Channel-Equalization-HAR: A Light-weight Convolutional Neural Network for Wearable Sensor Based Human Activity Recognition

Wenbo Huang, Lei Zhang✉, Hao Wu, Fuhong Min and Aiguo Song, *Senior Member, IEEE*

Abstract—Recently, human activity recognition (HAR) that uses wearable sensors has become a research hotspot due to its wide applications in a large variety of real-world scenarios including fitness, health-care, and sports tracking. In essence, HAR can be treated as multi-channel time series classification problem, where different channels may come from heterogeneous sensor modalities. Deep learning, especially convolutional neural networks (CNNs) have made major breakthroughs in ubiquitous HAR computing scenario. Various normalization methods have played an indispensable role in prior HAR works, which enable every layer of the network to do learning more independently by normalizing hybrid sensor features. However, normalization tends to produce a 'channel collapse' phenomenon, where a large fraction of channels only generates very small values. Most channels are inhibited and contribute very little to activity recognition. As a result, the network has to rely on only a few valid channels, which inevitably impair the generality ability of a network. In this paper, we provide an alternative called Channel Equalization to reactivate these inhibited channels by performing whitening or decorrelation operation, which compels all channels to contribute more or less to feature representation. Experiments conducted on several benchmarks including UCI-HAR, OPPORTUNITY, UniMiB-SHAR, WISDM, PAMAP2, and USC-HAD show that the proposed Channel Equalization module is an impressive alternative of convolution layers, and achieve higher recognition performance to baseline models with simliar computational cost, which significantly surpasses recent state-of-the-arts for activity recongition. To the best of our knowledge, the Channel Equalization is for the first time to be applied in multi-modal HAR scenario. Finally, the actual operation is evaluated on an embedded Raspberry Pi Model 3 B plus platform.

Index Terms—Sensor, convolutional neural network, human activity recognition, deep learning, channel equalization

1 Introduction

1.1 Motivation

During the past decade, wearable based human activity recognition (*HAR*) has become a research hot-spot due to rapid development of Internet-of-Things and sensor technology [1, 2]. Complex physical activities can be analyzed by hybrid sensor modalities such as accelerometer and gyroscope, which have been widely adopted in a large variety of real-world application [3, 4], *e.g.*, fitness, health-care and sports tracking. Shallow machine learning (*ML*) algorithms, *e.g.*, *random forest*, *SVM* and *KNN* have gained competitive results in *HAR* scenario. However, they are often constrained by complex feature engineering that relies on specific expert knowledge. Recently, due to increased computing power, deep learning, especially convolutional neural networks (*CNNs*) [5] have made major breakthroughs in ubiquitous *HAR* computing area, which could automatically extract high-level features from raw sensor time series.

- *Wenbo Huang (E-mail: wenbohuang1002@outlook.com) , Lei Zhang (E-mail: leizhang@njnu.edu.cn) and Fuhong Min (E-mail: minfuhong@njnu.edu.cn) are with the School of Electrical and Automation Engineering, Nanjing Normal University, Nanjing, 210023, China.*
- *Hao Wu (E-mail: haowu@ynu.edu.cn) is with the School of Information Science and Engineering, Yunnan University, Kunming, 650500, China.*
- *Aiguo Song (E-mail: a.g.song@seu.edu.cn) is with Department of Instrument Science and Engineering, Southeast University, Nanjing, 210096, China.*
- *Corresponding author: Lei Zhang*

At the first stage of a standard activity recognition chain [3, 6], raw data is always acquired using several sensors attached to different locations on human body. Fig.1 presents an entire pipeline of *HAR* that uses wearable sensors. Some sensors can provide multiple values (*e.g.*, an accelerometer sensor provides a *3-dimensional* acceleration signal typically referred to as x, y, and z direction), and sometimes multiple sensors are jointly sampled. Each of the sensors is sampled at regular intervals, which results in a multivariate time series. Therefore, *HAR* can be seen as multi-channel time series classification problem, where different channels may come from different sensor modalities. Using features clearly separate between activities is crucial. *CNN* is a favorable deep learning architecture for automatic feature extraction. Using a sliding window for segmenting the streaming data, the convolution operations with small kernels are directly applied along the temporal dimension of sensor signals so local dependencies between multi-channel sensor data can be captured. As is universally known, *CNNs* are typically stacked by multiple convolutional layers, each of which is followed by normalization layer such as batch normalization (*BN*) [5, 7]. From low to high, *CNNs* can automatically extract discriminative activity features in a hierarchical way. Normalization methods can enable every layer of the network to do learning more independently by normalizing input features, which plays an indispensable role in a wide range of *HAR* tasks. They always implement with rectified linear activation function such as rectified linear unit (*ReLU*) [8, 9] together, which are most mainstream building block for *HAR*.

However, recent researches indicate normalization layer tends to inhibit most channels during learning stage. For example,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3174816, IEEE Transactions on Mobile Computing

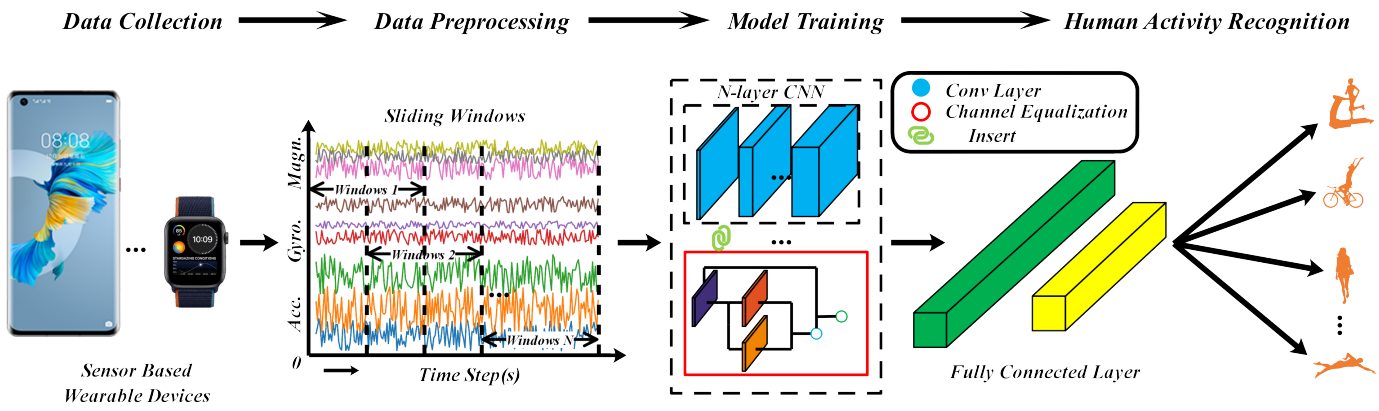


Fig. 1: The overview of network with *Channel Equalization*. The sensor time series is represented by curves.

[10, 11] shows that an over-parameterized *CNN* often contains a good number of inhibited channels, which are nearly close to zero values. Without loss of generality, this paper treats those channels with magnitudes smaller than $1e^{-3}$ as inhibited channels. As shown in Fig.2, it can be seen that a large proportion of channels only generate very small values and become inhibited channels during learning stage on several *HAR* benchmarks. This phenomenon is called as 'channel collapse' [11, 12], where a large proportion of channels only generate very small values and become inhibited channels. These inhibited channels could not learn useful feature representation and make nearly no contribution to activity recognition, which compels the network to rely on only a few valid channels. Actually, all channels should more or less take part in activity inference, and inhibiting most channels without considering their corresponding contributions will inevitably impair the generality ability of a network. To the best of our knowledge, how to handle this 'channel collapse' problem has been rarely explored in multi-modal *HAR* scenario.

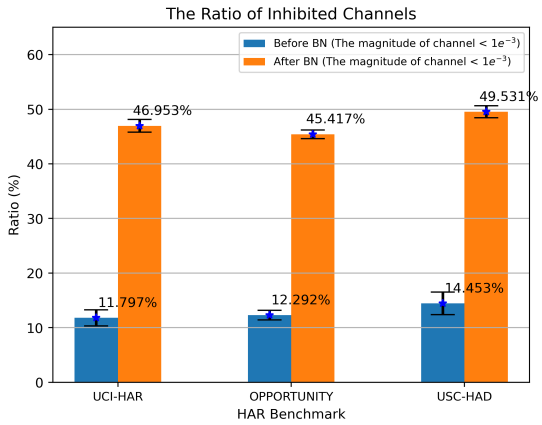


Fig. 2: The ratio (%) of inhibited channels before and after *BN*

1.2 Limitation of Prior Art

Several prior approaches have been proposed to handle the above 'channel collapse' problem. An intuitive strategy is channel pruning or network slimming [13, 14], which simply removes invalid channels after measuring the importance level of channels. For example, those inhibited channels could be easy to detected and pruned by imposing L_1 regularization on re-scaling factor in normalization unit. Although this pruning strategy can effectively decrease model size, it is somewhat ill-conceived with regards to the generalization ability to testing samples. Simply removing invalid channels inevitably impairs the generalization ability of a network.

Another important strategy is attention mechanism [6, 15], which has been extensively adopted in multi-modal *HAR* scenario. The core idea behind attention is that different sensor modalities excel in distinguishing different types of activities. For example, gyroscope is more useful in classifying activities of direction changing, while accelerometer is more meaningful in classifying activities of speed changing. Instead of treating all channels equally, attention idea tends to enhance important channels meanwhile suppressing or weakening unimportant channels. For the *Channel Equalization* case, it is used to handle 'channel collapse' problem caused by re-scaling factor within normalization layer, where these inhibited channels produce near-zero values and make nearly no contribution to activity inference. For the attention case, these enhanced or suppressed channels still more or less take part in activity inference, because these suppressed channels produce small but nonzero values. Thus, the proposed *Channel Equalization* is totally orthogonal to prior attention approaches.

1.3 Our Approach

In this paper, in order to avoid above 'channel collapse' phenomenon, we provide an alternative called *Channel Equalization* in *HAR* ubiquitous computing scenario, which aims to reactivate these inhibited channels by performing whitening or decorrelation operation over an entire mini-batch of samples, as well as each sample. The overview of the proposed method is illustrated in Fig.1. Our *Channel Equalization* method can compel all channels at the same layer more or less take part in feature learning, which prevents the network from relying on only a few specific activated channels and thus ensuring better generalization ability.

1.4 Challenges

In essence, this 'channel collapse' phenomenon can be attributed to re-scaling factor in normalization units. Our approach prevents inhibited channels with near-zero values by performing whitening operation after each normalization layer, where there are two main challenges.

The first one is how to perform the whitening or decorrelation operation over an entire mini-batch of samples. According to matrix theory, the whitening or decorrelation operation is computationally heavy, which often requires to compute an inverse square root of covariance matrix by implementing eigenvalue decomposition (*EVD*) or singular value decomposition (*SVD*). Due to limited computing power in wearable devices, the trade-off between accuracy and speed should be preferably considered. Thus, we propose an efficient Newton's Iteration to compute the inverse square root, which is able to effectively reduce computational overhead compared with the *EVD*.

The second one is how to perform the whitening or decorrelation operation over every single sample. There is an important challenge related to deep *HAR*: online or real-time activity inference. In order to satisfy this requirement, one cannot perform only decorrelation operation over an entire mini-batch. The dependencies among channels should be considered for every single sample. As a result, it needs to compute the inverse square root of variance along each channel. In particular, the inverse square root of variance should maintain the same magnitude as that of covariance within above batch decorrelation, which ensures that neither of them is dominant. Thus, we propose to use a *Sigmoid* activation to generate a set of channel weights, which may control the magnitude of the inverse square root of variance for each channel.

1.5 Contributions

To the best of our knowledge, the *Channel Equalization* is the first work that aims to equalize all channels in multi-modal HAR scenario. We perform comprehensive experiments to evaluate the effectiveness of our channel equalization method on several activity recognition benchmarks, including UCI-HAR, UniMiB-SHAR, WISDM, OPPORTUNITY, PAMAP2 and USC-HAD. The effect of decorrelating operation over an entire mini-batch and every single sample are analyzed independently, and the ratio of inhibited channels is quantitatively evaluated. We also run actual inference on a *Raspberry Pi* platform. The main contributions of this paper are summarized as follows:

- We present a new approach called channel equalization in ubiquitous *HAR* computing scenario, which compels all channels to contribute more or less to activity feature representation.
- By modifying normal normalization with whitening operation, our *Channel Equalization* method can reactive these inhibited channels instead of simply pruning them, which may ensure better generalization ability.
- The *Channel Equalization* can be readily integrated seamlessly into representative convolutional networks for *HAR*, at merely negligible computational cost.

The rest of this paper is organized as follows. Section 2 reviews recent related literature. Section 3 illustrates an overview of the proposed *Channel Equalization* in details. In Section 4, the specific experiment setup and benchmark datasets are introduced. Our main experimental results are analyzed and discussed. Finally, a conclusion is drawn in Section 5.

2 Related Works

Normalization & decorrelation. As is universally known, various normalization methods are quite beneficial for training neural networks, which have gained a lot of attention in deep learning community. For example, batch normalization (*BN*) [7] was the easiest work in this area, which performs normalization over every single mini-batch to enable training process more stable. There are two main stages, *e.g.*, standardization and re-scaling in the normalization scheme, which are usually performed after every convolutional layer. In order to handle small batch size, Ba *et al.* [16] introduced an idea of layer normalization (*LN*), which performs normalization over each example by computing the mean and variance from the same layer. Wu *et al.* [17] presented an alternative to *BN* that is called as group normalization (*GN*), where all channels are divided into groups and the mean and variance of each group are computed for normalization. However, above normalization methods only perform standardization and re-scaling. Motivated by the well-known fact that whitening inputs (*i.e.* centering, decorrelating, and scaling) can further accelerate and stable training, several researches have been devoted to the whitening or decorrelation operation. Huang *et al.* [18] proposed an idea of decorrelated batch normalization (*DBN*) by using zero-phase component analysis (*ZCA*) whitening technique, which allows successful learning. Cogswell *et al.* [19] introduced an iterative normalization method, which can whiten these activations iteratively by a Newton's method. Zhang *et al.* [20] proposed stochastic whitening batch normalization, which can compute whitening matrix dynamically during training in an online manner. Shao *et al.* [12] verified that normalization methods inevitably inhibit most raw channels, which severely impair the generality ability of a network. So far, these normalization methods are mainly investigated in visual recognition or natural language processing (*NLP*) tasks, which have been rarely explored in *HAR* scenario. This paper is the first work to modify batch normalization in *HAR* area by performing whitening over all channels, which is a direction orthogonal to all these prior works.

Deep learning in HAR. Over the past decade, deep learning has provided strong results for various activity recognition tasks, which has become a research focus in *HAR* research community. For example, Zeng *et al.* [21] at the earliest time realized *HA*R via deep *CNNs*, where a set of convolutional layers and pooling layers are stacked sequentially to automatically extract features from sensor time series for activity classification. Ignatov *et al.* [22] introduced a novel convolutional network for real-time activity recognition tasks via combining automatic feature extraction by *CNN* together with shallow hand-crafted features, which is able to effectively preserve the global form of sensor signals. Yang *et al.* [3] adopted *CNNs* to handle heterogeneous sensor modalities, where the signals from accelerometer and gyroscope need to be convolved separately for activity feature extraction. Ronao *et al.* [23] designed a deep *CNN*, which is able to automatically extract the scale invariance and hierarchical features of complex human activities. By transforming raw sensor signals into a two-dimensional activity image for classification, Jiang *et al.* [24] introduced a novel *CNN* with two layers, which

are composed of the filters of *5×5* followed by *4×4* and *2×2* average-pooling layers respectively. Huang *et al.* [25] introduced a channel-selectivity *CNN*, which allows the network to pay more attention to important channels meanwhile removing other unimportant channels. In summary, normalization methods have played an indispensable role in these representative convolutional networks, which inevitably inhibit most sensor channels and impair their generality ability. Thus, it requires deeper researches into exploring the effect of 'channel collapse' in multi-modal *HAR* scenario. In this paper, our main research motivation is to design a novel convolutional network with a modified normalization block, which compels all channels to contribute more or less to feature representation.

3 MODEL

We provide detailed descriptions of our model in this section. Specially, Section 3.1 introduces inhibited channels. Section 3.2 introduces the *Channel Equalization* block, where Section 3.2.1 and Section 3.2.2 detail its two main branches: batch decorrelation and instance reweighting [12, 26, 27] respectively. Section 3.3 analyzes the computational complexity of our frame.

3.1 Inhibited Sensor Channels

Until now, the success of batch normalization is indisputable, which has been used by default in most deep models. Due to hybrid sensor modalities, it is very crucial for *HAR* to normalize input sensor data into zero-mean and unit standard variance. The combination of normalization and rectified units $g(\cdot)$ can be formulated as follows:

$$\begin{aligned} &y_{ncij} = g\left(\tilde{x}_{ncij}\right), \\ &\tilde{x}_{ncij} = \gamma_c \bar{x}_{ncij} + \beta_c, \\ &\bar{x}_{ncij} = \left(x_{ncij} - \mu_k\right)/\sigma_k, \\ &\quad k \in \{IN, BN, ...\}, \\ &\quad n : the\ n^{th}\ sample, \\ &\quad c : the\ c^{th}\ channel, \\ &\quad i, j : location\,(i, j) \end{aligned} \tag{1}$$

Here $\bar{x}_{ncij}$ is the feature after standardization and $\tilde{x}_{ncij}$ is the feature after normalization. y_{ncij} represents final output feature after the rectified activation function and normalization. β_c and γ_c are two parameters used for re-shifting and re-scaling in normalization. σ_k and μ_k denote standard deviation and mean acquired by normalizer k respectively, where k is referred to as layer normalization (*LN*) [16], instance normalization (*IN*) [26] and batch normalization (*BN*) [7], *etc.* In this paper, we mainly investigate *BN* operation, which can be readily extended to another case.

Actually, normalization methods play an indispensable role in *HAR* tasks. *BN* is a normalization technique done between the layers of a deep network as well as in raw input data, which forces the data points from each layer to mimic a standard normal distribution with zero mean and unit variance. As also claimed by Ioffe & Szegedy (2015) [7], Arpit *et al.* (2016) [28], Teye *et al.* (2018) [29] and Shao *et al.* (2020) [12], with some weak assumptions and approximations, we approximately assume the pre-activation $\bar{x}_{ncij}$ has a Gaussian distribution. The above 'channel collapse' phenomenon can be attributed to this normalization operation. For $i \in [H]$ and $j \in [W]$, [11] indicates that y_{ncij} and γ_c become small after training:

$$\begin{aligned} &When: \\ &\quad z = \bar{x}_{ncij} \sim N(0,1), \\ &\quad y = \max\{0, \gamma_c z + \beta_c\}, \\ &\quad \mathbb{E}_z[y] = \mathbb{E}_z\left[y^2\right] = 0, \\ &If\ and\ only\ if: \\ &\quad \beta_c \leqslant 0, \\ &\quad \gamma_c \to 0 \end{aligned} \tag{2}$$

If γ_c is very small, it can be seen that the variance and mean of the output y_{ncij} are also very small. In this case, the c^{th} channel make very little contribution and turns into inhibited channel. For the convenience of notation, we denote these channels that are inhibited if their magnitudes are smaller than $1e^{-3}$. We conclude that most channels are always inhibited due to the existence of normalization and rectified units. To avoid these inhibited channels, a direct and practical idea is to perform decorrelation (whitening) over sensor channel features after normalization by using a covariance matrix, which enables the correlation between highly correlated channels to reduce to an ideal level.

3.2 Channel Equalization Block

The *Channel Equalization* block compels all channels to contribute more or less to feature representation via channel decorrelation [12] or whitening [19]. Specifically, we perform channel decorrelation after normalization, which can be formulated as:

$$\begin{aligned} &p_{nij} = D_n^{-\frac{1}{2}}\left(Diag\,(\gamma)\,\bar{x}_{nij} + \beta\right) \\ &p_{nij}, \bar{x}_{nij}, \gamma, \beta \in \mathbb{R}^{C\times 1} \end{aligned} \tag{3}$$

where p_{nij} is a *C-dimensional* vector (*e.g. C channels*) which represents the output of *Channel Equalization* block. We stack the elements $\bar{x}_{ncij}$ over all channels into a column vector $\bar{x}_{nij}$. In a similar way, we stack β_c and γ_c to get β and γ respectively. $Diag\,(\gamma)$ represent a diagonal matrix whose diagonal elements are γ. $D_n^{-\frac{1}{2}}$ denotes decorrelation or whitening operation, which compels all channels to contribute more or less to feature representation. One [12, 19, 30] usually trains model with batches and inferences with single sample (*i.e.*, online inference). Thus, it is very crucial to perform decorrelation operation over every single sample for activity recognition. The statistics of channel dependency involved in both the mini-batch and each sample should be considered. Thus, D_n is written as:

$$\begin{aligned} &D_n = \lambda\Sigma + (1-\lambda)\,Diag\,(v_n), \\ &v_n = f\left(\tilde{\sigma}_n^2\right), \\ &f : \mathbb{R}^{C\times 1} \to \mathbb{R}^{C\times 1}, \\ &v_n, \tilde{\sigma}_n^2 \in \mathbb{R}^{C\times 1}, \\ &\Sigma \in \mathbb{R}^{C\times C}, \\ &\lambda \in (0,1) \end{aligned} \tag{4}$$

where the covariance matrix Σ is computed over an entire batch of samples after normalization $\{\bar{x}_n\}_{n=1}^N$. $\tilde{\sigma}_n^2$ is a vector of variance estimated over all channels, which is used to adjust channel

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3174816, IEEE Transactions on Mobile Computing

correlations for each sample by the transformation f. Following the Jensen inequality [31], $D_n^{-\frac{1}{2}}$ can be further relaxed as:

$$\begin{aligned} D_n^{-\frac{1}{2}} &= [\lambda\Sigma + (1-\lambda)\,Diag\,(v_n)]^{-\frac{1}{2}} \\ &\preceq \lambda \underbrace{\Sigma^{-\frac{1}{2}}}_{\text{batch decorrelation}} + (1-\lambda)\underbrace{[Diag\,(v_n)]^{-\frac{1}{2}}}_{\text{instance reweighting}}, \\ &*A \preceq B \text{ means } B-A \text{ is semi-definite positive} \end{aligned} \tag{5}$$

The overview of the proposed model is detailed in Fig.3. In summary, D_n is composed of two main branches, *i.e.*, $\Sigma^{-\frac{1}{2}}$ and $[Diag\,(v_n)]^{-\frac{1}{2}}$. The former one performs batch decorrelation by using a covariance matrix estimated in an entire mini-batch, while the latter one performs instance reweighting by adjusting correlations among feature channels for each sample. We also use a learnable ratio λ to trade off batch decorrelation and instance reweighting. The above relaxation brings two additional benefits to *HAR* in wearable devices: (1) Faster inference: $D_n^{-\frac{1}{2}}$ can be computed as a moving-average statistic, which leads to a faster inference speed. (2) Lightweight computation: the inverse square root $D_n^{-\frac{1}{2}}$ only needs to be computed once during straining stage, while $Diag\,(v_n)$ is also very easy to calculate. We continue to detail batch decorrelation $D_n^{-\frac{1}{2}}$ and instance reweighting $[Diag\,(v_n)]^{-\frac{1}{2}}$.

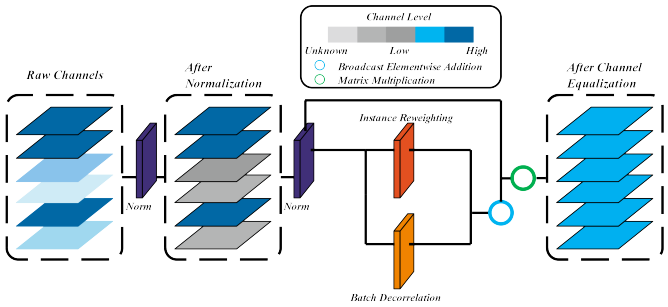


Fig. 3: Channel Equalization Block

3.2.1 Batch Decorrelation (BD) Branch

Intuitively, the covariance matrix D_n should be computed over a mini-batch of samples. Assuming that $\bar{x}$ indicates a standardized feature tensors, as denoted above, the dependency between i^{th} channel and j^{th} channel is represented by Σ_{ij}, which is scaled after normalization by $\gamma_i\gamma_j$. The covariance matrix can be formulated as:

$$\begin{aligned} &\Sigma = \gamma\gamma^T \odot \frac{1}{N\cdot H\cdot W}\bar{x}\bar{x}^T, \\ &\bar{x} \in \mathbb{R}^{C\times M}, \\ &*\odot \text{ means elementwise multiplication} \end{aligned} \tag{6}$$

Further, one usually can calculate $\Sigma^{-\frac{1}{2}}$ by performing *SVD* or eigenvalue decomposition (*EVD*) [18]. In order to avoid heavy computation, we adopt a Newton's Iteration to calculate the inverse square root during training stage, which can be easily integrated into convolutional layers at almost negligible computational burden.

3.2.2 Instance Reweighting (IR) Branch

Besides batch decorrelation, each sample [12, 26, 27] also needs to be decorrelated. Referring to [27], we can compute the input of instance reweighting by:

$$\begin{aligned} &\tilde{\sigma}_n^2 = \text{diag}\left(\gamma\gamma^T\right) \odot \frac{\left(\sigma_{IN}^2\right)_n}{\sigma_{BN}^2}, \\ &\tilde{\sigma}_n^2, \text{diag}\left(\gamma\gamma^T\right), \left(\sigma_{IN}^2\right)_n, \sigma_{BN}^2 \in \mathbb{R}^{C\times 1} \end{aligned} \tag{7}$$

where $\text{diag}\left(\gamma\gamma^T\right)$ is used to extract a given matrix's diagonal. σ_{BN}^2 and $\left(\sigma_{IN}^2\right)_n$ denote the variances estimated by *BN* and *IN* respectively. The division of vector is element-wisely applied in Eq.7. γ_c^2 for the c^{th} channel is used to scale the input of instance reweighting. The transformation $\tilde{f}$ is a *sigmoid* activation function, in a re-parameterized way, which can be formulated as follows [12]:

$$\begin{aligned} s &= \frac{1}{NC}\sum\nolimits_{n,c}^{N,C}\left(\tilde{\sigma}_n^2\right)_c, \\ [Diag\,(v_n)]^{-\frac{1}{2}} &= Diag\left(\tilde{f}\left(\tilde{\sigma}_n^2;\theta\right)\right)\cdot s^{-\frac{1}{2}} \end{aligned} \tag{8}$$

where $s^{-\frac{1}{2}}$ denotes the inverse square root of variances computed over all mini-batches and all examples. Following the guidance of Squeeze-and-excitation (*SE*) [32] block and Global Context (*GC*) [33] block, we can adjust the channel dependencies via a sub-network with the θ parameter. The *sigmoid* activation function $\tilde{f}$ can learn a set of weights to control the inverse square root of variances over all channels, which ensures that its output magnitude is close to the inverse square root of variance or covariance in batch decorrelation branch. In other words, neither of batch decorrelation and instance reweighting is dominant in *Channel Equalization*.

3.3 Computational Complexity Analysis

The computation cost of the proposed frame is composed of two parts, which include computing the inverse square root of covariance matrix in the *BD* branch and calculating two *FC* layers in the *IR* branch. For the former case, it should be noted that $\Sigma^{-\frac{1}{2}}$ remains fixed during inference, which only need to perform a linear transformation, *i.e.*, $\Sigma^{-\frac{1}{2}}\tilde{x}_{ncij}$ without other extra computational cost. For ease of discussion, omitting bias term, one can easily assume that the number of channels in the convolution is equal to that in the previous layer, *i.e.*, C. In this sense, given an input $\mathbb{R}^{N\times C\times H\times W}$ as denoted above, the *BD* branch only involves computational cost of matrix multiplication, $O\left(NHWC^2\right)$, which is comparable to the computational complexity of convolution operation. When H or W is too large, a pooling operation with kernel size k can be used to perform sub-sampling, which might further reduce the computational complexity to $O\left(NHWC^2/k^2\right)$. For the latter case, the *IR* branch focuses on the computational cost caused by the two *FC* layers. Without loss of generality, assuming that input neuron numbers are equal to output neuron numbers, the computational complexity is $O\left(NC^2\right)$.

4 Experiments

All the experiments are composed of four parts. Section 4.1 and 4.2 introduces the details of benchmark *HAR* datasets and experiment setup respectively. The quantitative comparisons are discussed in Section 4.3 and several ablation studies are analyzed in Section 4.4.

4.1 Benchmark Datasets

The several public *HAR* datasets including UCI-HAR, OPPORTUNITY, UniMiB-SHAR, WISDM, PAMAP2 and USC-HAD are used for our evaluation. The sliding window with a fixed length and overlap rate is used to divide multi-channel sensor time series into *4-dimension* input tensors with various shapes, where the data segmentation procedure is written by *Numpy* library. So far, there is still no clear consensus on what is the optimal window length. Referring to recent research literature [15, 21, 25], we utilize recent successful segmentation strategy to ensure fair comparisons. Table 1 shows the details of data preprocessing procedure. We divided each data set into training set, validation set and test set as a ratio of 7:1:2.

UCI-HAR dataset [34]. To evaluate various *ML* algorithms, the research team from University of California Irvine built this dataset from 30 volunteers between 19 and 48 years old. Each volunteer subject attached a *Samsung Galaxy S2* phone to his/her waist for data collection. All volunteers were asked to perform 6 different activities of daily living (*ADLs*) including 'walking upstairs', 'walking downstairs', 'walking', 'lying', 'standing', and 'sitting' in a supervised scenario. The sensor signals are sampled at 50 Hz by triaxial accelerometer and gyroscope.

OPPORTUNITY dataset [35]. Roggen *et al.* in University of Sussex built this dataset under a sensor-rich scenario, which is composed of 15 wireless and wired networked sensing nodes. Specifically, there are 72 sensors of 10 modalities. The subset of the OPPORTUNITY challenge including unsegmented sensor recordings is collected from 4 subjects by placing sensor nodes over 12 different body parts. Each subject was asked to perform 17 types of activities such as 'turning on/off the light', 'opening/closing the fridge', *etc*. for 20 repetitive runs in a breakfast scenario, which were sampled at 30 Hz.

UniMiB-SHAR dataset [36]. This is a new type of acceleration sensor dataset collected by Micucci *et al.* from the University of Milano-Bicocca. This dataset was designed for detecting falls and monitoring human activities. At a sample rate of 50 Hz, the researchers used Android based smartphones and collected all 11,771 samples from 30 volunteers whose ages range from 18 to 60 years old. All samples are divided into two coarse grained classes: 9 types of activities of daily living (*ADLs*) and 8 types of falls.

WISDM dataset [31]. The Wireless Sensor Data Mining (WISDM) Lab used modern mobile devices, *e.g.*, smart phones, laptop computers and music players to collect this sensor dataset. 29 volunteer subjects placed a smartphone on their front leg and collected 6 types of activities including 'walking upstairs and downstairs', 'standing', 'jogging', 'sitting', and 'walking'. The sampling rate is set to 20 Hz. The WISDM dataset includes 10,981 samples.

PAMAP2 dataset [37]. This dataset is a European project in the area of ambient assisted living, which aims to help elderly people and make their life as healthy and comfortable as possible. 9 subjects joined in the data collection process by performing 18 kinds of activities consisting of 'cycling', 'rope jumping', and 'walking', *etc*. At a sample rate of 100 Hz, each subject attached 3 Inertial Measurement Units (*IMUs*) and a heart-rate-monitor to his/her arm, ankle and chest respectively. For fair comparison, the sampling rate is down-sampled into 33 Hz. The whole dataset was made publicly available.

USC-HAD dataset [38]. The University of Southern California Human Activity Dataset (USC-HAD) is composed of low-level human activities, which is designed as a benchmark for comparing *ML* algorithms especially under healthcare scenarios. There are 12 physical activities ('lying', 'walking', 'sitting on a chair', *etc*.) collected from 14 volunteer subjects. The sampling rate is 100 Hz. Most researchers use the USC-HAD dataset for various healthcare applications such as elder care and health monitoring.

TABLE 1: Data preprocessing

Dataset	Categories	Window size	Overlap rate
UCI-HAR	6	128×9	50%
OPPORTUNITY	17	40×113	30%
UniMiB-SHAR	17	151×3	50%
WISDM	6	200×3	50%
PAMAP2	12	86×120	78%
USC-HAD	12	512×6	50%

4.2 Implementation details

In this subsection, the network architecture and training details are introduced. Following recent research literatures [15, 21, 39–41], we build a *3-layer CNN* as our baseline, which could provide very close results comparing to representative convolutional networks. A flowchart (Fig.4) is drawn to illustrate the complete network architecture. By subtracting mean and dividing by standard deviation, multi-channel time series could be first normalized into zero mean and unit standard variance. As indicated above, we insert the *Channel Equalization* block between the final normalization layer and activation function. The optimizer we choose is Adam, whose weight decay is initially set to $1e^{-4}$. For an initial learning rate $5e^{-4}$, we select exponential decay (radix: $\gamma = 0.9$) as the decay strategy of learning rate. *ReLU* is used as activation function and the number of epochs is set to 200. Different batch sizes are used according to different *HAR* datasets. A fully connected (FC) layer and Softmax are inserted after 3 convolutional layers. Other hyper-parameters are set to be exactly the same for fair comparisons. The implementation is based on Pytorch deep learning library. We conduct all experiments on a deep learning server. The *CPU* type is *Intel* 6^{th} generation *Core i7-6850K*, the *GPU* is *NVIDIA* 24GB *RTX*3090 and the *RAM* is 64GB. Table 2 illustrates the details of network architecture, where C(L_s) denotes convolutional layer with L_s feature maps. For a more comprehensive evaluation of the proposed method, four diverse evaluation metrics including precision, recall, F-measure and accuracy are utilized, which is mathematically expressed as:

$$\begin{aligned} Precision &= \frac{TP}{TP+FP}, \\ Recall &= \frac{TP}{TP+FN}, \\ F-measure &= 2 \times \frac{Precision \times Recall}{Precision + Recall}, \\ Accuracy &= \frac{TP+TN}{TP+FP+FN+TN} \end{aligned} \tag{9}$$

where TP, FN, FP, and TN represent true positives, false negatives, false positives, and true negatives respectively. Table 3 summarizes main experimental results. Code will be available at website: https://github.com/wenbohuang1002/Channel-Equalization-HAR.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3174816, IEEE Transactions on Mobile Computing

TABLE 2: Networks' architecture

Dataset	1^{st} layer	2^{nd} layer	3^{rd} layer	Batch size
UCI-HAR	C(64)	C(128)	C(256)	128
OPPORTUNITY	C(64)	C(256)	C(384)	64
UniMiB-SHAR	C(64)	C(256)	C(384)	64
WISDM	C(128)	C(256)	C(384)	128
PAMAP2	C(128)	C(256)	C(384)	128
USC-HAD	C(64)	C(128)	C(256)	64

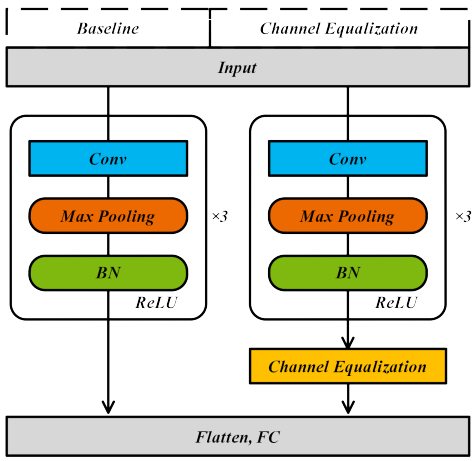


Fig. 4: Brief description for architecture

4.3 Quantitative Comparison

We perform quantitative comparisons between the baseline, our *Channel Equalization* method and other representative convolutional networks. All models are trained 5 times, and the mean accuracy is reported. Fig.5 illustrates error curves across several benchmark *HAR* datasets.

Results from Table 3, it can be clearly seen that there are significant performance improvements caused by *Channel Equalization* block over all 6 benchmark *HAR* datasets. On UCI-HAR and OPPORTUNITY, our *Channel Equalization* method surpasses their baselines by 1.12% and 2.91% respectively. As for UniMiB-SHAR and WISDM, our *Channel Equalization* method outperforms both baselines by 2.17% and 1.83%. There are 1.91% and 2.05% performance improvements over PAMAP2 and USC-HAD respectively when compared with corresponding baselines. Overall, experiments conducted on several benchmarks including UCI-HAR, OPPORTUNITY, UniMiB-SHAR, WISDM, PAMAP2, and USC-HAD demonstrate that the proposed *Channel Equalization* module is an impressive alternative of convolution layers, and achieve higher recognition performance than baseline models with similar computational cost.

Our *Channel Equalization* method is also compared with recent representative state-of-the-art approaches. As for UCI-HAR dataset, Jiang *et al.* [24] use fast fourier transform (FFT) to assemble sensor signals of accelerometers and gyroscopes into an activity image, which is used as input fed into *CNN* to automatically learn discriminative activity features. We can see that the state-of-the-art model with larger computational cost caused by FFT obtain lower accuracy (95.18%) than ours (97.35%). Our method is superior to Ronao *et al.*'s [23] method using the same *CNN* architecture without *Channel Equalization* block by 1.6%, but with similar computational cost. Our *Channel Equalization* method outperforms the *CNN* competitor [22] that ensembles statistical features with the higher performance (96.63%). Comparing to the strong baseline, our method is more lightweight, which needs no extra statistical features. In the case of OPPORTUNITY, Hu *et al.* [42] propose to use incremental *random forest* for activity recognition. However, similar to other versions of *random forest*, their method needs to store all examples all the time, which is impractical for resource-constrained mobile devices. Our *Channel Equalization* method significantly surpasses Hu *et al.*'s method by a large margin of 4.67%. Ordóñez *et al.* present a hybrid network architecture called *DeepConvLSTM* [46] that ensembles *CNN* and *LSTM* layers. Using *CNN* alone, our method is superior to *DeepConvLSTM* with an absolute gain of 2.32%. Moreover, it outperforms the best performing approach on OPPORTUNITY using bi-directional *LSTMs* (Hammerla et al. [50]) by 1.12%. When training our model on UniMiB-SHAR dataset, the proposed method is also able to provide a consistent performance improvement, beating the hybrid *CNN-LSTM* model proposed by Li *et al.* [51] by 3.99%, which ranks among the most efficient methods on UniMiB-SHAR. Moreover, our method outperforms the state-of-the-art algorithm, *i.e.*, Codebook Approach with Soft Variant [43] by 1.62%. Because a codebook needs to be constructed by applying the *k-means* clustering to one million subsequences randomly selected, this codebook approach is computationally expensive. Wang *et al.* [47] have recently proposed a recurrent attention network (*RAN*) consisting of *CNN*, *LSTM*, and attention module, which achieves an accuracy of 72.8% on this dataset. We can see that our *Channel Equalization* method using *CNN* alone obtains about 5.85% higher accuracy than *RAN*. For WISDM dataset, Zhang *et al.* [44] introduce a novel method that ensembles *CNN* and attention mechanism, where multi-head attention instead of a single attention is jointly implemented. Our *Channel Equalization* significantly surpasses Zhang *et al.*'s method by 2.64%, but with a simpler architecture (using *CNN* only). Noori *et al.* [48] propose a novel *CNN* using data-level, feature-level, and decision-level fusions, which shows promising performance with the best result reaching an overall accuracy of 98.7% on WISDM. However, they have not provided a generalized way about how to perform data-level fusion. We can also see that their method with multi-level fusion obtain lower accuracy than ours (99.04%). Khan *et al.* [52] propose an attention-based multi-head convolutional neural network using squeeze-and-excitation (*SE*) module, which has about 1.04M parameters with an accuracy of 98.18%. We further increase the accuracy to 99.04% by using *Channel Equalization*, with only 0.43M parameters. Concerning recent literatures on PAMAP2 dataset, [15] and [21] propose a combination of *CNN* and different *LSTM* architectures, where attention mechanism and multi-modal fusion are utilized to deal with sensor-based *HAR*. Our *Channel Equalization-based* method using *CNN* alone outperforms the two competitors with hybrid architectures by 2.84% and 2.18% respectively, but with lower computational cost. Khaertdinov *et al.* [53] further improve [15] by using triplet loss, while our method is still able to provide an absolute gain of 1.74%. Finally, together with *CNN* and *LSTM*, Singh *et al.* [54] show that a self-attention module that weighs the embedding of raw sensor input provides a significant improvement in accuracy (94.06%) on USC-HAD dataset. Our *Channel Equalization* method significantly surpasses the self-attention mechanism by 5.11% according to evaluation metric. Our method is also superior to Bi *et al.*'s method [49] using dynamic active learning by 7.47%. Due to the clustering technique used, their approach needs to store all samples, which will inevitably increase memory cost, subsequently limiting its widespread use for sensor-based activity recognition. Li *et al.* [45] have recently proposed a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3174816, IEEE Transactions on Mobile Computing

TABLE 3: Test Results on Benchmark Datasets

Benchmark Datasets		UCI-HAR	OPPORTUNITY	UniMiB-SHAR	WISDM	PAMAP2	USC-HAD
Baseline	Accuracy† (%)	96.23	90.01	76.48	97.21	90.23	97.21
	Precision≀ (%)	95.96	89.93	76.57	97.33	90.14	96.98
	Recall§ (%)	96.16	90.25	76.66	97.22	90.36	96.97
	F_1 Score⋆ (%)	96.05	90.1	76.61	97.27	90.24	96.97
	Parameters (M)	0.4	3.37	0.44	0.4	1.55	0.52
	Flops (M)	43.8	520.67	25.84	34.84	488.62	116.87
Ours	Accuracy† (%)	**97.35 (↑1.12)**	**93.82 (↑2.91)**	**78.65 (↑2.17)**	**99.04 (↑1.83)**	**92.14 (↑1.91)**	**99.17 (↑2.05)**
	Precision≀ (%)	**96.93 (↑0.97)**	**93.33 (↑3.4)**	**78.69 (↑2.12)**	**99.12 (↑1.79)**	**91.94 (↑1.8)**	**98.85 (↑1.87)**
	Recall§ (%)	**97.33 (↑1.17)**	**94.04 (↑3.79)**	**78.88 (↑2.22)**	**99.26 (↑2.04)**	**92.43 (↑2.07)**	**99.02 (↑2.05)**
	F_1 Score⋆ (%)	**97.12 (↑1.07)**	**93.68 (↑3.58)**	**78.78 (↑2.17)**	**99.18 (↑1.91)**	**92.18 (↑1.94)**	**98.93 (↑1.96)**
	Parameters (M)	0.43 (↑0.03)	3.51 (↑0.14)	0.47 (↑0.03)	0.43 (↑0.03)	1.68 (↑0.13)	0.55 (↑0.03)
	Flops (M)	43.82 (↑0.02)	520.76 (↑0.09)	25.86 (↑0.02)	34.86 (↑0.02)	488.69 (↑0.07)	116.88 (↑0.01)
Other Researchers		95.75 † [23] 95.18 † [24] 96.63 † [22]	89.15 † [42] 91.5 † [46] 92.7 ⋆ [50]	77.03 † [43] 72.8 † [47] 74.66 † [51]	96.4 † [44] 98.7 † [48] 98.18 † [52]	89.3 ⋆ [15] 89.96 ⋆ [21] 90.4 ⋆ [53]	93.79 † [45] 91.7 † [49] 94.06 † [54]

†: Test Accuracy. ≀: Test Precision. §: Test Recall. ⋆: Test F_1 Score.

Meta-HAR approach using deep federated representation learning for activity recognition, where their embedding network leverages both *CNN* and *RNN* to extract features from sensor readings. Our method using *CNN* alone significantly outperforms the *Meta-HAR* by 5.38%, but with lower computational cost.

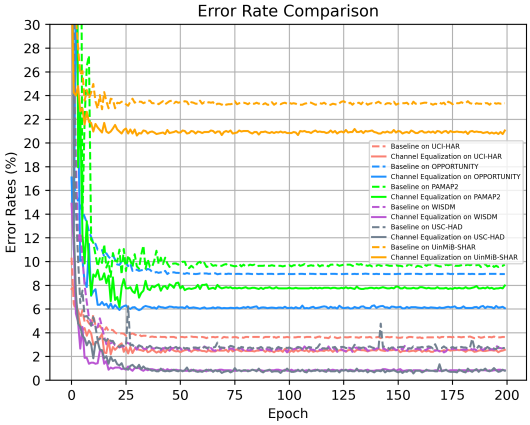


Fig. 5: Error (%) curves

comparisons between the baselines and channel-equalized networks. By inserting or removing *Channel Equalization* block, we train the *3-layer CNN* 5 times on UCI-HAR, OPPORTUNITY, and USC-HAD respectively. L_2-*norm* is used to calculate the magnitude of feature channels and the ratio of inhibited channels is averaged for our evaluation. As indicated above, *BN* utilizes additional learned scale γ_c and bias β_c to cast each channel output. For comparison, we consider a channel inactive if $|\gamma_c|$ for the channel is smaller than $1e^{-2}$ and $1e^{-3}$ respectively. That is to say, these channels whose values are smaller than a threshed value of $|\gamma_c|$ is treated as inhibited channels. The *Channel Equalization* block is inserted between final normalization layer and activation function. Results are shown in Fig.6. The blue and orange bars represent the ratio of inhibited channels within our baselines and *Channel Equalization* models. It can be observed that the ratio of inhibited channels caused by *Channel Equalization* block nearly decays by half comparing to our baselines. This similar phenomenon appears in all three cases. The results indicate that normalization layers cause too many inhibited channels, which could impair a network's generalization ability to test samples. Our *Channel Equalization* method provides a much lower ratio of inhibited channels, which implies the correlation between channels is reduced to an ideal level.

4.4 Ablation Study

Extensive ablation experiments are conducted in this part. Section 4.4.1 shows how the *Channel Equalization* block changes the ratio of inhibited channels. Section 4.4.2 investigates how the *Channel Equalization* block influences channel correlation. Section 4.4.3 explores the order of decorrelation and normalization. The effect of λ is explored in Section 4.4.4. Section 4.4.5 analyzes which is the best location to insert *Channel Equalization* block. In Section 4.4.6, the confusion matrices are used to visually show performance improvement caused by *Channel Equalization*. Section 4.4.7 provides readers with the visualization of channel equalization to indicate *Channel Equalization*'s effect. Finally, the actual operation on an embedded platform (*Raspberry Pi Model 3 B+*) is evaluated in Section 4.4.8.

4.4.1 *Reducing the ratio of inhibited channels*

To verify the effectiveness of channel inhibition and the ability of our *Channel Equalization* block to combat it, we perform

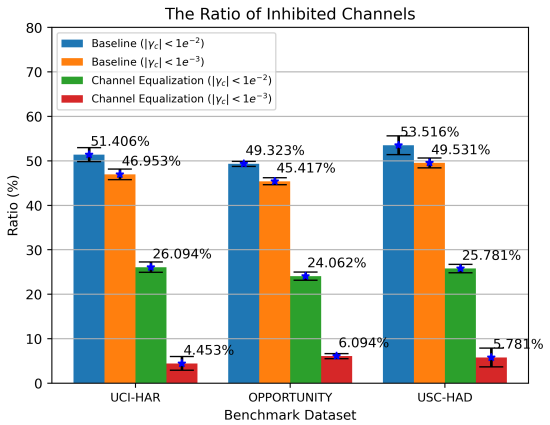
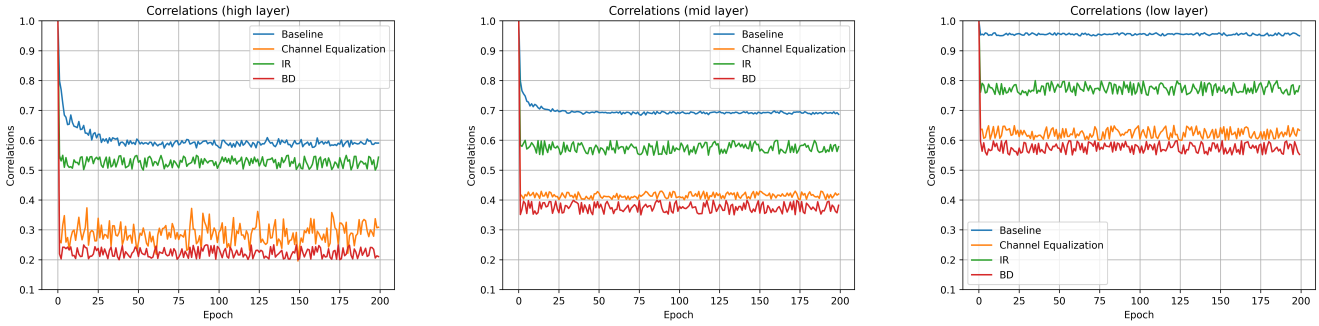


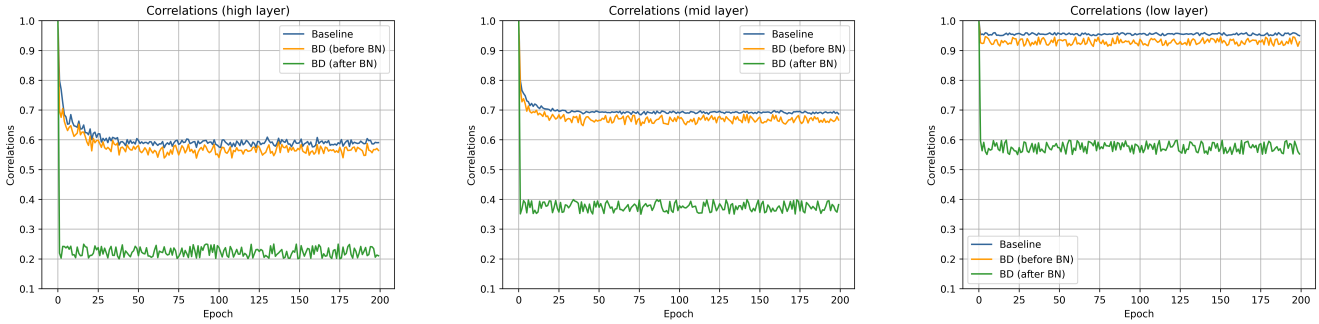
Fig. 6: The ratio (%) of inhibited channels

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3174816, IEEE Transactions on Mobile Computing



(a) Correlations at high layer (b) Correlations at middle layer (c) Correlations at low layer

Fig. 7: *PCCs* at different layers



(a) Correlations at high layer (b) Correlations at middle layer (c) Correlations at low layer

Fig. 8: The effect of *BD*'s location

4.4.2 *Decorrelations among feature channels*

As indicated in Section 3, the *Channel Equalization* block first performs channel decorrelation over a mini-batch of samples, and then performs decorrelation operation conditioned on every single sample by instance reweighting. In this part, we compare our *Channel Equalization* models with the baselines, as well as batch decorrelation and instance reweighting alone. The OPPORTUNITY dataset is used for our evaluation. In order to further investigate the effect of *Channel Equalization* block, we calculate the Pearson correlation coefficients (*PCCs*) [55] among all channels during training stage. From low to high, the curves of *PCCs* at different depths are illustrated in Fig.7. All of *Channel Equalization*, batch decorrelation and instance reweighting can provide lower channel correlations than the baselines. The correlation curves of batch decorrelation are the lowest among all cases. Since the instance reweighting itself is designed for performing decorrelations between samples, it only has a litter effect on channel decorrelation. Thus, its correlation curves are higher than those of batch decorrelation. Due to the use of instance reweighting, the correlations of our *Channel Equalization* are higher than those of batch decorrelation alone. Despite this, our *Channel Equalization* block perform the best in terms of classification performance, which can be attributed to the combination of batch decorrelation and instance reweighting. Moreover, it can be observed that our *Channel Equalization* block can learn adaptive correlations at different depths. Compared with the baselines (blue line), the correlations are remarkably decreased by *Channel Equalization* (orange line), which means that our *Channel Equalization* can effectively reduce channel dependencies. Compared with lower layers, it performs obviously better in higher layers, where there are more highly abstracted features.

4.4.3 *The order of decorrelation and normalization*

We further investigate the impact caused by decorrelation operation when it is performed before normalization. The *PCCs* curves are shown in Fig.8. Comparing to baselines at different layers, it can be seen that the correlation decreases drastically if the correlation operation is performed after normalization. Instead, if the correlation operation is performed before normalization, their *PCCs* curves are very close to those of baselines. There is a very slight decrease according to the *PCCs* curves. This is due to that the high correlation between channels itself is caused by the normalization procedure. If the decorrelation operation is performed before normalization, the correlation between channels is reduced by the decorrelation operation will increase again after normalization. Thus, the decorrelation operation should be performed after normalization in order to avoid this drawback.

4.4.4 *Different strength λ between two branches*

According to Eq.5, it can be seen that the strength of the two branches within *Channel Equalization* block is tuned by λ. To analyze its effect on final performance, we set three different λ and compare them with baselines. (a) The instance reweighting branch

alone works when λ=0. (b) The batch decorrelation branch alone works when λ=1. (c) Both instance reweighting branch and batch decorrelation branch work together when λ is learnable between (0, 1), which can be seen as a whole *Channel Equalization* case. The test accuracy among all cases are shown in Table 4. We can find that on WISDM dataset, the instance reweighting and batch decorrelation can produce an accuracy improvement of 1.33% and 1.37% respectively than the baselines. On UniMiB-SHAR dataset, the instance reweighting and batch decorrelation can produce an accuracy improvement of 0.94% and 1.4% respectively than the baselines. When both are combined, our *Channel Equalization* block may lead to an accuracy improvement of 1.83% and 2.17% on WISDM and UniMiB-SHAR respectively. The accuracy curves are shown in Fig.9, which shows that the *Channel Equalization* block performs the best among all three cases. Therefore, it is important for *Channel Equalization* block to train λ with stochastic gradient descent (*SGD*) and learn a tunable ratio, which can adaptively trade off instance reweighting and batch decorrelation in a flexible manner.

TABLE 4: Test accuracy (%) under different λ

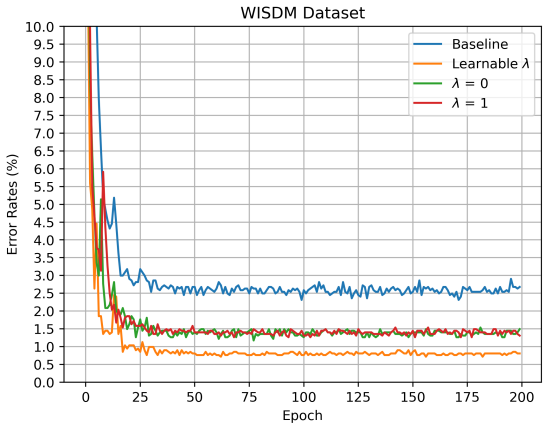
Dataset	WISDM	UniMiB-SHAR
Baseline	97.21	76.48
λ= 0 (*IR*)	98.54(+1.33)	77.42(+0.94)
λ= 1 (*BD*)	98.58(+1.37)	77.88(+1.4)
Learnable λ (*Ours*)	99.04(+1.83)	78.65(+2.17)

4.4.5 *The best location to place Channel Equalization block*

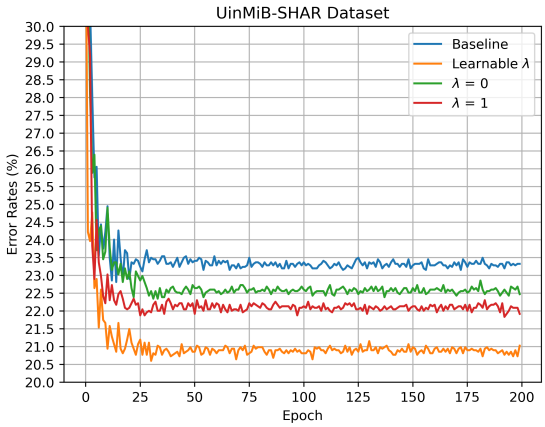
In this part, we explore which is the best location to insert the *Channel Equalization* block. On UCI-HAR dataset, we train the baselines and *Channel Equalization* models 5 times, and all accuracy are averaged for final evaluation. Results are shown in Table 5. It can be seen that inserting the *Channel Equalization* block into all convolutional layers does not bring any performance improvement. Thus, one can further decrease computational complexity by inserting the *Channel Equalization* block into only one convolutional layer. Specially, the *Channel Equalization* block performs the best at the 3^{rd} layer, rather than the 1^{st} layer and the 2^{nd} layer. Because there are more abstracted features in higher layers, it is quite beneficial to perform channel decorrelations in higher layers, instead of lower layers, which is in good agreement with the results in Fig.7. In addition, Table 5 also verifies that the *Channel Equalization* block should be inserted after normalization, which can reactivate these channels inhibited by normalization layers. As can be evidently seen, it does not bring any additional benefit to test accuracy if one inserts the *Channel Equalization* block before normalization.

4.4.6 *Confusion matrices*

In order to show the performance improvement caused by *Channel Equalization* block, we compute the confusion matrices on PAMAP2 dataset. Fig.10 shows that there are 141 examples to be correctly predicted as 'Standing', while there are 58 and 66 samples to be wrongly predicted as 'Vacuum cleaning' and 'Sitting' respectively. We find that the number of the samples correctly predicted by *Channel Equalization* is significantly larger than that of the baseline, which is increased to 273. Our *Channel Equalization* method perform better than the baseline, which confirms its superiority in this activity recognition task.



(a) WISDM dataset



(b) UniMiB-SHAR dataset

Fig. 9: Error curves (%) under different λ

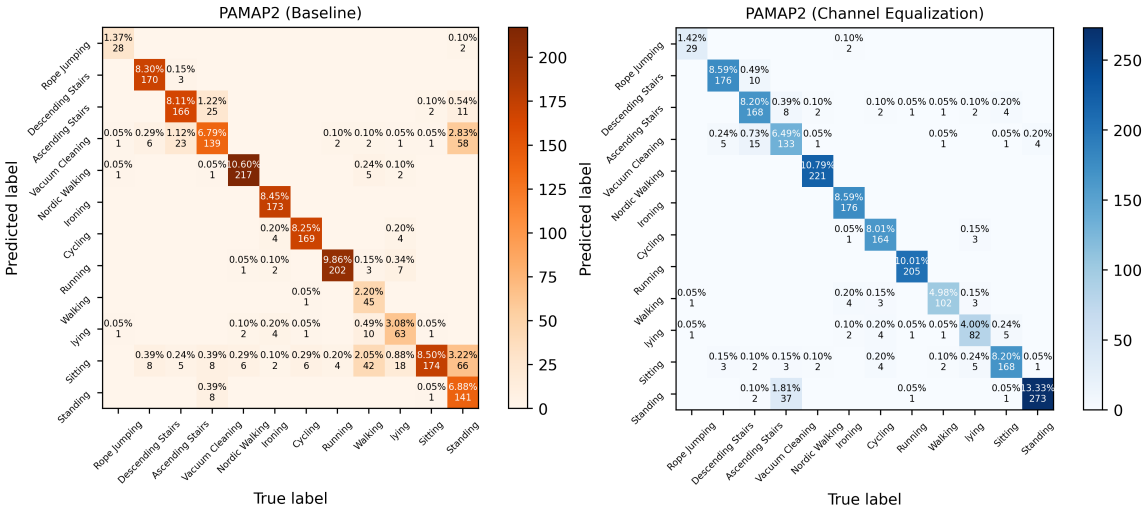
TABLE 5: Test accuracy (%) of *Channel Equalization* block at different locations

Location		Test accuracy (%)
Baseline (without *Channel Equalization*)		96.23
After BN	1^{st} layer	96.57
	2^{nd} layer	96.67
	3^{rd} layer	**97.35**
	1^{st} layer & 2^{nd} layer	96.74
	1^{st} layer & 3^{rd} layer	96.97
	2^{nd} layer & 3^{rd} layer	97.14
	All layers	96.77
Before BN	3^{rd} layer	96.25

4.4.7 *Visualization of feature channels*

The ablation experiment is conducted on USC-HAD dataset, where each subject wore a six degree of freedom (*6-DOF*) *IMU* on his/her front right hip. The *6-DOF IMU* was connected to a mini-sized laptop, which was hold on hand via *USB*. All kinds of activities were sampled from the accelerometer with 3 axes and gyroscope with 3 axes. The *x-axis* points to the ground, which is perpendicular to the plane of the *y-axis* and the *z-axis*. Fig.11 illustrates the distribution of word cloud of all sampled

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2022.3174816, IEEE Transactions on Mobile Computing



(a) Baseline (b) *Channel Equalization*

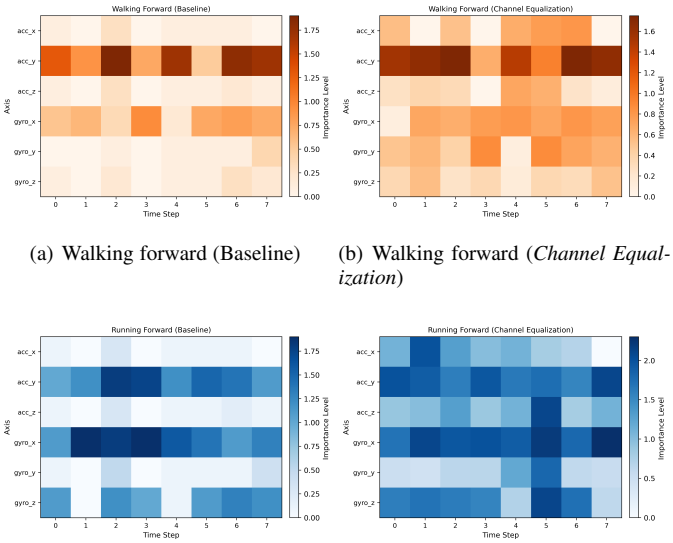
Fig. 10: Confusion Matrices

activities from this dataset, where the number of specific words is proportional to the fraction of the corresponding activities. We select the most representative two activities, *e.g.*, 'Walking forward' and 'Running forward' to present the visualizations of channels. The *Channel Equalization* model is compared with the baseline in terms of the magnitude of feature channels. Fig.12 shows that there are more inhibited channels within normal baseline, and only a few channels are activated. For example, the *acc-y*, *gyro-x*, and *gyro-z* channels contribute more to the classification of 'Walking forward' activity, while the *acc-y*, and *gyro-x* channels contribute more to the classification of 'Running forward' activity. Compared with the baseline, our *Channel Equalization* block encourages the correlation between channels to be reduced to an ideal level. In other words, because of channel equalization, these inhibited channels become more useful in the feature representation.

Fig. 11: Word cloud of USC-HAD dataset

4.4.8 *Actual test on an embedded platform (Raspberry Pi Model 3 B+)*

For efficient consideration, the actual operation of our channel equalization method is evaluated in an embedded *Raspberry Pi* platform. We implement the *HAR* system by three main steps: *1)* The model is trained by the training set from USC-HAD dataset and then saved; *2)* The saved model is loaded into the embedded platform; *3)* The loaded model run real-time inference and the



(a) Walking forward (Baseline) (b) Walking forward (*Channel Equalization*)

(c) Running forward (Baseline) (d) Running forward (*Channel Equalization*)

Fig. 12: The effectiveness of *Channel Equalization*

inference time is calculated. The platform used for testing is a *Raspberry Pi 3 Model 3B+* with *ARM Cortex-A53 (ARM v8) 64-bit SoC @ 1.4GHz* and *1GB LPDDR2 SDRAM*. A *micro SD* is used for loading *Raspbian* operating system (*OS*) and storing data. The *PyTorch* library has a good compatibility with *Raspberry Pi OS*. Fig.13 is the graphical user interface developed in *Python*. We perform inference 500 times, and the corresponding inference times are shown in Table 6 and Fig.14 respectively. It can be seen that our *Channel Equalization* block can maintain almost the same inference time with the baseline, which is of great significance in practical *HAR* applications on wearable devices.

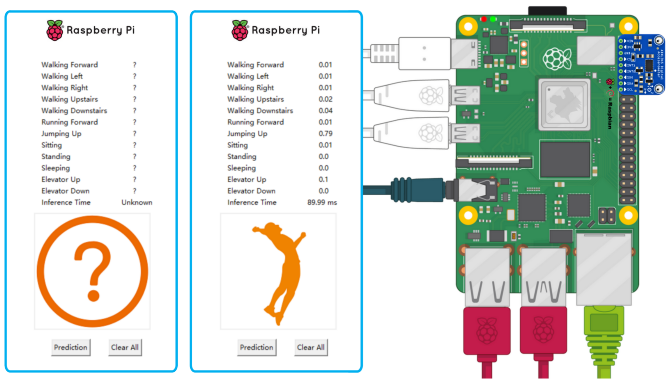


Fig. 13: Real-time *HAR* system on *Raspberry Pi Model* 3 *B+*

TABLE 6: Inference Speed of Actual Testing (*window*/*ms*)

	Number	Baseline	*Channel Equalization*
Inference time (*windows*/*ms*)	1^{st} test	68.73	69.81
	2^{nd} test	61.83	64.81
	3^{rd} test	64.92	65.83
	4^{th} test	64.85	65.82
	5^{th} test	64.76	65.83
	...		
	Average	**64.42**	**66.42**

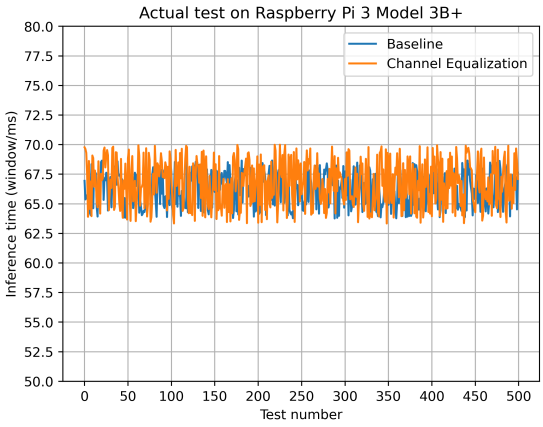


Fig. 14: Inference speed over 500 times

5 Conclusion

During the past decade, deep learning has provided competitive results in *HAR* area. As is universally recognized, normalization layers have played an indispensable role in activity recognition, which is prone to produce a 'channel collapse' effect. Most sensor channels are inhibited, which contribute very little to activity recognition. In this paper, we propose a novel architecture called as *Channel Equalization* in multi-modal *HAR* scenario. Standard normalization is modified by perform whitening or decorrelating operation on a mini-batch of samples as well as every single sample, which compels all channels to contribute more or less to feature representation, ensuring better generalization ability of a network. In particular, the *Channel Equalization* block can be readily integrated into representative convolutional architectures, whose advantages have been demonstrated on several benchmark *HAR* tasks. We evaluate actual operation of the proposed *Channel Equalization* method on a *Raspberry Pi* platform, which shows that it can maintain almost the same inference speed. We hope that the researches of *Channel Equalization* could encourage future work in architecture design for *HAR* in wearable devices.

Acknowledgments

The work was supported in part by the National Nature Science Foundation of China under Grant 61962061 and the Industry-Academia Cooperation Innovation Fund Projection of Jiangsu Province under Grant BY2016001-02, and in part by the Natural Science Foundation of Jiangsu Province under grant BK20191371. Lei Zhang is the corresponding author.

References

[1] Z. Chen, C. Cai, T. Zheng, J. Luo, J. Xiong, and X. Wang, "Rf-based human activity recognition using signal adapted convolutional neural network," *IEEE Transactions on Mobile Computing*, 2021.

[2] A. Gupta, H. P. Gupta, B. Biswas, and T. Dutta, "A fault-tolerant early classification approach for human activities using multivariate time series," *IEEE Transactions on Mobile Computing*, vol. 20, no. 5, pp. 1747–1760, 2020.

[3] P. Yang, L. Xie, C. Wang, and S. Lu, "Imu-kinect: a motion sensor-based gait monitoring system for intelligent healthcare," in *Adjunct Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the ACM International Symposium on Wearable Computers (UbiComp)*. ACM, 2019, pp. 350–353.

[4] Y. Bu, L. Xie, Y. Gong, C. Wang, L. Yang, J. Liu, and S. Lu, "Rf-dial: Rigid motion tracking and touch gesture detection for interaction via rfid tags," *IEEE Transactions on Mobile Computing*, 2020.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[6] M. Straczkiewicz, P. James, and J.-P. Onnela, "A systematic review of smartphone-based human activity recognition methods for health research," *Nature Partner Journal: Digital Medicine*, vol. 4, no. 1, pp. 1–15, 2021.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the International conference on machine learning (ICML)*. ACM, 2015, pp. 448–456.

[8] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the 14th international conference on artificial intelligence and statistics (AISTATS)*. JMLR, 2011, pp. 315–323.

[9] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International conference on machine learning (ICML)*. ACM, 2010.

[10] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz, and M. Botvinick, "On the importance of single directions for generalization," *arXiv preprint arXiv:1803.06959*, 2018.

[11] D. Mehta, K. I. Kim, and C. Theobalt, "On implicit filter level sparsity in convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019, pp. 520–528.

[12] W. Shao, S. Tang, X. Pan, P. Tan, X. Wang, and P. Luo, "Channel equilibrium networks for learning deep representation," in *Proceedings of the International Conference on Machine Learning (ICML)*. ACM, 2020, pp. 8645–8654.

[13] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proceedings of the IEEE international conference on computer vision (ICML)*. ACM, 2017, pp. 1389–1397.

[14] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE international conference on computer vision (ICML)*. ACM, 2017, pp. 2736–2744.

[15] H. Ma, W. Li, X. Zhang, S. Gao, and S. Lu, "Attnsense: Multi-level attention mechanism for multimodal human activity recognition." in *International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, 2019, pp. 3109–3115.

[16] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[17] Y. Wu and K. He, "Group normalization," in *Proceedings of the European conference on computer vision (ECCV)*. Springer, 2018, pp. 3–19.

[18] L. Huang, D. Yang, B. Lang, and J. Deng, "Decorrelated batch normalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, pp. 791–800.

[19] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra, "Reducing overfitting in deep networks by decorrelating representations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[20] S. Zhang, E. Nezhadarya, H. Fashandi, J. Liu, D. Graham, and M. Shah, "Stochastic whitening batch normalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2021, pp. 10 978–10 987.

[21] M. Zeng, H. Gao, T. Yu, O. J. Mengshoel, H. Langseth, I. Lane, and X. Liu, "Understanding and improving recurrent networks for human activity recognition by continuous attention," in *Proceedings of the ACM International Symposium on Wearable Computers*. ACM, 2018, pp. 56–63.

[22] A. Ignatov, "Real-time human activity recognition from accelerometer data using convolutional neural networks," *Applied Soft Computing*, vol. 62, pp. 915–922, 2018.

[23] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert systems with applications*, vol. 59, pp. 235–244, 2016.

[24] W. Jiang and Z. Yin, "Human activity recognition using wearable sensors by deep convolutional neural networks," in *Proceedings of the 23rd ACM international conference on Multimedia (MM)*. ACM, 2015, pp. 1307–1310.

[25] W. Huang, L. Zhang, Q. Teng, C. Song, and J. He, "The convolutional neural networks training with channel-selectivity for human activity recognition based on sensors," *IEEE Journal of Biomedical and Health Informatics*, 2021.

[26] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv preprint arXiv:1607.08022*, 2016.

[27] J. Yang, Z. Ren, C. Gan, H. Zhu, and D. Parikh, "Cross-channel communication networks," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NeurIPS)*. MIT Press, 2019, pp. 1297–1306.

[28] D. Arpit, Y. Zhou, B. Kota, and V. Govindaraju, "Normalization propagation: A parametric technique for removing internal covariate shift in deep networks," in *International Conference on Machine Learning (ICML)*. PMLR, 2016, pp. 1168–1176.

[29] M. Teye, H. Azizpour, and K. Smith, "Bayesian uncertainty estimation for batch normalized deep networks," in *International Conference on Machine Learning (ICML)*. PMLR, 2018, pp. 4907–4916.

[30] R. Zhang, Z. Peng, L. Wu, Z. Li, and P. Luo, "Exemplar normalization for learning deep representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2020, pp. 12 726–12 735.

[31] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[32] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*. IEEE, 2018, pp. 7132–7141.

[33] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. IEEE, 2019.

[34] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine," in *International workshop on ambient assisted living*. Springer, 2012, pp. 216–223.

[35] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha *et al.*, "Collecting complex activity datasets in highly rich networked sensor environments," in *2010 Seventh international conference on networked sensing systems (INSS)*. IEEE, 2010, pp. 233–240.

[36] D. Micucci, M. Mobilio, and P. Napoletano, "Unimib shar: A dataset for human activity recognition using acceleration data from smartphones," *Applied Sciences*, vol. 7, no. 10, p. 1101, 2017.

[37] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *2012 16th International Symposium on Wearable Computers (ISWC)*. IEEE, 2012, pp. 108–109.

[38] 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 International Conference on Ubiquitous Computing (UbiComp)*. ACM, 2012, pp. 1036–1043.

[39] W. Huang, L. Zhang, W. Gao, F. Min, and J. He, "Shallow convolutional neural networks for human activity recognition using wearable sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–11, 2021.

[40] Y. Guan and T. Plötz, "Ensembles of deep lstm learners for activity recognition using wearables," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 2, pp. 1–28, 2017.

[41] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Computing Surveys (CSUR)*, vol. 46, no. 3, pp. 1–33, 2014.

[42] C. Hu, Y. Chen, L. Hu, and X. Peng, "A novel random forests based class incremental learning method for activity

recognition," *Pattern Recognition*, vol. 78, pp. 277–290, 2018.

[43] I. De Falco, G. De Pietro, and G. Sannino, "Evaluation of artificial intelligence techniques for the classification of different activities of daily living and falls," *Neural Computing and Applications*, vol. 32, no. 3, pp. 747–758, 2020.

[44] 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.

[45] C. Li, D. Niu, B. Jiang, X. Zuo, and J. Yang, "Meta-har: Federated representation learning for human activity recognition," in *Proceedings of the International World Wide Web Conferences (WWW)*. ACM, 2021, pp. 912–922.

[46] 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.

[47] K. Wang, J. He, and L. Zhang, "Sequential weakly labeled multiactivity localization and recognition on wearable sensors using recurrent attention networks," *IEEE Transactions on Human-Machine Systems*, 2021.

[48] F. M. Noori, M. Riegler, M. Z. Uddin, and J. Torresen, "Human activity recognition from multiple sensors data using multi-fusion representations and cnns," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 2, pp. 1–19, 2020.

[49] H. Bi, M. Perello-Nieto, R. Santos-Rodriguez, and P. Flach, "Human activity recognition based on dynamic active learning," *IEEE Journal of Biomedical and Health Informatics*, vol. 25, no. 4, pp. 922–934, 2020.

[50] N. Y. Hammerla, S. Halloran, and T. Plötz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," in *Proceedings of the 25-th International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, 2016.

[51] F. Li, K. Shirahama, M. A. Nisar, L. Köping, and M. Grzegorzek, "Comparison of feature learning methods for human activity recognition using wearable sensors," *Sensors*, vol. 18, no. 2, p. 679, 2018.

[52] Z. N. Khan and J. Ahmad, "Attention induced multi-head convolutional neural network for human activity recognition," *Applied Soft Computing*, vol. 110, p. 107671, 2021.

[53] B. Khaertdinov, E. Ghaleb, and S. Asteriadis, "Deep triplet networks with attention for sensor-based human activity recognition," in *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2021, pp. 1–10.

[54] S. P. Singh, M. K. Sharma, A. Lay-Ekuakille, D. Gangwar, and S. Gupta, "Deep convlstm with self-attention for human activity decoding using wearable sensors," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8575–8582, 2020.

[55] P. Schober, C. Boer, and L. A. Schwarte, "Correlation coefficients: appropriate use and interpretation," *Anesthesia & Analgesia*, vol. 126, no. 5, pp. 1763–1768, 2018.

Wenbo Huang received the B.S. degree (2019) from Nanjing Tech University, Nanjing, China. He is currently pursuing the M.S. degree with Nanjing Normal University.
His research interests include human activity recognition, computer vision, and machine learning.

Lei Zhang received the B.Sc. degree (2001) in computer science from Zhengzhou Universitiy, China, and the M.S. degree (2004) in pattern recognition and intelligent system from Chinese Academy of Sciences, China. He received the Ph.D. degree (2007) from Southeast University, China, in 2011. He was a Research Fellow (2008) with IPAM, UCLA, Los Angeles.
He is currently an Associate Professor with the School of Electrical and Automation Engineering, Nanjing Normal University. His research interests include machine learning, human activity recognition and computer vision.

Hao Wu received the B.S degree (2001) in computer science from Zhengzhou University, He received M.S degree (2004) and Ph.D. degree (2007) in computer science from Huazhong University of Science and Technology. Now, he is an associate professor at School of Information Science and Engineering, Yunnan University, China. He has published more than 50 papers in peer-reviewed international journals and conferences, such as JASIST, FGCS, J.Supercomputing, KBS and PUC.
He has coauthored a monograph published in World Scientific. He has also served as reviewers and PC members for many venues. His research interests include service computing, information filtering and recommender systems.

Fuhong Min received the M.S degree (2003) from school of Communication and Control Engineering, Jiangnan University and Ph.D. degree (2007) from school of Automation, Nanjing University of Science and Technology. She was a Post-Doctoral Fellow (2009∼2010) with the School of Mechanical Engineering, University of Southern Illinois.
She is currently a professor with the school of Electrical and Automation Engineering, Nanjing Normal University. Her research interests include circuits and signal processing.

Aiguo Song (Senior Member, IEEE) received the Ph.D. degree (1996) in measurement and control from Southeast University, Nanjing. He is currently a Professor with the School of Instrument Science and Engineering, Southeast University.
He is the chair of the China Chapter of the IEEE Robotics and Automation Society. His current research interests include teleoperation, haptic display, the Internet telerobotics, distributed measurement systems and machine learning.