, we use resampling data augmentation for SimCLRHAR.

Note that our comparison focuses on the difference in model performance by the negative example definition method in contrastive learning. The experimental results are shown in Table II, and 95% confidence intervals were calculated.

The experimental results show that our proposed framework outperforms all state-of-the-art contrastive learning methods in both linear evaluation and fine-tuning for all three datasets. Moreover, the confidence intervals of our method do not overlap with those of state-of-the-art methods in most experimental settings. With 1% data labels, ClusterCLHAR outperforms the previous best contrastive learning work by 4.82% for USC-HAD, 3.3% for MotionSense and 4.89% for UCI-HAR with linear evaluation. ClusterCLHAR outperforms the previous best work by 3.39% on USC-HAD, by 1.16% on MotionSense and by 3% on UCI-HAR with fine-tuning. SimCLRHAR (resampling) is most similar to our work, and the only difference is the contrastive loss function (pretraining task). Our method outperformed SimCLR (resampling) by 5.7%, 2.93%, and 5.25% for the three datasets with linear evaluation of 1% data labels. This finding demonstrates that our approach to masking same-cluster negative examples in the contrasting loss function has a positive impact on the downstream classification task and effectively alleviates overclustering.

With the 1% labeled data, our method performs better than supervised learning in both linear evaluations and fine-tuned evaluations for the USC-HAD and MotionSense datasets. However, our approach achieves competitive performance with supervised learning on UCI-HAR possibly because under supervised learning the model is easier to train with the UCI-HAR dataset relative to other datasets. This phenomenon compresses the improvement space of contrastive learning, rendering contrastive learning models without an explicit task insignificantly worse than supervised learning with an explicit classification task.

VI. Ablation Studies

In this section, we perform ablation studies to investigate the impact of certain model details on the final performance. Suggestions for selecting hyperparameters based on relevant experimental results are also presented. The experiments in this section follow the semi-supervised experimental protocol, using 1% labeled data and fine-tuned evaluation by default.

A. Clustering Methods

The clustering results are critical to the performance of ClusterCLHAR. Assuming that the clustering results are close to or equal to the true classification, the contrastive learning model will encode more significant differences in the samples of different classes to better serve the downstream task. Here, we discuss the impact of using different clustering methods and corresponding clustering similarity measures on the final model performance when performing negative selection. The clustering methods applied in this subsection are K-means [44], [45], DBSCAN [48], Hierarchical clustering [49] and BIRCH [46]. The number of true classifications is employed as the cluster number by default. The experimental results are shown in Table III.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945

TABLE I
SELF-SUPERVISED LEARNING

Method	Type	Backbone	USC-HAD	MotionSense	UCI-HAR
1D Conv [38]	Sup.		49.09	86.66	79.79
Transformer [27]	Sup.		**60.56**	-	**95.26**
TPN [29]	Sup.		55.60	**93.00**	94.27
Multi-task SSL [29]	SSL	TPN	45.37	83.30	80.20
CAE [36]	SSL	CNN	48.82	82.50	80.26
Masked Reconstruction. [37]	SSL	Transformer	49.31	88.02	81.89
CPCHAR [38]	SSL	1D Conv	52.01	89.05	81.65
CSSHAR [27]	SSL	Transformer	57.76	-	91.14
ClusterCLHAR (ours)	SSL	TPN	**58.85**	**90.95**	**92.63**

"Sup." indicates supervised learning, and "SSL" indicates self-supervised learning. The evaluation criteria for the above results are F1-score. In the MotionSense and UCI-HAR datasets, 20% of the subjects' data were used as the test set, with a five-fold cross-validation to obtain experimental results. In the USC-HAD dataset, data from subjects 13 and 14 were used as the test set.

TABLE II
SEMI-SUPERVISED LEARNING

	1%		10%	
	Linear.	Fine.	Linear.	Fine.
USC-HAD				
TPN (Sup.) [29]	70.23 [69.11, 71.36]		85.93 [85.51, 86.36]	
SimCLRHAR [26]	44.84 [44.12, 45.56]	63.82 [62.29, 65.35]	53.16 [52.62, 53.71]	83.32 [82.84, 83.82]
SimCLRHAR (resampling [28])	71.48 [69.90, 73.06]	72.87 [71.47, 74.26]	83.51 [83.30, 83.71]	85.50 [85.00, 85.99]
MoCoHAR [28]	68.60 [67.66, 69.55]	69.20 [67.87, 70.52]	78.51 [78.21, 78.81]	85.84 [85.65, 86.02]
NNCLR [25]	72.36 [70.97, 73.74]	74.70 [73.84, 75.56]	83.81 [83.41, 84.21]	87.07 [86.65, 87.49]
ClusterCLHAR (ours)	**77.18** [76.23, 78.14]	**78.09** [77.24, 78.94]	**85.01** [84.64, 85.37]	**87.86** [87.60, 88.13]
MotionSense				
TPN (Sup.) [29]	84.91 [83.90, 85.93]		95.06 [94.86, 95.27]	
SimCLRHAR [26]	79.16 [78.11, 80.20]	84.62 [83.88, 85.37]	84.25 [83.98, 84.53]	95.77 [95.54, 95.99]
SimCLRHAR (resampling [28])	83.76 [81.91, 85.61]	86.55 [85.36, 87.73]	92.60 [92.40, 92.79]	96.08 [95.89, 96.27]
MoCoHAR [28]	77.66 [77.56, 77.75]	85.49 [84.60, 86.36]	91.72 [91.57, 91.87]	96.63 [96.36, 96.90]
NNCLR [25]	83.39 [82.20, 84.58]	86.19 [85.13, 87.26]	92.09 [91.74, 92.44]	96.40 [96.24, 96.56]
ClusterCLHAR (ours)	**86.69** [85.91, 87.47]	**87.35** [86.38, 88.37]	**94.02** [93.85, 94.18]	**96.44** [96.30, 96.57]
UCI-HAR				
TPN (Sup.) [29]	90.50 [89.90, 91.20]		95.47 [95.17, 95.77]	
SimCLRHAR [26]	54.52 [53.79, 55.26]	62.44 [61.08, 63.81]	59.14 [58.63, 59.64]	78.27 [77.40, 79.14]
SimCLRHAR (resampling [28])	83.53 [82.23, 84.84]	86.75 [85.71, 87.80]	92.91 [92.68, 93.13]	95.58 [95.47, 95.68]
MoCoHAR [28]	83.56 [83.36, 83.76]	87.41 [86.65, 88.17]	91.89 [91.73, 92.05]	95.49 [95.33, 95.65]
NNCLR [25]	83.89 [83.06, 84.72]	87.53 [86.80, 88.27]	92.59 [92.32, 92.86]	95.55 [95.37, 95.73]
ClusterCLHAR (ours)	**88.78** [87.93,89.62]	**90.53** [89.43, 91.69]	**94.68** [94.51, 94.84]	**95.91** [95.79, 96.02]

The percentages in the table header indicate the proportion of the training set in the downstream task, and the remaining data are used as the test set. "Linear." indicates the linear evaluation and "Fine." indicates the fine-tuned evaluation. "Sup." indicates the performance of the backbone network under supervised learning. Pretraining uses all unlabeled data. The evaluation criterion for the above results is the mean F1-Score. The value inside the square brackets indicates a 95% confidence interval.

TABLE III
MODEL PERFORMANCE UNDER DIFFERENT CLUSTERING METHODS

Method	Metric	USC-HAD	MotionSense	UCI-HAR
K-means	euclidean	**78.09**	87.35	86.03
BIRCH	euclidean	68.41	**87.97**	**90.53**
Hierarchical.	euclidean	76.31	86.33	89.19
	cosine	76.41	87.90	89.61
DBSCAN	euclidean	72.08	86.21	88.58
	cosine	71.82	85.95	86.10

"Metric" denotes the similarity measure used by the clustering method, where "euclidean" denotes the Euclidean distance and "cosine" denotes the cosine similarity.

The experimental results show that the best performing clustering method for the USC-HAD dataset is K-means. K-means has a large gap with other methods, so it is chosen as the primary clustering method for this dataset. The best performing clustering method on the MotionSense dataset is Birch. However, K-means was chosen as the primary clustering method for this dataset because it performs similarly to Birch and follows Occam's Razor. The best-performing clustering method on the UCI-HAR dataset is BIRCH, which has a larger gap with other methods, so it is chosen as the primary clustering method for this dataset. The difference between the best performance and worst performance with different clustering methods was 9.68%, 2.02%, and 4.5% for the three datasets. This result demonstrates that the choice of clustering method has a crucial impact on the performance of ClusterCLHAR. Overall, we recommend k-means as the first clustering method because it is simple, runs faster, and performs well under different distributions.

B. Number of Clusters

For clustering methods, the number of different clusters can also have a large impact on the model performance. We set 2 (batch size/512), 4 (batch size/256), 8 (batch size/128), 16 (batch size/64), and 32 (batch size/32) as the number of clusters to explore their effect on the model performance. K-means was employed as the clustering method for contrastive loss functions for the USC-HAD and MotionSense datasets, and BIRCH was utilized for the UCI-HAR dataset. The experimental results are shown in Table IV.

TABLE IV
MODEL PERFORMANCE WITH DIFFERENT NUMBER OF CLUSTERS

Number of clusters	USC-HAD (12)	MotionSense (6)	UCI-HAR (6)
True	**78.09**	87.35	90.56
2	73.50	85.92	87.20
4	76.94	86.98	89.80
8	77.61	87.11	**90.73**
16	77.31	87.19	90.05
32	76.66	**87.95**	90.35

The number in the table header indicates the true number of categories in the dataset. "True" means that the clustering method uses the true number of categories in the dataset for the number of clusters.

TABLE V
MODEL PERFORMANCE AT DIFFERENT α

Proportion	USC-HAD	MotionSense	UCI-HAR
cluster = true			
$\alpha = 100$	**78.09**	87.35	**90.56**
$\alpha = 95$	75.69	**87.69**	88.66
$\alpha = 90$	74.76	86.32	88.11
$\alpha = 80$	76.25	86.67	87.10
$\alpha = 0$	72.87	85.49	86.45
cluster = 16			
$\alpha = 100$	77.31	87.19	**90.05**
$\alpha = 95$	**77.48**	87.30	89.77
$\alpha = 90$	76.56	**87.48**	89.86
$\alpha = 80$	76.71	86.67	89.27
$\alpha = 0$	72.87	85.49	86.45

"cluster = true" indicates that the number of clusters in pretraining uses the true number of categories in the dataset. "cluster = 16" means the number of clusters used for pretraining is 16. α denotes the proportion of samples that are close to the cluster center, and these samples use our negative selection strategy.

The experimental results show that the difference between the best performance and worst performance with different cluster numbers for the three data sets is 4.59%, 2.03%, and 3.53%. This result demonstrates that the choice of the number of clusters for the same clustering method has a significant impact on the performance of the contrastive learning model. This is because the number of clusters will directly affect the number of negative example pairs, which will impact the model training. In addition, we discover that the model performs better when the number of clusters is close to or slightly larger than the true number of classifications and worse when the number of clusters is less than the true number of classifications. For this reason, we can conclude that when pretraining the unknown classification dataset for contrastive learning, we can choose the number of clusters that is slightly larger than the estimated number of classifications.

C. Influence of the hypeparameter α

In this subsection, we implement the method of Section III-B to control the proportion of samples using our negative selection strategy by setting the hyperparameter α. We will discuss the performance for the downstream activity recognition task with different α. The difference between Cluster-NT-Xent ($\alpha = 100$) and NT-Xent ($\alpha = 0$) is shown in this experiment. Here, we set two cluster numbers: the correct classification number and 16. The experimental results are shown in Table V.

Overall, regardless of the number of clusters that is chosen, the model performs better with a large α. The more closely the idea of Cluster-NT-Xent is followed, the better the performance. However, when all samples are used with our negative selection strategy (α = 100), the performance is not optimal in some cases, which proves that the method of reducing the number of samples using the negative selection strategy plays a role. In other words, this method mitigates the negative impact on the model caused by clustering results that differ from the true classification. In summary, when pre-training with a new dataset, we recommend that the hyperparameter α be set to a value close to 100.

D. Batch size and epochs

This subsection explores the impact of using different batch sizes and epochs on model performance in the pretraining of contrastive learning. The clustering method uses the conclusions of Section VI-A, and the number of clusters uses the true classification number. The experimental results are shown in Fig. 6.

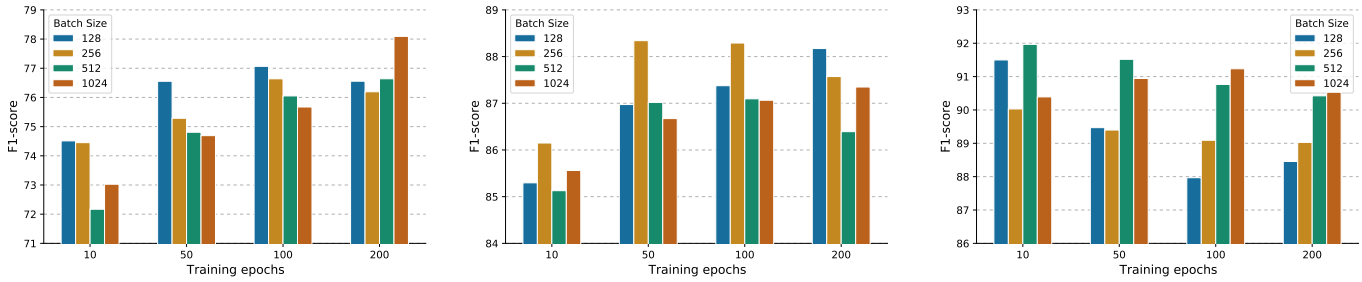
The experimental results show that the three datasets have different sensitivities for different batch sizes and epochs, and this changing trend is not easily captured. The model may overfit as the epoch increases when the batch size is determined. The selection of different batch sizes can also yield significant differences in model performance under deterministic epochs. As the batch size increases, the number of negative pairs also increases, which increases the difficulty of clustering. We speculate that such an unstable trend is that the batch size and epoch affect the performance of the clustering, which in turn affects the performance of the overall model. For this reason, the batch size and training epoch should be carefully chosen when pretraining other datasets.

E. t-SNE

To visualize the effect of the pretraining task, we perform t-SNE [34], [35] dimensionality reduction on the output of ClusterCLHAR (Cluster-NT-Xent) and SimCLR (NT-Xent). On the MotionSense dataset, we randomly select 1000 data samples to obtain 96-dimensional representations (output of the projection header) by pretraining the model with contrastive learning. For these representations using the t-SNE method to reduce the dimensionality to 2 dimensions, the effect is obtained, as shown in Fig. 7.

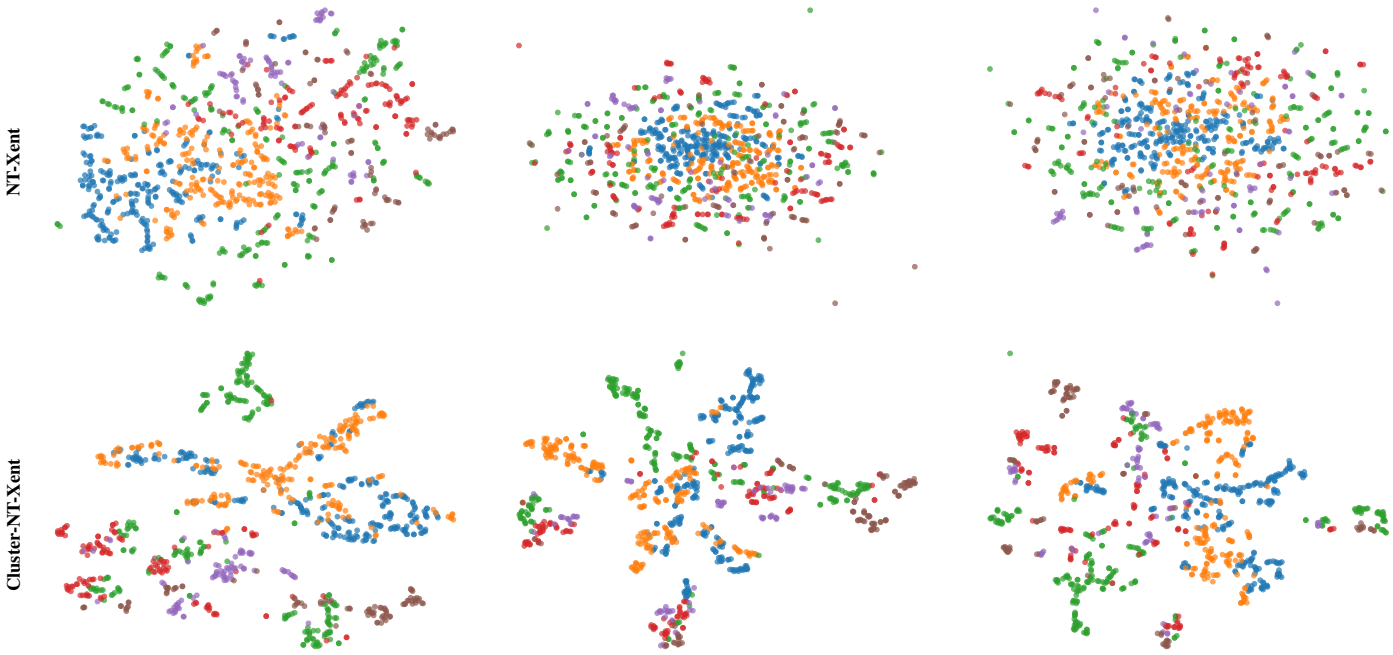
As observed in the figure, the NT-Xent distribution is rather chaotic, approximating that each sample is divided into a single class (similar to Fig. 2(a)). The above situation is consistent with the idea of instance discrimination. Our proposed Cluster-NT-Xent can better aggregate the same class of representations, which is very popular for downstream classification tasks. This finding again demonstrated that the introduction of clustering methods in the contrastive loss function could help the model present excellent performance in the downstream activity recognition task and effectively alleviate

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945



(a) USC-HAD (b) MotionSense (c) UCI-HAR

Fig. 6. Batch size and epochs. The above figure shows the performance of our proposed framework with different batch sizes and epochs during pretraining.



(a) 10 epochs (b) 100 epochs (c) 200 epochs

Fig. 7. t-SNE Visualization. The above is the distribution of the pretrained representation under the latent space, and the real labels are not involved in the generation of these representations.

overclustering. The phenomenon that sample representations of the same class are more similar under the latent space can explain why our proposed method works.

F. Training time

Here we explore the time required for the pretraining of several contrastive learning frameworks. We follow the semi-supervised experimental setup to count the time needed to train for one epoch on the USC-HAD dataset. In addition, the corresponding backbone networks and the number of trainable parameters are presented. Code running on an Intel i9-10900KF CPU and an NVIDIA GeForce RTX 3090 GPU. The clustering method is implemented by scikit-learn, and scikit-learn-intelex is used to accelerate the clustering process. Experimental results are shown in Table VI.

From the experimental results, it can be seen that our suggested technique has the largest time consumption under the TPN backbone, which is because ClusterCLHAR requires the additional computation of clustering in comparison with SimCLRHAR and NNCLR. In addition, the backbone network

TABLE VI
TRAINING TIME

Method	Backbone	Param(K)	Time(s)
SimCLRHAR [26]	TPN	114	0.89
NNCLR [25]	TPN	114	1.15
MoCoHAR [28]	DeepConvLSTM	538	1.63
CSSHAR [27]	Transformer	1190	24.39
ClusterCLHAR (ours)	TPN	114	1.81

The above contrastive learning frameworks all use the resampling augmentation method. "Backbone" denotes the encoder used by contrastive learning frameworks, and "Param" denotes the number of trainable parameters of the model, in thousands. "Time" indicates the time required to train the model for one epoch, in seconds.

has a great impact on the pretraining time consumption. For example, the only difference between SimCLRHAR and CSSHAR is the backbone network, but the difference in time consumption between the two is huge because the GPU does not accelerate the Transformer as well as the CNN. Through the above analysis, we found that time consumption is one of our shortcomings, and we will bridge this gap in terms of backbone networks and clustering optimization.

VII. Conclusions

To alleviate overclustering and avoid distancing the same-class sample representations, we propose a contrastive learning framework named ClusterCLHAR to select negatives by clustering in HAR. It outperforms the contrastive learning framework applied to HAR with the pretraining task of instance discrimination. In addition, ClusterCLHAR also outperforms all state-of-the-art work on self-supervised learning and semi-supervised learning for activity recognition tasks. The effectiveness of our proposed method in alleviating overclustering is demonstrated in the t-SNE visualization experiment. The paper concludes with a detailed discussion of the impact of certain model details. Suggestions for selecting hyperparameters are presented based on the relevant experimental results.

Our proposed framework has some shortcomings, such as limited improvement over supervised learning with all the data labels and longer training time. In addition, we indicated that the performance of clustering can easily affect the final performance of the model in the discussion of model details. Based on the above issues, we decided to optimize the definition of same-cluster samples in the contrastive learning framework. Real-life applications of the contrastive learning framework will also be investigated.

References

[1] P. Rashidi and D. J. Cook, "Keeping the resident in the loop: Adapting the smart home to the user," *IEEE Transactions on systems, man, and cybernetics-part A: systems and humans*, vol. 39, no. 5, pp. 949–959, 2009. I

[2] E. Park, Y. Cho, J. Han, and S. J. Kwon, "Comprehensive approaches to user acceptance of internet of things in a smart home environment," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 2342–2350, 2017. I

[3] L. Gutiérrez-Madroñal, L. La Blunda, M. F. Wagner, and I. Medina-Bulo, "Test event generation for a fall-detection iot system," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6642–6651, 2019. I

[4] S. Kianoush, S. Savazzi, F. Vicentini, V. Rampa, and M. Giussani, "Device-free rf human body fall detection and localization in industrial workplaces," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 351–362, 2016. I

[5] S. Patel, H. Park, P. Bonato, L. Chan, and M. Rodgers, "A review of wearable sensors and systems with application in rehabilitation," *Journal of neuroengineering and rehabilitation*, vol. 9, no. 1, pp. 1–17, 2012. I

[6] X. Zhou, W. Liang, I. Kevin, K. Wang, H. Wang, L. T. Yang, and Q. Jin, "Deep-learning-enhanced human activity recognition for internet of healthcare things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6429–6438, 2020. I

[7] W. Zhang, T. Zhu, C. Yang, J. Xiao, and H. Ning, "Sensors-based human activity recognition with convolutional neural network and attention mechanism," in *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*. IEEE, 2020, pp. 158–162. I

[8] K. Chen, L. Yao, D. Zhang, X. Wang, X. Chang, and F. Nie, "A semisupervised recurrent convolutional attention model for human activity recognition," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 5, pp. 1747–1756, 2019. I

[9] H. Zhang, Z. Xiao, J. Wang, F. Li, and E. Szczerbicki, "A novel iot-perceptive human activity recognition (har) approach using multihead convolutional attention," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1072–1080, 2019. I

[10] Y. Zhao, Q. Li, F. Farha, T. Zhu, L. Chen, and H. Ning, "Indoor activity recognition by using recurrent neural networks," in *Cyberspace Data and Intelligence, and Cyber-Living, Syndrome, and Health*. Springer, 2019, pp. 205–215. I

[11] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, "Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities," *ACM Computing Surveys (CSUR)*, vol. 54, no. 4, pp. 1–40, 2021. I

[12] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, vol. 8, pp. 193 907–193 934, 2020. I

[13] P. Bachman, R. D. Hjelm, and W. Buchwalter, "Learning representations by maximizing mutual information across views," *Advances in neural information processing systems*, vol. 32, 2019. I, II-A

[14] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020. I, II-A

[15] W. Falcon and K. Cho, "A framework for contrastive self-supervised learning and designing a new approach," *arXiv preprint arXiv:2009.00104*, 2020. I

[16] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738. I, II-A

[17] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020. I, I

[18] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607. I, I, I, II-A

[19] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," *Advances in neural information processing systems*, vol. 33, pp. 22 243–22 255, 2020. I

[20] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3733–3742. I, I

[21] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman, "With a little help from my friends: Nearest-neighbor contrastive learning of visual representations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9588–9597. I, II-A, V-B

[22] S. A. Koohpayegani, A. Tejankar, and H. Pirsiavash, "Mean shift for self-supervised learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 326–10 335. I, II-A

[23] G. Wang, K. Wang, G. Wang, P. H. Torr, and L. Lin, "Solving inefficiency of self-supervised representation learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9505–9515. I, I, II-A

[24] J. Robinson, C.-Y. Chuang, S. Sra, and S. Jegelka, "Contrastive learning with hard negative samples," in *International Conference on Learning Representations (ICLR)*, 2021. I, II-A

[25] H. Qian, T. Tian, and C. Miao, "What makes good contrastive learning on small-scale wearable-based tasks?" in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2022. I, V-B, II, VI

[26] C. I. Tang, I. Perez-Pozuelo, D. Spathis, and C. Mascolo, "Exploring contrastive learning in human activity recognition for healthcare," in *Machine Learning for Mobile Health Workshop at NeurIPS*, 2020. I, II-B, IV-B, V-B, II, VI

[27] B. Khaertdinov, E. Ghaleb, and S. Asteriadis, "Contrastive self-supervised learning for sensor-based human activity recognition," in *2021 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2021, pp. 1–8. I, II-B, V-A, I, VI

[28] J. Wang, T. Zhu, J. Gan, L. Chen, H. Ning, and Y. Wan, "Sensor data augmentation by resampling in contrastive learning for human activity recognition," *IEEE Sensors Journal*, 2022. I, II-B, IV-B, IV-B, IV-C, V-B, II, VI

[29] A. Saeed, T. Ozcelebi, and J. Lukkien, "Multi-task self-supervised learning for human activity detection," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 2, pp. 1–30, 2019. I, II-B, IV-B, IV-B, IV-B, I, II

[30] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016. I
[31] M. Zhang and A. A. Sawchuk, "Usc-had: a daily activity dataset for ubiquitous activity recognition using wearable sensors," in *Proceedings of the 2012 ACM conference on ubiquitous computing*, 2012, pp. 1036–1043. I, IV-A
[32] M. Malekzadeh, R. G. Clegg, A. Cavallaro, and H. Haddadi, "Protecting sensory data against sensitive inferences," in *Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems*, 2018, pp. 1–6. I, IV-A
[33] D. Anguita, A. Ghio, L. Oneto, X. Parra Perez, and J. L. Reyes Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proceedings of the 21th international European symposium on artificial neural networks, computational intelligence and machine learning*, 2013, pp. 437–442. I, IV-A, IV-B
[34] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008. I, VI-E
[35] L. Van Der Maaten, "Accelerating t-sne using tree-based algorithms," *The journal of machine learning research*, vol. 15, no. 1, pp. 3221–3245, 2014. I, VI-E
[36] H. Haresamudram, D. V. Anderson, and T. Plötz, "On the role of features in human activity recognition," in *Proceedings of the 23rd International Symposium on Wearable Computers*, 2019, pp. 78–88. II-B, IV-B, I
[37] H. Haresamudram, A. Beedu, V. Agrawal, P. L. Grady, I. Essa, J. Hoffman, and T. Plötz, "Masked reconstruction based self-supervision for human activity recognition," in *Proceedings of the 2020 International Symposium on Wearable Computers*, 2020, pp. 45–49. II-B, I
[38] H. Haresamudram, I. Essa, and T. Plötz, "Contrastive predictive coding for human activity recognition," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 2, pp. 1–26, 2021. II-B, V-A, I
[39] A. Van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv e-prints*, pp. arXiv–1807, 2018. II-B
[40] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," *Advances in neural information processing systems*, vol. 29, 2016. II-B, III-A
[41] L. Rokach and O. Maimon, "Clustering methods," in *Data mining and knowledge discovery handbook*. Springer, 2005, pp. 321–352. III-B
[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International conference on learning representations*, 2015. IV-B
[43] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016. IV-B
[44] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297. IV-B, VI-A
[45] D. Sculley, "Web-scale k-means clustering," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 1177–1178. IV-B, VI-A
[46] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: an efficient data clustering method for very large databases," *ACM sigmod record*, vol. 25, no. 2, pp. 103–114, 1996. IV-B, VI-A
[47] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," *arXiv preprint arXiv:2010.16061*, 2020. IV-B
[48] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *kdd*, vol. 96, no. 34, 1996, pp. 226–231. VI-A
[49] S. Dasgupta and P. M. Long, "Performance guarantees for hierarchical clustering," *Journal of Computer and System Sciences*, vol. 70, no. 4, pp. 555–569, 2005. VI-A

Jinqiang Wang received his B.E. degree from Henan Normal University in 2020. He is currently a M.S. student in the School of Computer Science, University of South China. His research interests include intelligent perception and pattern recognition.

Tao Zhu received his B.E. degree from Central South University, Changsha, China, and Ph.D. from University of Science and Technology of China, Hefei, China, in 2009 and 2015 respectively. He is currently an associate professor at University of South China, Hengyang, China. He is the principal investigator of several projects funded by the National Natural Science Foundation of China and Science Foundation of Hunan Province etc. He is now the Chair of IEEE CIS Smart World Technical Committee Task Force on "User-Centred Smart Systems". His research interests include IoT, pervasive computing, assisted living and evolutionary computation.

Liming Luke Chen is Professor of Data Analytics in the School of Computing, Ulster University, UK. He received his BEng and MEng degrees at Beijing Institute of Technology, China, and DPhil on Computer Science at De Montfort University, UK. His current research interests include pervasive computing, data analytics, artificial intelligence, user-centred intelligent systems and their applications in smart healthcare and cyber security. He has published over 250 papers in the aforementioned areas. Liming is an IET Fellow and a Senior Member of IEEE.

Huansheng Ning received his B.S. degree from Anhui University in 1996 and his Ph.D. degree from Beihang University in 2001. He is currently a Professor and Vice Dean with the School of Computer and Communication Engineering, University of Science and Technology Beijing and China and Beijing Engineering Research Center for Cyberspace Data Analysis and Applications, China, and the founder and principal at Cybermatics and Cyberspace International Science and Technology Cooperation Base. He has authored several books and over 70 papers in journals and at international conferences/ workshops. He has been the Associate Editor of IEEE Systems Journal and IEEE Internet of Things Journal, Chairman (2012) and Executive Chairman (2013) of the program committee at the IEEE international Internet of Things Conference, and the Co-Executive Chairman of the 2013 International Cyber Technology Conference and the 2015 Smart World Congress. His awards include the IEEE Computer Society Meritorious Service Award and the IEEE Computer Society Golden Core Member Award. His current research interests include Internet of Things, Cyber Physical Social Systems, electromagnetic sensing and computing.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2023.3239945

Yaping Wan required Ph.D. degree from Huazhong University of Science and Technology. His research interests include big data causal inference.