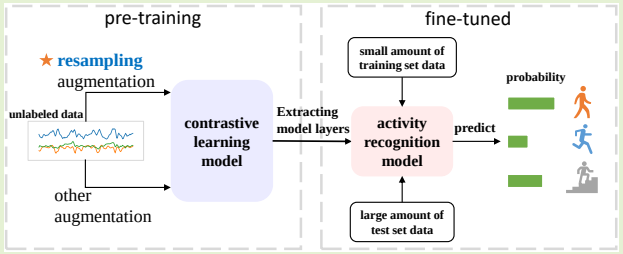
Sensor Data Augmentation by Resampling in Contrastive Learning for Human Activity Recognition

Sensor Data Augmentation by Resampling in Contrastive Learning for Human Activity Recognition

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

***Abstract*— While deep learning models have contributed to the advancement of sensor-based Human Activity Recognition (HAR), it usually requires large amounts of annotated sensor data to extract robust features. To alleviate the limitations of data annotation, contrastive learning has been applied to sensor-based HAR. One of the essential factors of contrastive learning is data augmentation, significantly impacting the performance of pre-training. However, current popular augmentation methods do not achieve competitive performance in contrastive learning for sensor-based HAR. Motivated by this issue, we propose a new sensor data augmentation method by resampling, which introduces variable domain information and simulates realistic activity data by varying the sampling frequency to maximize the coverage of the sampling space. The resampling augmentation method was evaluated in supervised learning and contrastive learning (SimCLRHAR and MoCoHAR). In the experiment, we use four datasets, UCI-HAR, MotionSense, USC-HAD, and MobiAct, using the mean F1-score as the evaluation metric for downstream tasks. The experiment results show that the resampling data augmentation outperforms all state-of-the-art augmentation methods in supervised learning and contrastive learning with a small amount of labeled data. The results also demonstrate that not all data augmentation methods have positive effects in contrastive learning frameworks.**



***Index Terms*— Resampling, Sensor Data Augmentation, Contrastive Learning, Human Activity Recognition, Wearable Sensors.**

I. Introduction

THE development of sensor-based Human Activity Recognition (HAR) technology has brought many intelligent applications into our lives, such as smart homes [1], medical rehabilitation [2], [3] and skill assessment [4]. Due to the popularity of the Internet of Things, sensors can be better embedded into mobile phones, watches, and other portable devices to obtain a data stream more conveniently. By analyzing and inferring sensor data on pervasive computing platforms, computing devices can better understand human activity and help humans with disease prevention [5].

Currently, there are many methods for processing wearable sensor data (accelerometers, gyroscopes, magnetometers), such as traditional machine learning methods including decision trees, Bayesian networks, and support vector machines [6]. In recent years, deep learning has been applied to human activity recognition based on sensor data. Many supervised learning models such as RNN [7], LSTM [8], CNN [9], DeepConvLSTM [10], DeepConvLSTM-Attention [11], and Multi-Head Convolutional Attention [12] have significantly improved the accuracy of HAR. However, these supervised learning methods usually need a large number of labeled data to train a model, which requires manual labeling of sensor data through a time-consuming and tedious process. In addition, the labeling are affected by various noise sources, such as sensor noise, segmentation problems, and changes in the activities of different people, which make the annotation process error-prone [5]. Therefore, the limitation of sensor data annotation is a major challenge for HAR.

To alleviate the limitations of data annotation, contrastive learning was proposed as a main paradigm of self-supervised learning in computer vision, natural language processing, and other fields [13]. Contrastive learning generates two different sets of pseudo-labels through data augmentation, enabling the model to distinguish between positive pairs and negative pairs in these two sets. By learning to pull together different augmented views of the same instance, contrastive learning enables the model to obtain the most essential representation

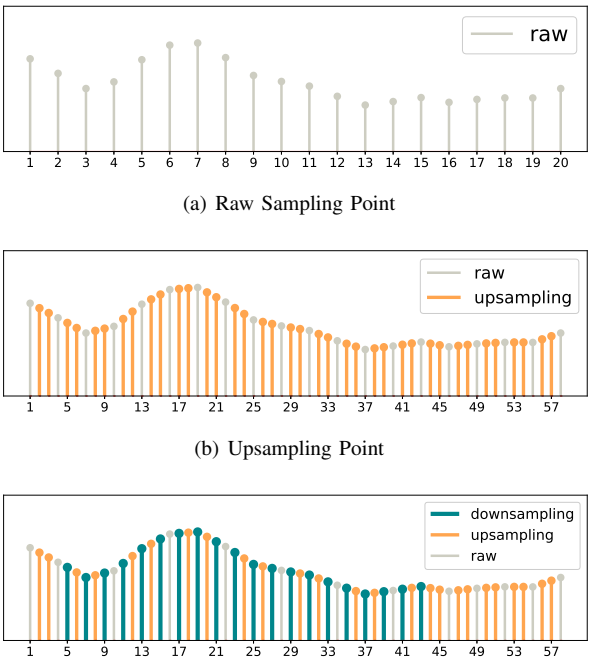
Jinqiang Wang, Tao Zhu, Jingyuan Gan and Yaping Wan are with the School of Computer Science, University of South China, 421001 China (e-mail: tzhu@usc.edu.cn).

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).

Code: `https://github.com/diheal/resampling`

This article has been accepted for publication in IEEE Sensors 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/JSEN.2022.3214198



(a) Raw Sampling Point

(b) Upsampling Point

(c) Downsampling Point

Fig. 1. Resampling Diagram. First, the data in figure (a) is upsampled to generate the data in figure (b). All sampling points in Figure b are the result of upsampling, where the orange line indicates the difference before and after upsampling. Then, the data in figure (b) is downsampled to obtain the data marked by the green line in figure (c).

of the instance. Such a model can perform well in downstream tasks after requiring only a small amount of label data fine-tuning [13], [14], [15]. In contrastive learning, data augmentation plays an important role in generating pseudo-labels by simulating variable domain information existing in reality [16]. However, current sensor data augmentation methods hardly outperform supervised learning even with a small amount of labels and fail to take full advantage of the features of contrastive learning to improve the activity recognition accuracy [17] [18].

Motivated by the limitations of current augmentation methods, we propose a resampling method for sensor data augmentation. The purpose of this method is to simulate realistic data by varying the sampling frequency to maximize the coverage of the sampling space. The resampling method is divided into two steps: upsampling and downsampling. Upsampling is the process of fitting new values by using interpolation methods along the sensor data time axis. In this paper, linear interpolation is used unless otherwise specified. Downsampling is the process of filtering values by random or regular sampling to revert to the length of the raw sample. A demo will be shown here. Fig. 1(a) shows one axis of the acceleration sensor data, which has a total of 20 sampling points. In the process of upsampling, we used linear interpolation to insert two new sampling points between every two raw sampling points. All sampling points in Fig. 1(b) are the result of upsampling, where the orange lines indicate the new sampling points fitted by interpolation. In the downsampling process, we take values at every sample interval until reaching the length of the raw sample. The green line in Fig. 1(c) is the result after downsampling. The whole resampling process is from the gray line sampling points to the green line sampling points.

The resampling augmentation method was evaluated in supervised learning and contrastive learning. In supervised learning, we use DeepConvLSTM as the backbone network with four datasets, UCI-HAR [19], MotionSense [20], USC-HAD [21], and MobiAct [22]. The experiment results show that resampling outperforms all state-of-the-art augmentation methods at 1% and 10% label proportion on the four datasets.

The resampling data augmentation was mainly evaluated on contrastive learning. Since there is less work on contrastive learning applied to HAR, we extend SimCLR [23] and MoCo [24], [25] into HAR-suitable frameworks called SimCLRHAR and MoCoHAR. The above two frameworks are only used to evaluate augmentation methods and are not used for comparison with other work. The resampling augmentation method was evaluated in the SimCLRHAR and MoCoHAR contrastive learning frameworks. In the experiments, we use the UCI-HAR, MotionSense, USC-HAD, and MobiAct datasets. The final experiment results show that the resampling augmentation method outperforms state-of-the-art methods under most settings. Note that SimCLRHAR is similar to the work [17], so it is not included as a contribution point in this paper. Nevertheless, we evaluated the resampling augmentation method in this contrastive learning work [17], see Section VI-B. Finally, we explored the performance of combined augmentation and experiment results found that there are some combined augmentations that outperform the individual augmentations.

The contributions of this paper are as follows: 1. A new sensor-based resampling augmentation method is proposed that outperforms other methods in both supervised learning and contrastive learning. 2. To evaluate the performance of resampling augmentation methods in contrastive learning, we extend MoCo for HAR to include the resampling data augmentation and DeepConvLSTM encoder, which is called MoCoHAR. This framework outperforms SimCLRHAR for big batch sizes.

The remainder of this paper is organized as follows. Section II reviews previous related work and background on sensor data augmentation and self-supervised contrastive learning for HAR. Section III details the resampling data augmentation method and the contrastive learning framework suitable for HAR. Section IV designed experiment protocols for evaluating resampling augmentation methods on supervised learning and contrastive learning. Section V presents the performance of the resampling augmentation method on supervised learning. Section VI presents the performance of the resampling augmentation method on contrastive learning. Section VII summarizes this paper and presents future work based on the identified deficiencies.

II. BACKGROUND

A. Contrastive Learning for HAR

Contrastive learning generates two different sets of pseudo-labels through data augmentation, enabling the model to distinguish between positive pairs and negative pairs in these two

sets. Such a model can perform well with a small amount of label fine-tuning in downstream tasks. Thus contrastive learning can alleviate the problem of lack of label data [14]. The pre-training task of contrastive learning is generally instance discrimination, and its aim is to make different augmented versions of the same instance close to each other, with different instances trying to push apart to obtain the essential features of the raw instance. In computer vision, contrastive learning frameworks represented by SimCLR, MoCo, and BYOL [26] have achieved performance comparable to supervised learning in some datasets for image classification, which shows the great potential of contrastive learning.

At present, a few contrastive learning studies have been applied to HAR. The study [27] proposes a scalogram contrastive network whose objective at a high level is to contrast raw signals and their corresponding visual representations of the wavelet transform so that a network learns to discriminate between aligned and unaligned scalogram-signal pairs. This approach achieves competitive performance in fully supervised networks and outperforms pre-training using auto encoders in both central and federal contexts. But this approach uses only a single transform methods, which does not capture most of the real interference. This study [28] uses Contrastive Predictive Coding [29] for the first time in HAR, and it outperforms supervised learning with a small number of labels. However, this method is more tedious than SimCLR and does not outperform supervised learning in linear evaluation. The study [17] applies SimCLR [23] to HAR for the first time, which uses instance discrimination [30] as the pre-training task and uses NT-Xent [23], [31] as the loss function. In the downstream activity classification task, the fine-tuned model [17] achieved competitive performance. This work also analyzed the performance of different combinations of augmentation methods on contrastive learning. However, experimental results show that a small number of combinations of augmentation methods in contrastive learning outperform supervised learning, and the performance improvement is small. CSSHAR [18] replaces SimCLR's backbone network with a custom Transformer encoder to improve the performance of contrastive learning. SemiC-HAR [32] uses contrastive learning to improve the performance of self-training. However, the innovations in the above two works are frameworks rather than augmentation methods. It is possible to find an augmentation method suitable for sensor data in contrastive learning to improve the performance of activity recognition.

B. Sensor Data Augmentation Methods

Insufficient labeled data is a major challenge in training deep learning models, and data augmentation is an effective approach to alleviate this problem. Data augmentation is also critical to the performance of contrastive learning [16]. Data augmentation can be seen as the injection of a priori knowledge about the invariant properties of data for certain transformations [33]. A key challenge in data augmentation is how to accurately simulate the same class of data under different domain information. In other words, how to ensure that the augmented samples have the same semantics as the raw samples. At present, the commonly used sensor data augmentation methods include [33], [34], [35]:

Noised: A method for simulating additional sensor noise by multiplying the raw sample values with values that match a Gaussian or uniform distribution.

Rotated: A method for simulating different sensor positions by plotting a uniformly distributed 3D random axis and a random rotation angle and applying the corresponding rotation to the sample.

Scaling: Multiply by a random scalar to scale the size of the data in the window to simulate the motion of weaker magnitudes.

Magnify: Multiply by a random scalar to magnify the size of the data in the window to simulate stronger amplitude motion.

Inverting: The sample value multiplied by -1 produces a vertical flip or mirror image of the input signal.

Reversing: The entire window of the sample is flipped in the time direction. The second half of the cycle is simulated by the first half of the cycle motion.

Permutation: A simple method to randomly disturb the window temporal positions. The sample is first split into N segments of the same length, and then, the segments are randomly arranged to create a new window.

Time warping: A method for disturbing the temporal position of a sample can use time warping to change the temporal position of the sample by smoothly distorting the time interval between samples.

Cropping: Randomly crop the raw sample according to a certain time window size.

Shuffling: Randomly disrupted channels of sensor data are used to simulate different wearing directions of the sensor.

Most of the above methods are transferred from time series augmentation methods without considering the characteristics of the sensor data, or the generated new samples cannot well represent the raw label data. Therefore, their performances will fluctuate greatly due to changes in the datasets [33], [34], [35]. The study [17] used the above augmentation methods in contrastive learning, but most of the augmentation methods did not perform better than supervised learning. For this reason, we need to propose an augmentation method that is suitable for contrastive learning.

III. METHODS

Data augmentation has the following functions: Increase the training set data to alleviate model overfitting; Introduce variable domain information to simulate more realistic and multi-view data. For example, the angle, resolution, and color of different pictures of the same cat, taken by different cameras or persons, could be different. Here, angle, resolution, and color can be regarded as domain information, and the essential features of this cat are domain-invariant information. Data augmentation methods generally change the domain information of the sample but do not change the domain-invariant information. The data augmentation method plays an important role in contrastive learning [16]. Contrastive learning changes the domain information of the samples by different data

augmentation methods to obtain different views of the same sample. It then forces the encoder to output the same feature vector for these different views. The purpose is to filter out domain information from the samples and learn the feature representation of the essential domain-invariant information. However, current sensor data augmentation methods do not perform well in contrastive learning [17], and we need to propose a new augmentation method to address this problem.

A. Resampling Augmentation

The sensor collects activity data and is sampled at a certain frequency, so a continuous activity signal is usually represented by a discrete sequence of sensor data. There is domain information during sampling, such as the sampling frequency (Too low a sampling frequency will miss some essential information. Too high a sampling frequency will be more likely to collect noise, especially high-frequency non-smooth noise), initial time point of sampling (phase angle), noise (such as the movement of the device, interference generated inside the device), and the duration of sampling. To better simulate the above domain information and address the limitations of current augmentation methods, we proposed a sensor data augmentation method called resampling. The method introduces variable domain information and simulates realistic activity data by varying the sampling frequency to maximizing the coverage of the sampling space, enabling the model to learn more extensive and robust representations.

The resampling method is divided into two stages: upsampling and downsampling. Upsampling generates new sampling points on the time axis by interpolation methods (e.g., linear interpolation, cubic spline interpolation) to simulate the sampling points after increasing the sampling frequency. In this paper, linear interpolation is used for the interpolation methods unless otherwise specified. Downsampling is taken on the time axis according to random or regular sampling to keep the sample length constant. A concrete example is shown in Fig. 1.

The raw sample data for one axis of the sensor can be expressed as

$$X[i], i = 1, ..., I \tag{1}$$

Its values are arranged in chronological order for a total of I moments. The upsampling process inserts M new moments within every two moments by using linear interpolation, where M is an integer, $M >= 1$, and new moments can be generated according to Eq. (2). Sequence $X^{'}$ is the result of upsampling, and its length is $I^{'} = (M+1) * (I-1) + 1$.

$$X^{'}[(M+1) * (i-1) + k] = X[i] + (X[i+1] - X[i]) * \frac{k-1}{M+1}$$
$$i = 1, ..., I-1; k = 1, ..., M+1$$
$$X^{'}[(M+1) * (I-1) + 1] = X[I] \tag{2}$$

After upsampling, the samples must be downsampled to recover the raw time series length. For upsampling results, downsampling takes the value every N time interval, where N is an integer, $0 <= N <= M-1$. When N=0, it means continuous sampling, and when N=1, it means sampling at every interval. To ensure that the length of the sequence is constant and more approaches to sampling can be taken, the starting point is obtained according to Eq. (3), where $random(a, b)$ means randomly taking a value from a to b. After that, sequence $X^{''}$ is obtained through Eq. (4), and its length is I. $X^{''}$ is the result of downsampling as well as the result of the entire resampling.

$$s = random(1, I^{'} - I * (N+1)) \tag{3}$$
$$X^{''}[i] = X^{'}[s + (i-1) * (N+1)], i = 1, ..., I \tag{4}$$

Algorithm 1: Resampling Pseudocode

Input : one-axis sensor data $\{x[i]\}_{i=1}^{I}$, data length I, number of interpolation M, sampling interval N

Output: sensor data after transformation $\{x^{''}[i]\}_{i=1}^{I}$

```
# upsampling;
for i = 1 to I − 1 do
    for k = 1 to M + 1 do
        # according to Eq. (2);
        # get the index of the new sequence;
        index = (M + 1) * (i − 1) + k;
        # calculate the value corresponding to the
          index;
        value = x[i] + (x[i + 1] − x[i]) * (k−1)/(M+1);
        # assignment;
        x'[index] = value;
x'[(M + 1) * (I − 1) + 1] = x[I];
# downsampling;
# the length of the sequence after upsampling;
I' = (M + 1) * (I − 1) + 1;
# the sampling starting point is obtained according to
  Eq. (3);
s = random(1, I' − I * (N + 1));
for i = 1 to I do
    # according to Eq. (4);
    x''[i] = x'[s + (i − 1) * (N + 1)];
```

The resampling pseudocode using linear interpolation is shown in Algorithm 1.

Here we try to explain the reasons why the resampling augmentation method works. The data obtained from the same activity is different for different people using different sensors with different sampling frequencies. The resampling method proposed in the manuscript, by first interpolating and then down sampling, is actually generating different views of the same activity with different domain information, such as sampling frequencies, sampling durations, and sampling start and end time points. Therefore, with the proposed resampling methods, contrastive learning can filter out these kinds of domain information and learn good essential domain-invariant features.

B. Contrastive Learning for HAR

Contrastive learning has become a popular self-supervised learning paradigm in the field of computer vision, such as SimCLR and MoCo, which have a similar structure and both use instance discrimination [30] as a pre-training task. The general process of contrastive learning is that the raw samples are first augmented with different methods to obtain two sets of samples containing different domain information, which are then encoded by an encoder to obtain a dimension-specific representation. The pre-training task is to distinguish which augmented samples representations are from the same instance and which augmented samples representations are not from the same instance and then to make different augmented representations of the same instance similar and different instances far apart so that to obtain the most essential representation of the raw sample. The model generated by pre-training can better serve the downstream tasks.

To evaluate the performance of the resampling augmentation method in contrastive learning, we follow SimCLR and MoCo, two contrastive learning frameworks applied in computer vision, to construct two contrastive learning frameworks SimCLRHAR and MoCoHAR applied in HAR. The above two frameworks are only used to evaluate augmentation methods and are not used for comparison with other work.

1) SimCLRHAR: SimCLR shows in a simple and intuitive way the general process of contrastive learning in the field of computer vision, where a better result can be obtained by turning up the batch size within a certain range. In this paper, we extend SimCLR for HAR to include the resampling data augmentation and DeepConvLSTM encoder, which is called SimCLRHAR. This framework is shown in Fig. 2.

Data Augmentation: Two different augmentation methods are used for the raw sensor samples to generate two new samples. Alternatively, one branch does not use augmentation and uses the raw sample directly, and the other branch augments.

Encoder: Using DeepConvLSTM as the base encoder, which is a classical framework in sensor activity recognition, two newly generated samples are encoded, which in turn generates a representation of certain dimensions. The encoder parameters are shared between the two branches.

Projection Head: According to the experience of the work [23], we use a nonlinear projection head to remap the representation generated by the encoder to a new dimension representation. The projection header parameters are shared between the two branches.

Contrastive Loss Function: NT-Xent [23], [31] is used as the loss function, and the action objects are Z_i and Z_j generated each time. The loss function formula is as follows:

$$l(i,j) = -\log \frac{exp(sim(i,j)/\tau)}{\sum_{k=1}^{2N} I_{[k\neq i]} exp(sim(i,k)/\tau)} \tag{5}$$

$$\mathcal{L} = \frac{1}{2N}\sum_{k=1}^{N}[l(2k-1,2k) + l(2k,2k-1)] \tag{6}$$

where $sim(a,b)$ is the cosine similarity function, τ denotes a temperature parameter, and N is the batch size. $I_{[k\neq i]}$ is an indicator function, whose value is 1 when k is not equal to i. 2k-1 is a positive sample pair with 2k only, and 2k-1 and other values are negative sample pairs.

Return: Discard the projection header and return to the encoder.

The above is the entire structure of SimCLRHAR. Since it is similar to the work [17], it is not used as a contribution point for this paper, but it includes a new augmentation method and encoder relative to this previous work.

2) MoCoHAR: MoCo expands the negative samples for the contrastive approach by using a memory queue, which in turn can obtain better results with a smaller batch size. In this paper, we extend MoCo for HAR to include the resampling data augmentation and DeepConvLSTM encoder, which is called MoCoHAR. This framework is shown in Fig. 3.

Data Augmentation: Two different augmentation methods are used for the raw sensor samples to generate two new samples. Alternatively, one branch does not use augmentation and uses the raw sample directly, and the other branch augments.

Encoder and Projection Head: Two newly generated samples were encoded using DeepConvLSTM as the basic encoder to generate a representation of certain dimensions. The representation is then remapped to a new dimension using a nonlinear mapping header. The network parameter θ is updated by the back propagation of the neural network, and the network parameter ξ is updated in the following manner:

$$\xi \leftarrow m\xi + (1-m)\theta \tag{7}$$

Here, $m \in [0,1)$ is the momentum coefficient, the queue shape is (K,P), where K represents the capacity of the queue, and P is the dimension finally generated by the projection head. The Z_j generated each time will be added to the queue. If the queue stack is full, the earliest data added to the queue will be overwritten.

Contrastive Loss Function: InfoNCE [29] is used as the loss function, and the action objects are Z_i and the queue generated each time. The loss function formula is as follows:

$$\mathcal{L}_q = -\log \frac{exp(q\cdot k_+/\tau)}{\sum_{i=1}^{K} exp(q\cdot k_i/\tau)} \tag{8}$$

where q is Z_i generated each time by the network, k_+ is the positive sample that corresponds to q in the queue, and τ is the temperature parameter.

Return: Return only the encoder for which the network parameter is θ, discarding all other structures.

IV. Experiment

A. Datasets

The UCI-HAR [19] activity recognition dataset was built from the recordings of 30 subjects performing basic activities and postural transitions while carrying a waist-mounted smartphone with embedded inertial sensors. Six basic activities were included: standing, sitting, lying, walking, upstairs and 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. In this experiment, 128 readings are sampled as a sliding window, and the sliding window has 50% overlap.

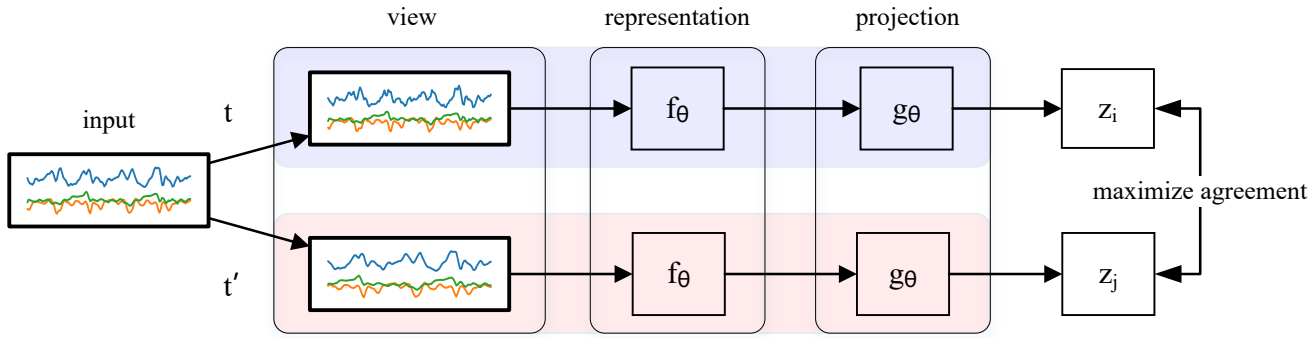


Fig. 2. SimCLR for Human Activity Recognition (SimCLRHAR). "view" indicates the samples generated by different augmentation methods. f denotes the encoder and g denotes the projection head. The upper and lower branches share a set of parameters θ. Only the encoder f_θ is used in downstream tasks.

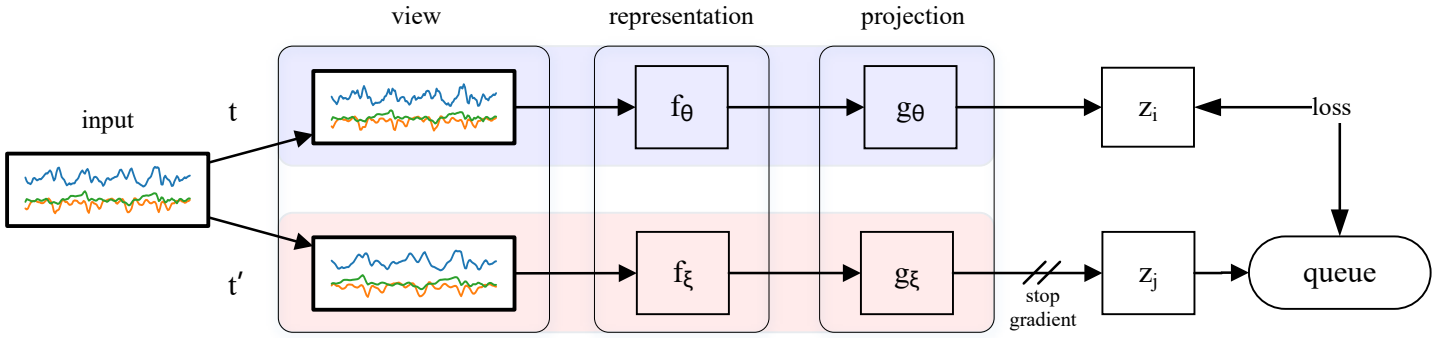


Fig. 3. MoCo for Human Activity Recognition (MoCoHAR). The main structure is similar to the SimCLRHAR. The difference is that the upper and lower branches do not share parameters, and the lower branch uses momentum to update parameters. Only the encoder f_θ is used in downstream tasks.

The MotionSense [20] 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 was collected from the core motion framework on the IOS device using SensingKit. All of the data was collected at a 50 Hz sampling rate. A total of 24 participants of different genders, ages, weights and heights performed six activities: downstairs, upstairs, walking, jogging, sitting and standing in 15 trials under the same environment and conditions. In this experiment, 200 readings are sampled as a sliding window, and the sliding window has a 12.5% overlap.

The USC-HAD [21] dataset was collected on the MotionNode sensing platform and contains accelerometer and gyroscope data. The dataset consists of data from 14 subjects, which recorded 12 activities, including walking forward, walking left, walking right, going upstairs, going downstairs, running forward, jumping, sitting, standing, sleeping, going up and going down the elevator. All data were collected at a 100 Hz sampling rate. In this experiment, 200 readings are sampled as a sliding window, and the sliding window has a 25% overlap.

The MobiAct [22] dataset was collected from inertial measurement unit data by a Samsung Galaxy S3 smartphone placed in a pocket. It consists of accelerometer, gyroscope and orientation signals. This dataset contains eleven daily behavioral activities and four types of falls. We use the second version of this dataset and select only daily behavior data. This part of the dataset was completed by 61 participants and contained activities such as standing, walking, jogging, jumping, stairs up, stairs down, stand to sit, sitting on chair, sit to stand, car-step in, and car-step out. All data were collected at a 200 Hz sampling rate. In this experiment, 128 readings are sampled as a sliding window, and the sliding window has a 50% overlap.

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

B. Data Augmentation

Based on the research [17], [36], this paper adopts the following parameter setting augmentation method:

Noised: Add random noise signals with a maximum value of 0.1 and a minimum value of -0.1 to the data sample, which are subject to a uniform distribution.

Rotated: Draw a uniformly distributed three-dimensional random axis and a random rotation angle, and apply the corresponding rotation to the sample, rotating according to the random angle.

Scaling: Each channel of the signal is scaled by a random value between 0.7 and 0.9.

Magnify: Each channel of the signal is amplified by a random value between 1.1 and 1.3.

Inverting: The sample value times negative 1.

Reversing: The entire window of the sample is flipped in the time direction.

Resampling: Upsampling involves inserting a specified or random number of new sampling points between every

two sampling points, and the interpolation method is linear. Downsampling takes the value with every specified or random time interval.

Due to the poor performance of the other augmentation methods in the work [17], the experiment in this paper no longer compares them.

C. Supervised Learning Experiment Setup

The performance of the resampling augmentation method is first validated on supervised learning, and the experiment backbone network is DeepConvLSTM, which is tested on the UCI-HAR, MotionSense, and USC-HAD datasets. Similar to work [37], [38], to simulate different degrees of labeled data shortages, the training set proportions were set to 1%, 10%, and 60%. The remaining data is used as a test set. The code is built on the TensorFlow [39] platform with an optimizer using Adam [40] and an initial learning rate of 5e-4. The batch size is set to 50, 500, and 1000 according to the different training set proportions of 1%, 10%, and 60%, respectively. The model was trained for 200 epochs, using mean F1-score [41] as the evaluation metric. An NVIDIA Tesla V100 GPU was used to accelerate the training process. All of the experiments following this part of the setup were trained 10 times in different training and test sets divided, and the test results were averaged.

D. Contrastive Learning Experiment Setup

In this paper, two contrastive learning frameworks, SimCLRHAR and MoCoHAR, are used to evaluate the performance of the resampling augmentation method on HAR.

SimCLRHAR: The pre-training task uses DeepConvLSTM as the backbone network and NT-Xent as the contrastive loss function. Three layers of the MLP projection head are added with dimensions 256, 128, and 50. The optimizer uses Adam with an initial learning rate of 1e-3. The temperature is 0.1, the batch size is 2048, and 200 epochs are trained. This pre-training uses all of the data from a single dataset.

MoCoHAR: The pre-training task uses DeepConvLSTM as the backbone network and InfoNCE as the contrastive loss function. Two layers of MLP projection heads with dimensions of 256,128 are added. The optimizer is Adam with an initial learning rate of 1e-3. The temperature is 0.07, K is 8192, m is 0.999, the batch size is 1024, and 200 epochs are trained. This pre-training uses all of the data from a single dataset.

After pre-training, the encoder is obtained. To verify the performance of the pre-training model, this paper adopts two evaluation protocols to evaluate the downstream classification tasks:

Linear evaluation: Freeze the encoder and add a linear classification layer at the end of the model. The optimizer uses Adam, and the initial learning rate is 1e-2.

Fine tuning: Unfreeze the encoder and add a linear classification layer at the end of the model. The optimizer uses Adam, and the initial learning rate is 5e-4.

In both evaluation experiments, the loss function is cross-entropy loss. Similar to work [37], [38], the batch size is set to 50, 500, and 1000 according to different training set proportions of 1%, 10%, and 60%, respectively. The remaining data is used as a test set. Since we mainly compare the differences between the augmentation methods and better simulate the situation of insufficient data labels, we use a random division of the training street and test set. Experiments with training and test sets divided by subject data are presented in Section VI-B. The model was trained for 200 epochs, using mean F1-score as the evaluation metric. All of the experiments were trained 10 times in different training and test sets divided, and the test results were averaged.

TABLE I
HYPERPARAMETRIC SENSITIVITY ANALYSIS

	$M=1$	$M=2$		$M=3$		
	$N=0$	$N=0$	$N=1$	$N=0$	$N=1$	$N=2$
UCI-HAR	**88.20**	86.63	87.76	87.14	88.01	86.89
MotionSense	87.05	86.56	**87.20**	87.02	86.95	86.52
USC-HAD	71.09	69.37	70.85	68.51	**71.63**	70.89
MobiAct	**91.73**	91.07	91.66	90.81	**91.73**	91.23

The above experimental results are produced under supervised learning with a training set proportion of 1%. M denotes the number of interpolations at two moments and N denotes the sampling interval.

V. SENSOR DATA AUGMENTATION METHODS FOR SUPERVISED LEARNING

The default experimental protocol used in this section is presented in Section IV-C.

A. Resampling Hyperparameter Study

To explore the best performance and parameter sensitivity of the resampling augmentation method, two hyperparameters are listed: 1. M denotes the number of interpolations at two moments; 2. N denotes the sampling interval. Before training, the training set samples are augmented four times. The specific experiment results are shown in Table I.

The experiment results show that the values of different parameters of the resampling augmentation method can affect the classification performance of the supervised task to varying degrees. In other words, the results of this experiment demonstrate that different sampling frequencies can impact activity classification performance. This finding is caused by the fact that when humans perform the same movement, the amplitude of the movement we use each time is not exactly the same. For example, when walking, the amplitude of our swinging arms and the span of our legs will be different, and thus, the data collected by the sensors will also be different, and we simulate changing the sampling frequency of the sensors, which is equivalent to generating one more set of the subjects' motion data that is closer to the sample of that subject in the test set.

B. Nonlinear Interpolation Study

There are many nonlinear interpolation methods, such as Lagrange interpolation and cubic spline interpolation. In this subsection, we will explore the performance of replacing linear interpolation with Lagrange interpolation [42] and cubic spline interpolation [43] in the upsampling phase. We use

segmented nonlinear interpolation in the upsampling phase and a randomly clipped segment of continuous samples in the downsampling phase. The experiment results are shown in Table II.

The experiment results show that linear interpolation performs better than nonlinear interpolation in the UCI-HAR, MobiAct, and USC-HAD datasets. In the MotionSense dataset, linear interpolation and nonlinear interpolation perform very similarly. In general, multiple interpolation methods perform similarly. We analyze that no matter which interpolation method is used, they aim to change the sampling frequency. Any interpolation method that conforms to the resampling will likely produce a similar performance. This finding demonstrates that the upsampling phase of the resampling method is suitable for multiple interpolation methods.

C. Comparison with State-of-the-art augmentation methods under supervised learning

To evaluate the performance of resampling augmentation methods on supervised tasks, we compared other existing sensor data augmentation methods. The final experiment results and 95% confidence intervals are shown in Table III, while we performed Wilcoxon signed-rank test [44] by resampling method with other methods, and the non-parametric statistical hypothesis test result is shown in Table IV.

The experiment results show that the resampling method outperforms the supervised learning and all state-of-the-art data augmentation methods for classification performances with 1% and 10% labeled data. In particular, the resampling augmentation method improves significantly under 1% labeled data, outperforming the best method by 4.18%, 0.92%, 3.83%, and 0.88% in the UCI-HAR, MotionSense, USC-HAD, and MobiAct datasets, respectively. At 60% labeled data, the resampling augmentation method outperforms other augmentation methods on the MotionSense and USC-HAD datasets and is insignificantly worse than the magnify method on the UCI-HAR dataset and the noise method on the MobiAct dataset. Moreover, the confidence intervals of our method do not overlap with those of the state-of-the-art methods in most experiment setting. The non-parametric statistical hypothesis test found that the resampling method was significantly better than state-of-the-art augmentation methods in most settings. Even when the resampling method is worse than the other methods, it is insignificantly worse. This finding demonstrates that the resampling augmentation method can alleviate the lack of labeled data and significantly improve activity classification performance relative to previous methods.

D. Comparison under LSTM backbone network

The work [8] proposed a network consisting of 3 layer LSTM, which achieved the best results on the UCI-HAR dataset at that time. We use this backbone network to evaluate the resampling augmentation method and compare it with other augmentation methods. We performed this experiment under the supervised learning experimental setup and the experimental results are presented in Table V.

The experimental results show that the resampling augmentation method outperforms supervised learning and is better than other augmentation methods with a small number of labels. The above conclusion is the same as the case where the backbone network is DeepConvLSTM. Although the resampling augmentation method is not the best at 60% training set proportion, the difference is very small. The above analysis shows that the resampling augmentation method still performs well when the backbone network is 3 layer LSTM.

E. Comparison of Inference Times

In this subsection, we show the comparison of augmentation methods for inferring time. We used 10240 sensor data of the same length to calculate the time consumption required to perform one data augmentation. The experimental setup uses sensor data of different lengths, where the results are averaged after performing 100 times.

Table VI shows the execution time of different augmentation methods. It can be noticed that the execution time of the resampling augmentation method is longer than that of the other methods, which will be a part of our future optimization. However, the time required for the augmentation method is negligible compared to the training time of the deep learning model.

VI. Sensor Data Augmentation for Contrastive Learning

The default experimental protocol used in this section is presented in Section IV-D.

A. Comparison with State-of-the-art augmentation methods under contrastive learning

To evaluate the performance of the resampling augmentation method, it will be used with two contrastive learning frameworks, SimCLRHAR and MoCoHAR, and it will be compared with other augmentation methods at different labeled data proportions. To compare more clearly with other augmentation methods, we do not take augmentation in the augmentation phase for the first branch but use the raw samples, and we use specific augmentation methods for the second branch. The experiment results are shown in Table VII, while we performed Wilcoxon signed-rank test by resampling method with other methods, and the non-parametric statistical hypothesis test result is shown in Table VIII.

First, we analyze the experimental results under the linear evaluation protocol. The experiment results show that resampling augmentation methods outperformed all state-of-the-art methods. In particular, with 1% labeled data, the resampling augmentation method was improves significantly on SimCLRHAR and MoCoHAR, outperforming the best method by 4.27% and 8.26% on the UCI-HAR, 9.46% and 0.14% on the MotionSense, 23.75% and 8.68% on the USC-HAD, 24.2% and 1.01% on the MobiAct respectively. Moreover, the confidence intervals of resampling method do not overlap with those of the state-of-the-art methods in most settings. Note that contrastive learning frameworks with the resampling

TABLE II
NONLINEAR INTERPOLATION

Mode	UCI-HAR			MotionSense			USC-HAD			MobiAct		
	Linear	Lagrange	Cubic Spline	Linear	Lagrange	Cubic Spline	Linear	Lagrange	Cubic Spline	Linear	Lagrange	Cubic Spline
A	-	86.16	85.42	-	86.35	86.32	-	70.83	71.03	-	90.96	90.92
B	-	85.45	87.25	-	87.35	87.24	-	71.16	71.02	-	91.17	91.04
Best	88.20	86.16	87.25	87.20	87.35	87.24	71.63	71.16	71.03	91.73	91.17	91.04

This experiment was performed under supervised learning with 1% training set proportion. Linear, Lagrange, and Cubic Spline denote different interpolation methods. Mode A is to fit a curve for every four consecutive sample points in each channel of sensor data, inserting a sample point between the 2nd and 3rd sample points. Mode B is to fit a curve for every eight consecutive sampling points, inserting one sample point between the 2nd and 3rd sampling points, the 4th and 5th sampling points, and the 6th and 7th sampling points, respectively. "Best" indicates the largest value of the two modes.

TABLE III
COMPARISON WITH STATE-OF-THE-ART AUGMENTATION METHODS UNDER SUPERVISED LEARNING

	UCI-HAR			MotionSense			USC-HAD			MobiAct		
	1%	10%	60%	1%	10%	60%	1%	10%	60%	1%	10%	60%
Supervised	77.94 [75.38,80.50]	94.46 [94.30,94.62]	96.13 [95.78,96.48]	81.13 [79.02,83.24]	96.23 [96.01,96.46]	98.62 [98.49,98.74]	63.93 [62.28,65.58]	85.58 [85.22,85.93]	90.94 [90.78,91.10]	89.76 [89.44,90.07]	96.41 [96.34,96.48]	98.32 [98.29,98.36]
Noise	80.14 [77.93,82.35]	95.12 [94.87,95.36]	97.95 [97.55,98.35]	80.14 [78.44,81.83]	96.55 [96.43,97.04]	99.08 [98.97,99.19]	65.31 [63.67,66.94]	85.77 [85.58,85.97]	90.52 [90.34,90.70]	89.99 [89.80,90.19]	96.53 [96.45,96.60]	**98.39** [98.34,98.45]
Rotated	76.52 [75.37,77.67]	93.22 [92.94,93.49]	95.48 [95.08,95.87]	78.29 [76.87,79.72]	94.04 [94.66,95.75]	98.13 [98.02,98.25]	53.78 [52.36,55.19]	81.22 [80.77,81.68]	87.66 [87.40,87.92]	89.46 [88.98,89.94]	95.41 [95.29,95.54]	97.65 [97.59,97.72]
Scaling	81.55 [79.33,83.78]	94.91 [94.68,95.14]	98.23 [97.65,98.82]	81.97 [80.68,83.25]	95.73 [95.68,96.31]	98.86 [98.74,98.97]	64.94 [63.37,66.51]	85.36 [85.09,85.63]	91.00 [90.82,91.18]	90.03 [89.76,90.30]	96.25 [96.16,96.34]	98.32 [98.29,98.34]
Magnify	83.61 [82.36,84.87]	95.11 [94.84,95.39]	**98.39** [98.17,98.60]	81.54 [80.57,82.51]	95.86 [95.69,96.49]	98.88 [98.64,99.12]	63.78 [62.55,65.02]	85.78 [85.41,86.14]	90.99 [90.74,91.25]	89.89 [89.67,90.15]	96.33 [96.21,96.44]	98.31 [98.23,98.38]
Inverting	81.60 [79.29,83.90]	94.69 [94.33,95.06]	97.66 [97.42,97.91]	77.66 [76.23,77.59]	95.82 [96.30,96.83]	98.64 [98.53,98.74]	56.91 [55.17,58.64]	83.54 [82.93,84.15]	90.68 [90.41,90.95]	90.85 [90.40,91.29]	96.50 [96.39,96.61]	98.24 [98.18,98.29]
Reversing	84.02 [82.38,85.65]	94.81 [94.43,95.19]	98.30 [97.87,98.73]	86.28 [85.18,87.60]	96.67 [96.77,97.03]	98.92 [98.78,99.05]	67.80 [66.70,68.89]	85.72 [85.20,85.94]	91.49 [91.29,91.68]	89.94 [89.63,90.24]	96.25 [96.11,96.39]	98.24 [98.18,98.29]
Resampling(ours)	**88.20** [87.59,88.82]	**95.27** [95.02,95.51]	98.28 [97.79,98.76]	**87.20** [86.75,87.65]	**97.00** [97.21,97.58]	**99.35** [99.28,99.42]	**71.63** [70.54,72.71]	**86.97** [86.77,87.17]	**92.28** [92.07,92.48]	**91.73** [91.44,92.02]	**96.58** [96.51,96.64]	98.32 [98.27,98.38]

The percentages indicate the proportion of the training set, and the remaining data are used as the test set. "Supervised" means that the original samples (without augmentation) are used to train DeepConvLSTM. Except for our resampling augmentation method, the implementation of other methods is derived from this work [33]. The evaluation criterion for the above results is the mean F1-Score. The value inside the square brackets indicates a 95% confidence interval, which means that there is a 95% probability that the newly trained results fall within this interval.

TABLE IV
THE STATISTICAL ANALYSIS RESULTS OF THE RESAMPLING AUGMENTATION METHOD WITH OTHER METHODS ABOUT THE WILCOXON SIGNED-RANK TEST ON SUPERVISED TASKS

	UCI-HAR			MotionSense			USC-HAD			MobiAct		
	1%	10%	60%	1%	10%	60%	1%	10%	60%	1%	10%	60%
Supervised	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	+
Noise	s+	+	+	s+	+	s+	s+	s+	s+	s+	+	-
Rotated	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Scaling	s+	+	+	s+	+	s+	s+	s+	s+	s+	s+	+
Magnify	s+	+	-	s+	s+	s+	s+	s+	s+	s+	s+	+
Inverting	s+	+	s+	s+	s+	s+	s+	s+	s+	s+	+	s+
Reversing	s+	s+	-	+	+	s+	s+	s+	s+	s+	s+	+

All non-parametric statistical hypothesis test results have been shown here at the 0.05 level of significance. These signs "+", "-", "s+", "s-" indicate that the resampling method is insignificantly better than, insignificantly worse than, significantly better than, and significantly worse than other augmentation methods, respectively.

augmentation method outperform supervised learning under 1% labeled data on the UCI-HAR, USC-HAD, and MobiAct datasets in some experimental settings. The difference between the best and the worst results is approximately 33% on USC-HAD dataset, which demonstrates that contrastive learning is very much focused on the selection of augmentation methods. With 60% labeled data, the linear evaluation of the contrastive learning model is not as good as supervised learning. This result is caused by the fact that supervised learning with a clear classification task under a large proportion of labeled data is better in classification than contrastive learning, where the task is to learn a better sample representation.

Then, we analyze the experimental results under the fine-tuning protocol. Contrastive learning frameworks using resampling data augmentation methods outperform supervised learning on the UCI-HAR, MotionSense, and USC-HAD datasets. However, they are inferior to supervised learning on the MobiAct dataset, and we speculate that this is because the sample size of the MobiAct dataset is much larger than other datasets, which is more suitable for supervised learning with a clear goal. The resampling augmentation method outperformed

TABLE V
COMPARISON OF AUGMENTATION METHODS UNDER 3 LAYER LSTM

	Training Set Proportion		
	1%	10%	60%
Supervised	74.03	92.84	95.43
Noise	75.48	94.18	95.79
Rotated	72.66	92.36	95.51
Scaling	75.68	94.30	95.92
Magnify	76.09	94.15	**95.95**
Inverting	77.44	93.94	95.67
Reversing	78.53	93.75	95.78
Resamping(ours)	**79.20**	**94.42**	95.89

The experiment used the UCI-HAR dataset and the mean F1-score as the evaluation criterion.

TABLE VI
INFERENCE TIME IN MILLISECONDS

	Sequence Length		
	64	128	200
Noise	40	68	102
Rotated	28	53	77
Scaling	36	49	73
Magnify	36	49	73
Inverting	9	13	22
Reversing	0	0	0
Resamping(ours)	128	262	384

This experiment uses the MotionSense dataset and the results are in milliseconds. Code execution on an Intel i5-10600KF CPU. One complete count is the augmentation method from receiving data to returning data.

the other methods in most experimental settings and was inferior to the other methods in a few experimental settings, but the differences in the latter were found to be small by calculating confidence intervals and non-parametric statistical analysis.

In conclusion, the resampling augmentation method achieves the best performance in 42 out of 48 experimental settings. In particular, it improves more significantly with a small amount of label data. This finding is consistent with the characteristic that contrastive learning performs well with a small amount of labeled data. In addition, the resampling augmentation method performs excellent regardless of which contrastive learning framework is used. Through the above discussion, it was demonstrated that the resampling augmentation method could indeed generate new sample data that represent the raw sample features as well as improve the performance of contrastive learning.

B. Comparison with State-of-the-art augmentation methods based on the TPN backbone network

The work [17] is the first time to apply contrastive learning to HAR, and it uses the Transformation Prediction Network (TPN) [36] as the backbone network to evaluate performance on MotionSense data. It pretrained in a modified SimCLR framework, conducted a combined augmentation study to explore the best augmentation method, and it was experimentally concluded that the best augmentation method is to use rotated for both augmentations. We used the same experiment setup according to the work [17], with the same experiment parameters, where the activity data of five random subjects were chosen for the test set and the activity data of the remaining subjects were used for the training set. The evaluation process used a five-fold cross-validation to ensure that each subject's activity data is available as a test set. The experiment results are shown in Table IX.

The experiment results show that the resampling augmentation method outperforms the previous best method (rotated) in linear evaluation. But the resampling method insignificantly outperforms the rotated methods by calculating the 95 % confidence intervals in fine-tuning evaluation. We analyze the work [17] divides a large proportion of the training set, leading to an easier training process, making the classification performance difference insignificant. In conclusion, this finding demonstrates that the resampling augmentation method can also have better results under the TPN backbone network.

C. Combinations of Augmentation Methods

To explore the performance of different combinations of augmentation methods on contrastive learning, we use SimCLRHAR and MoCoHAR as contrastive learning frameworks and use a combination of augmentation in the data augmentation phase. To compare more clearly with different combinations of augmentation methods, we do not take augmentation in the augmentation phase for the first branch but use the raw samples, and we use a specific combination of augmentation methods for the second branch. This experiment uses 1% labels as the training set for the downstream classification task, which is validated on the UCI-HAR dataset. The experiment results are shown in Fig. 4.

The experiment results show that the combination of augmentation methods under the SimCLRHAR framework did not outperform supervised learning in the linear evaluation, but the combination of resampling and rotation was close to supervised learning and outperformed the best individual augmentation of resampling by 9%. On the MoCoHAR framework, there are some combinations of augmentations that outperform supervised learning, the highest being the combination of resampling and scaling, which outperforms supervised learning by 6.36% and outperforms the best individual augmentation by 0.74%. In the fine-tuning evaluation, there are some combinations of augmentation methods under SimCLRHAR that outperform supervised learning, the best being the combination of resampling and rotation, outperforming supervised learning by 10.93% and outperforming individual augmentation by 1.05%. There are also some combinations of augmentation methods under the MoCoHAR framework that outperform supervised learning, the best being the combination of magnification and resampling, which outperforms supervised learning by 10.84% and outperforms individual augmentation by 1.37%. The combination of the above optimal augmentation methods all includes resampling, which proves the excellence of the resampling augmentation methods. It is also demonstrated that the combined augmentation is superior to the individual augmentation in the SimCLRHAR and MoCoHAR frameworks.

TABLE VII

COMPARISON WITH STATE-OF-THE-ART AUGMENTATION METHODS UNDER CONTRASTIVE LEARNING

	1%				10%				60%			
	Linear evaluation		Fine-tuned		Linear evaluation		Fine-tuned		Linear evaluation		Fine-tuned	
	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.
UCI-HAR(Sup.)	77.94 [75.38,80.50]				94.46 [94.30,94.62]				96.13 [95.78,96.48]			
Noise	54.60 [54.30,54.90]	61.63 [61.53,61.72]	81.90 [80.49,83.32]	82.02 [81.03,83.01]	77.19 [76.77,77.60]	76.30 [75.85,76.75]	94.63 [94.45,94.81]	95.02 [94.86,95.18]	81.82 [81.58,82.05]	82.63 [82.19,83.06]	97.30 [96.90,97.69]	97.25 [96.73,97.77]
Rotated	53.72 [53.26,54.18]	62.39 [62.18,62.59]	79.93 [78.85,81.00]	80.89 [80.10,81.68]	72.14 [71.76,72.52]	78.81 [78.45,79.18]	94.69 [94.60,94.78]	94.68 [94.56,94.81]	77.50 [77.15,77.84]	84.23 [83.95,84.50]	**97.35** [96.88,97.82]	96.71 [96.21,97.22]
Scaling	53.80 [53.60,53.99]	62.70 [62.47,62.93]	77.98 [76.51,79.46]	81.72 [80.48,82.96]	75.67 [75.35,75.99]	77.73 [77.49,77.99]	94.61 [94.41,94.81]	95.08 [94.79,95.36]	79.65 [79.40,79.90]	83.31 [82.91,83.71]	97.21 [96.58,97.84]	**97.49** [97.13,97.84]
Magnify	57.89 [57.68,58.09]	61.81 [61.64,61.99]	79.24 [77.55,80.93]	80.51 [79.58,81.44]	78.68 [78.38,78.97]	75.83 [75.59,76.07]	94.49 [94.33,94.66]	95.04 [94.88,95.20]	81.76 [81.45,82.07]	81.61 [81.22,82.01]	96.57 [96.14,96.99]	96.79 [96.32,97.24]
Inverting	57.70 [57.39,58.02]	63.81 [63.58,64.04]	73.97 [73.07,74.88]	81.11 [80.20,82.02]	67.04 [66.76,67.31]	76.23 [75.98,76.49]	94.37 [94.07,94.67]	94.77 [94.61,94.93]	70.34 [70.18,70.50]	81.75 [81.43,82.06]	96.40 [95.88,96.92]	96.24 [95.96,96.52]
Reversing	63.60 [63.26,63.93]	75.30 [74.94,75.65]	84.37 [82.87,84.91]	83.54 [82.06,85.02]	75.49 [75.22,75.77]	88.31 [88.01,88.60]	95.13 [94.96,95.30]	95.29 [95.09,95.50]	78.60 [78.36,78.85]	90.95 [90.77,91.14]	96.85 [96.53,97.16]	97.28 [96.77,97.78]
Resampling(ours)	**67.87** [67.47,68.26]	**83.56** [83.36,83.76]	**87.82** [87.36,88.28]	**87.41** [86.65,88.17]	**80.21** [79.93,80.50]	**91.89** [91.73,92.05]	**95.26** [95.07,95.45]	**95.49** [95.33,95.65]	**85.24** [84.87,85.62]	**93.95** [93.79,94.11]	96.78 [96.43,97.12]	96.86 [96.35,97.36]
MotionSense(Sup.)	81.13 [79.02,83.24]				96.23 [96.01,96.46]				98.62 [98.49,98.74]			
Noise	51.70 [51.59,51.80]	61.86 [61.78,61.95]	78.09 [77.08,79.10]	80.92 [80.17,81.67]	73.80 [73.55,74.06]	73.36 [72.99,73.72]	95.54 [94.71,96.37]	96.24 [95.97,96.51]	78.66 [78.35,78.96]	78.60 [78.42,78.78]	98.62 [98.31,98.93]	98.98 [98.89,99.08]
Rotated	48.67 [48.47,48.88]	63.27 [63.21,63.34]	78.67 [76.76,80.59]	81.77 [80.98,82.55]	72.53 [72.30,72.77]	76.13 [75.67,76.58]	95.76 [95.54,95.98]	96.16 [95.85,96.47]	76.55 [76.29,76.80]	81.66 [81.51,81.81]	98.59 [98.49,98.68]	98.90 [98.78,99.03]
Scaling	48.51 [48.41,48.60]	60.23 [60.15,60.32]	75.91 [74.27,77.56]	81.10 [79.82,82.37]	70.16 [69.94,70.38]	73.36 [73.05,73.67]	95.37 [94.74,96.01]	96.37 [96.10,96.65]	75.35 [75.02,75.68]	78.22 [77.98,78.46]	98.47 [98.32,98.62]	98.87 [98.64,99.09]
Magnify	56.25 [56.00,56.50]	61.76 [61.70,61.81]	77.66 [75.99,79.32]	79.73 [79.17,80.29]	69.13 [68.90,69.36]	71.61 [71.29,71.92]	95.78 [95.50,96.06]	**96.48** [96.25,96.71]	75.43 [75.06,75.79]	78.91 [78.76,79.07]	98.64 [98.53,98.74]	98.95 [98.73,99.17]
Inverting	54.09 [54.04,54.15]	62.23 [62.16,62.31]	74.79 [72.86,76.71]	81.74 [80.84,82.63]	67.73 [67.54,67.92]	72.68 [72.14,73.22]	95.27 [94.91,95.63]	95.80 [95.49,96.11]	70.67 [70.44,70.90]	77.26 [76.97,77.54]	98.52 [98.19,98.85]	98.80 [98.68,98.91]
Reversing	66.99 [66.88,67.11]	77.52 [77.42,77.62]	83.17 [81.46,84.87]	84.89 [83.83,85.94]	77.58 [77.42,77.74]	85.24 [85.06,85.43]	96.77 [96.58,96.96]	95.39 [95.03,95.75]	81.20 [80.86,81.53]	88.02 [87.82,88.22]	98.94 [98.84,99.04]	98.77 [98.67,98.86]
Resampling(ours)	**76.45** [76.32,76.57]	**77.66** [77.56,77.74]	**86.04** [84.41,87.66]	**85.48** [84.60,86.36]	**85.84** [85.64,86.06]	**91.55** [91.40,91.69]	**97.05** [96.95,97.15]	96.32 [96.05,96.59]	**89.43** [89.19,89.66]	**93.64** [93.46,93.81]	**99.11** [99.04,99.18]	**99.09** [99.00,99.18]
USC-HAD(Sup.)	63.93 [62.28,65.58]				85.58 [85.22,85.93]				90.94 [90.78,91.10]			
Noise	36.60 [35.93,37.27]	39.33 [38.54,40.13]	46.53 [44.94,48.12]	61.17 [59.47,62.87]	51.39 [50.84,51.39]	50.47 [49.84,51.10]	77.87 [77.04,78.70]	85.66 [85.36,85.97]	56.54 [56.36,56.71]	56.38 [56.11,56.65]	88.83 [88.67,88.99]	91.20 [90.83,91.20]
Rotated	28.50 [27.77,29.22]	42.16 [41.04,43.27]	44.58 [42.23,46.93]	56.83 [54.73,58.94]	38.90 [38.60,39.21]	55.02 [55.02,55.58]	83.58 [83.11,84.05]	84.43 [84.10,84.76]	43.26 [42.97,43.56]	60.65 [60.27,61.04]	89.85 [89.65,90.05]	90.52 [90.27,90.77]
Scaling	22.93 [22.40,23.46]	37.11 [36.40,37.81]	32.16 [30.81,33.50]	58.27 [56.19,60.36]	30.66 [30.44,30.89]	47.61 [47.12,48.11]	55.70 [53.70,57.71]	84.82 [84.53,85.12]	34.83 [34.52,35.15]	53.53 [53.32,53.75]	84.34 [83.83,84.86]	90.89 [90.74,91.05]
Magnify	20.21 [19.68,20.73]	36.08 [35.15,37.01]	36.67 [35.20,38.14]	58.59 [56.91,60.26]	27.77 [27.48,28.05]	46.91 [46.08,47.74]	64.60 [61.87,67.33]	86.08 [85.74,86.42]	31.65 [31.36,31.93]	54.90 [54.61,55.18]	87.02 [86.69,87.36]	91.11 [90.87,91.35]
Inverting	26.51 [26.00,27.00]	38.89 [38.04,39.74]	38.74 [36.90,40.58]	59.89 [58.37,61.41]	35.44 [35.05,35.83]	49.24 [48.87,49.60]	81.81 [81.27,82.35]	85.37 [85.10,85.63]	39.48 [39.15,39.82]	55.60 [55.38,55.83]	89.79 [89.55,90.03]	91.05 [90.88,91.21]
Reversing	45.94 [44.85,47.02]	59.92 [58.78,61.05]	57.98 [56.87,59.09]	65.83 [64.27,67.39]	58.11 [57.75,58.47]	72.14 [71.81,72.47]	82.94 [82.43,83.44]	83.84 [83.39,84.30]	62.79 [62.54,63.04]	76.80 [76.66,76.95]	90.29 [90.11,90.46]	90.53 [90.27,90.79]
Resampling(ours)	**69.69** [68.90,70.49]	**68.60** [67.66,69.55]	**70.97** [70.14,71.80]	**69.20** [67.87,70.52]	**84.84** [84.61,85.07]	**78.51** [78.21,78.81]	**87.58** [87.34,87.81]	**85.84** [85.65,86.02]	**88.66** [88.49,88.83]	**82.07** [81.82,82.32]	**92.29** [92.10,92.47]	**91.32** [91.12,91.51]
MobiAct(Sup.)	89.76 [89.44,90.07]				96.41 [96.34,96.48]				98.32 [98.28,98.36]			
Noise	61.48 [61.30,61.65]	63.19 [62.98,63.39]	71.32 [70.25,72.39]	85.22 [84.87,85.57]	64.40 [64.31,64.48]	67.84 [67.79,67.89]	90.36 [90.08,90.65]	94.89 [94.81,94.97]	65.19 [65.07,65.30]	68.50 [68.42,68.59]	96.50 [96.31,96.69]	97.95 [97.88,98.02]
Rotated	56.84 [56.66,57.03]	78.66 [78.56,78.75]	65.12 [64.08,66.16]	**88.53** [88.17,88.88]	60.31 [60.24,60.38]	82.39 [82.34,82.44]	88.27 [87.78,88.75]	94.69 [94.50,94.88]	60.99 [60.90,61.08]	83.31 [83.57,83.34]	96.02 [95.93,96.11]	97.17 [97.11,97.23]
Scaling	58.26 [58.16,58.37]	63.38 [63.26,63.51]	61.80 [61.24,62.37]	77.87 [76.95,78.80]	58.87 [58.77,58.96]	67.25 [67.20,67.31]	68.26 [66.43,70.09]	93.11 [92.94,93.29]	59.32 [59.22,59.42]	68.01 [67.95,68.07]	81.16 [80.23,82.09]	97.40 [97.32,97.48]
Magnify	52.77 [52.77,52.78]	64.88 [64.67,65.09]	62.73 [62.64,62.81]	85.13 [84.25,86.00]	52.77 [52.75,52.79]	69.61 [69.53,69.69]	64.77 [64.74,64.79]	**95.52** [95.36,95.67]	59.81 [59.77,59.85]	71.20 [71.13,71.27]	81.80 [81.77,81.85]	**97.96** [97.89,98.02]
Inverting	67.39 [67.23,67.54]	69.12 [68.94,69.29]	77.35 [76.48,78.21]	86.51 [86.03,86.98]	69.69 [69.62,69.76]	73.27 [73.23,73.32]	89.10 [88.40,89.81]	94.99 [94.85,95.14]	70.16 [70.04,70.28]	74.05 [73.98,74.11]	96.12 [95.95,96.29]	97.87 [97.79,97.94]
Reversing	64.37 [64.22,64.53]	78.40 [78.16,78.64]	77.86 [77.46,78.27]	86.43 [86.17,86.68]	67.70 [67.65,67.76]	82.75 [82.69,82.97]	89.88 [89.64,90.11]	94.24 [94.14,94.35]	68.79 [68.73,68.85]	83.64 [83.60,83.68]	95.95 [95.88,96.02]	97.50 [97.43,97.57]
Resampling(ours)	**91.59** [94.62,94.68]	**79.67** [79.50,79.85]	**88.98** [88.66,89.30]	87.20 [86.75,87.64]	**94.08** [94.04,94.12]	**83.38** [83.35,83.41]	**95.38** [95.28,95.48]	95.45 [95.30,95.60]	**94.65** [94.61,94.68]	**84.24** [84.19,84.30]	**97.98** [97.94,98.02]	97.71 [97.64,97.78]

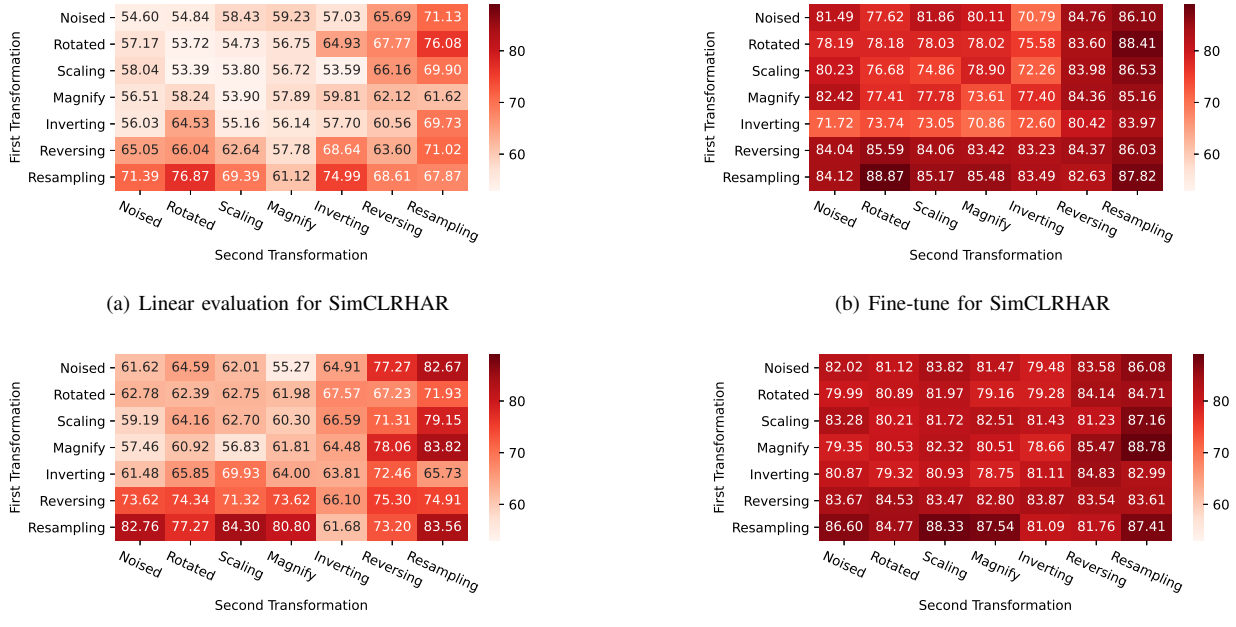
The percentages in the table header indicate the proportion of the training set in the downstream task, and the remaining data are used as the test set. Linear evaluation and fine-tuned correspond to the experimental setup of Section IV-D. "SimCLR." and "MoCo." denote the two contrastive learning frameworks, SimCLRHAR and MoCoHAR, respectively. "Sup" indicates the performance of the backbone network under supervised learning. The evaluation criterion for the above results is the mean F1-Score. The value inside the square brackets indicates a 95% confidence interval. Except for our resampling augmentation method, the implementation of other methods is derived from this work [33].

TABLE VIII

THE STATISTICAL ANALYSIS RESULTS OF THE RESAMPLING AUGMENTATION METHOD WITH OTHER METHODS ABOUT THE WILCOXON SIGNED-RANK TEST ON CONTRASTIVE LEARNING TASKS

	1%				10%				60%			
	Linear evaluation		Fine-tuned		Linear evaluation		Fine-tuned		Linear evaluation		Fine-tuned	
	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.	SimCLR.	MoCo.
UCI-HAR(Sup.)	s-	s+	s+	s+	s-	s-	s+	s+	s-	s-	s+	s+
Noise	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s-	s-
Rotated	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	-	+
Scaling	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	+	s-
Magnify	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	-	+
Inverting	s+	s+	s+	s+	s+	s+	+	+	s+	s+	+	s+
Reversing	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	-	-
MotionSense(Sup.)	s-	s-	s+	s+	s-	s-	s+	+	s-	s-	s+	s+
Noise	s+	s+	s+	s+	s+	s+	s+	+	s+	s+	s+	+
Rotated	s+	s+	s+	s+	s+	s+	s+	+	s+	s+	s+	s+
Scaling	s+	s+	s+	s+	s+	s+	s+	-	s+	s+	s+	+
Magnify	s+	s+	s+	s+	s+	s+	s+	-	s+	s+	s+	+
Inverting	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Reversing	s+	+	+	+	s+	s+	+	s+	s+	s+	s+	s+
USC-HAD(Sup.)	s+	s+	s+	s+	s-	s-	s+	+	s-	s-	s+	s+
Noise	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Rotated	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Scaling	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Magnify	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	+
Inverting	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	+
Reversing	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
MobiAct(Sup.)	s+	s-	s-	s-	s-	s-	s-	s-	s-	s-	s-	s-
Noise	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s-
Rotated	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Scaling	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+
Magnify	s+	s+	s+	s+	s+	s+	s+	-	s+	s+	s+	s-
Inverting	s+	s+	s+	+	s+	s+	s+	s+	s+	s+	s+	s-
Reversing	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+	s+

All non-parametric statistical hypothesis test results have been shown here at the 0.05 level of significance. These signs ”+”, ”-”, ”s+”, ”s-” indicate that the resampling method is insignificantly better than, insignificantly worse than, significantly better than, and significantly worse than other augmentation methods, respectively.



(a) Linear evaluation for SimCLRHAR

(b) Fine-tune for SimCLRHAR

(c) Linear evaluation for MoCoHAR

(d) Fine-tune for MoCoHAR

Fig. 4. Study on Combinatorial Augmentation Method. Diagonal Elements Use Individual Augmentation Methods.

TABLE IX
TPN BACKBONE NETWORK ON MOTIONSENSE DATASET

	Linear evaluation	Fine-tuned
Rotated [17]	85.67 [82.55,88.79]	91.41 [90.44,92.38]
Resampling (ours)	**89.50** [88.42,90.57]	**92.10** [89.36,94.84]

TABLE X
THE RESAMPLING DATA AUGMENTATION WAS EVALUATED IN NNCLR AND SIMSIAM

	NNCLR			SimSiam		
	40%	60%	80%	40%	60%	80%
Noise	86.07	79.47	69.07	76.84	72.29	67.83
Rotated	70.57	68.68	71.11	77.10	74.17	70.87
Scaling	85.13	79.04	72.36	81.72	78.83	70.04
Magnify	86.75	83.16	70.26	81.35	78.81	69.97
Inverting	83.61	79.04	72.36	79.87	78.23	70.78
Reversing	85.92	85.48	75.18	85.13	78.49	74.53
Resampling (ours)	**87.35**	**85.68**	**77.12**	**87.04**	**81.10**	**76.44**

The experiments were conducted using the UCI-HAR dataset, using the mean F1-score as the evaluation criterion. The backbone network used by the two contrastive learning frameworks is DeepConvLSTM. The percentages in the table header indicate the proportion of the test set. In the remaining data, 40% of the data is used for the validation set, and 60% of the data is used for the training set. The training epoch for both pre-training and downstream tasks is 100.

D. A Study of Batch Size

To explore the sensitivity of the contrastive learning framework to batch size, we trained two contrastive learning frameworks, SimCLRHAR and MoCoHAR, with different batch sizes in the pre-training phase and used a linear evaluation protocol to validate the classification performance after randomly selecting 1% of the labeled data as the training set and the remainder as the test set. The experiment results are shown in Fig. 5.

SimCLRHAR is more effective at smaller batch sizes, while MoCoHAR is more effective at larger batch sizes. In the field of computer vision, MoCo is more effective than SimCLR at smaller batch sizes. We suppose that the reason for the opposite result is the different sensitivity of image data and sensor data to batch size under contrastive learning.

E. Small Annotated Datasets

In this subsection, we explored the impact of limited amounts of annotated data on downstream tasks. According to the experimental setup of these works [18], [28], we randomly sampled x labeled data per class as the training set for the downstream task, where $x \in \{1, 2, 5, 10, 25, 50, 100\}$, and the remaining data as the test set. For each dataset, pre-training uses all data (without labels). We use SimCLRHAR as the contrastive learning framework to evaluate the impact of a small amount of labeled data on the downstream task on the UCI-HAR, MotionSense, USC-HAD, and MobiAct datasets.

As shown in Fig. 6, the model performs better as the number of samples increases, but the magnitude of the increase gradually becomes smaller. It is demonstrated that contrastive learning under the effect of resampling augmentation is more sensitive with a small amount of labeled data. This characteristic can alleviate the problem of limited data labeling under the large amount of unlabeled sensor data collected.

F. Other contrastive learning frameworks

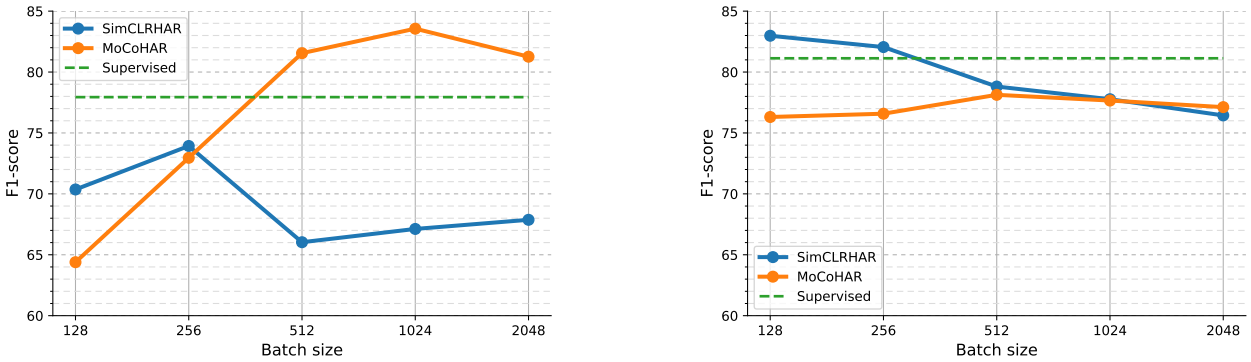
In this work [45], we found two contrastive learning frameworks, NNCLR [46] and SimSiam [47], applied to human activity recognition. The resampling augmentation method will be evaluated in the above two frameworks to demonstrate that it can be applied to multiple contrastive learning frameworks. We refer to the experimental setup of work [45] and use its released code (replacing only the augmentation methods) for our experiments. The experimental results are presented in Table X.

The experimental results found that the resampling augmentation method significantly outperforms other augmentation methods in various experimental settings. The experiments in this subsection and Section VI-A demonstrate that the resampling augmentation method can work in various contrastive learning frameworks.

VII. CONCLUSIONS

In this paper, we propose a resampling sensor data augmentation method to alleviate data labeling difficulties and address the poor performance of data augmentation in contrastive learning. SimCLRHAR and MoCoHAR as two contrastive learning environments to evaluate the resampling data augmentation. We conducted experiments on UCI-HAR, MotionSense, USC-HAD, and MobiAct, using multiple proportions of labeled data as the training set. The experiment results show that the resampling augmentation method outperforms all SOTA augmentation methods in both supervised learning and contrastive learning under a small amount of labeled data. In linear evaluation, with 1% labeled data, the resampling augmentation method was improves significantly on SimCLRHAR and MoCoHAR, outperforming the best method by 4.27% and 8.26% on the UCI-HAR, 9.46% and 0.14% on the MotionSense, 23.75% and 8.68% on the UCI-HAR, 4.2% and 1.01% on the MobiAct, respectively. MoCoHAR performs well relative to SimCLRHAR in linear evaluation protocols and larger batch size environments. Finally, we also studied the performance of batch size and combined augmentation on contrastive learning.

In addition, there are some shortcomings in this paper. The performance improvement of this resampling augmentation method is insignificant with a large amount of labeled data. Also, we found that the inference speed of the resampling augmentation method is longer compared to other methods. We will further improve the performance and inference speed of the resampling augmentation method in the future. We will also start with real-life applications of contrastive learning. For example, we use contrastive learning to analyze and train models on unlabeled data collected in any environment, so that the models in the new environment can perform well in activity recognition with only a small amount of labeled data fine-tuning.



(a) Linear evaluation in the UCI-HAR dataset, using 1% labeled data

(b) Linear evaluation in the MotionSense dataset, using 1% labeled data

Fig. 5. Impact of Batch Size on Contrastive Learning Framework (Mean F1-score). The supervised curve does not participate in the change of batch size.

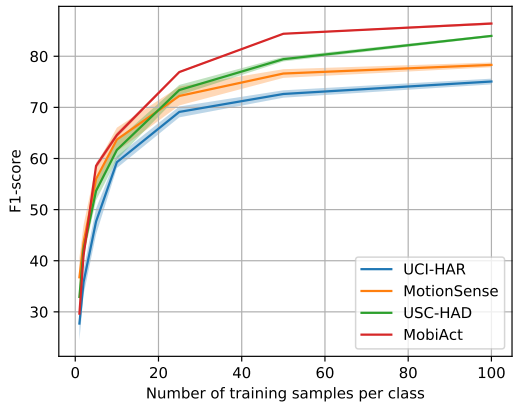


Fig. 6. Results of downstream tasks with a small amount of labeled data. All results were trained 10 times and the average and confidence interval were calculated. Shaded areas are 95% confidence intervals.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (61872038, 62006110).

REFERENCES

[1] H. Ghayvat, J. Liu, S. C. Mukhopadhyay, and X. Gui, "Wellness sensor networks: A proposal and implementation for smart home for assisted living," *IEEE Sensors Journal*, vol. 15, no. 12, pp. 7341–7348, 2015. I

[2] I. Herrera-Luna, E. J. Rechy-Ramirez, H. V. Rios-Figueroa, and A. Marin-Hernandez, "Sensor fusion used in applications for hand rehabilitation: A systematic review," *IEEE Sensors Journal*, vol. 19, no. 10, pp. 3581–3592, 2019. I

[3] 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

[4] S. Guo, Y. Wang, Y. Zhao, J. Cui, Y. Ma, G. Mao, and S. Hong, "A surgeon's operating skills-based non-interference operation detection method for novel vascular interventional surgery robot systems," *IEEE Sensors Journal*, vol. 20, no. 7, pp. 3879–3891, 2019. I

[5] 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

[6] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE communications surveys & tutorials*, vol. 15, no. 3, pp. 1192–1209, 2012. I

[7] S. Bai, M. Yan, Q. Wan, L. He, X. Wang, and J. Li, "Dl-rnn: An accurate indoor localization method via double rnns," *IEEE Sensors Journal*, vol. 20, no. 1, pp. 286–295, 2019. I

[8] N. Tufek, M. Yalcin, M. Altintas, F. Kalaoglu, Y. Li, and S. K. Bahadir, "Human action recognition using deep learning methods on limited sensory data," *IEEE Sensors Journal*, vol. 20, no. 6, pp. 3101–3112, 2019. I, V-D

[9] Z. Ahmad and N. Khan, "Cnn-based multistage gated average fusion (mgaf) for human action recognition using depth and inertial sensors," *IEEE Sensors Journal*, vol. 21, no. 3, pp. 3623–3634, 2020. I

[10] 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

[11] V. S. Murahari and T. Plötz, "On attention models for human activity recognition," in *Proceedings of the 2018 ACM international symposium on wearable computers*, 2018, pp. 100–103. I

[12] 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

[13] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, 2020. I

[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, 2021. I, II-A

[15] W. Falcon and K. Cho. (2020) A framework for contrastive self-supervised learning and designing a new approach. [Online]. Available: https://arxiv.org/abs/2009.00104 I

[16] P. Bachman, R. D. Hjelm, and W. Buchwalter. (2019) Learning representations by maximizing mutual information across views. [Online]. Available: https://arxiv.org/abs/1906.00910 I, II-B, III

[17] 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, I, II-A, II-B, III, III-B.1, IV-B, VI-B, IX

[18] 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-A, VI-E

[19] D. Anguita, A. Ghio, L. Oneto, X. Parra, J. L. Reyes-Ortiz *et al.*, "A public domain dataset for human activity recognition using smartphones." in *Esann*, vol. 3, 2013, p. 3. I, IV-A

[20] 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

[21] 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

[22] C. Chatzaki, M. Pediaditis, G. Vavoulas, and M. Tsiknakis, "Human daily activity and fall recognition using a smartphone's acceleration

sensor," in *International Conference on Information and Communication Technologies for Ageing Well and e-Health*. Springer, 2016, pp. 100–118. I, IV-A

[23] 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, II-A, III-B.1

[24] 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

[25] X. Chen, H. Fan, R. Girshick, and K. He. (2020) Improved baselines with momentum contrastive learning. [Online]. Available: https://arxiv.org/abs/2003.04297 I

[26] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar *et al.* (2020) Bootstrap your own latent: A new approach to self-supervised learning. [Online]. Available: https://arxiv.org/abs/2006.07733 II-A

[27] A. Saeed, F. D. Salim, T. Ozcelebi, and J. Lukkien, "Federated self-supervised learning of multisensor representations for embedded intelligence," *IEEE Internet of Things Journal*, vol. 8, no. 2, pp. 1030–1040, 2020. II-A

[28] 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-A, VI-E

[29] A. v. d. Oord, Y. Li, and O. Vinyals. (2018) Representation learning with contrastive predictive coding. [Online]. Available: https://arxiv.org/abs/1807.03748 II-A, III-B.2

[30] 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. II-A, III-B

[31] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in neural information processing systems*, 2016, pp. 1857–1865. II-A, III-B.1

[32] D. Liu and T. Abdelzaher, "Semi-supervised contrastive learning for human activity recognition," in *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2021, pp. 45–53. II-A

[33] T. T. Um, F. M. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, 2017, pp. 216–220. II-B, III, VII

[34] N. Dawar, S. Ostadabbas, and N. Kehtarnavaz, "Data augmentation in deep learning-based fusion of depth and inertial sensing for action recognition," *IEEE Sensors Letters*, vol. 3, no. 1, pp. 1–4, 2018. II-B

[35] K. M. Rashid and J. Louis, "Times-series data augmentation and deep learning for construction equipment activity recognition," *Advanced Engineering Informatics*, vol. 42, p. 100944, 2019. II-B

[36] 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. IV-B, VI-B

[37] H. Qian, S. J. Pan, and C. Miao, "Weakly-supervised sensor-based activity segmentation and recognition via learning from distributions," *Artificial Intelligence*, vol. 292, p. 103429, 2021. IV-C, IV-D

[38] Y. Tang, Q. Teng, L. Zhang, F. Min, and J. He, "Layer-wise training convolutional neural networks with smaller filters for human activity recognition using wearable sensors," *IEEE Sensors Journal*, vol. 21, no. 1, pp. 581–592, 2020. IV-C, IV-D

[39] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.* (2016) Tensorflow: Large-scale machine learning on heterogeneous distributed systems. [Online]. Available: https://arxiv.org/abs/1603.04467 IV-C

[40] D. P. Kingma and J. Ba. (2014) Adam: A method for stochastic optimization. [Online]. Available: https://arxiv.org/abs/1412.6980 IV-C

[41] D. M. Powers. (2020) Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. [Online]. Available: https://arxiv.org/abs/2010.16061 IV-C

[42] E. Waring, "Vii. problems concerning interpolations," *Philosophical transactions of the royal society of London*, no. 69, pp. 59–67, 1779. V-B

[43] S. McKinley and M. Levine, "Cubic spline interpolation," *College of the Redwoods*, vol. 45, no. 1, pp. 1049–1060, 1998. V-B

[44] R. F. Woolson, "Wilcoxon signed-rank test," *Wiley encyclopedia of clinical trials*, pp. 1–3, 2007. V-C

[45] 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. VI-F

[46] 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. VI-F

[47] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758. VI-F

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.

Jingyuan Gan is currently a B.E. student in the School of Computer Science, University of South China. Her research interests include intelligent perception and pattern recognition.

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.

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