CapMatch: Semi-supervised Contrastive Transformer Capsule with Feature-based Knowledge Distillation for Human Activity Recognition

Zhiwen Xiao *Member, IEEE*, Huagang Tong, Rong Qu *Senior Member, IEEE*, Huanlai Xing *Member, IEEE*, Shouxi Luo *Member, IEEE*, Zonghai Zhu, and Fuhong Song

***Abstract*—This paper proposes a semi-supervised contrastive capsule transformer method with feature-based knowledge distillation (KD) that simplifies existing semisupervised learning (SSL) techniques for wearable human activity recognition (HAR), called CapMatch. CapMatch gracefully hybridizes supervised learning and unsupervised learning to extract rich representations from input data. In unsupervised learning, CapMatch leverages the pseudo-labeling, contrastive learning (CL), and feature-based KD techniques to construct similarity learning on lower- and higher-level semantic information extracted from two augmentation versions of the data, "weak" and "timecut", to recognize the relationships among the obtained features of classes in the unlabeled data. CapMatch combines the outputs of the weak- and timecut-augmented models to form pseudo-labeling and thus CL. Meanwhile, CapMatch uses the feature-based KD to transfer knowledge from the intermediate layers of the weak augmented model to those of the timecut augmented model. To effectively capture both local and global patterns of HAR data, we design a capsule transformer network consisting of four capsule-based transformers and one routing layer. Experimental results show that compared with a number of state-of-the-art semi-supervised and supervised algorithms, the proposed CapMatch achieves decent performance on three commonly used HAR datasets, namely, HAPT, WISDM, and UCI_HAR. With only 10% of data labeled, CapMatch achieves F_1 values of higher than 85.00% on these datasets, outperforming 14 semi-supervised algorithms. When the proportion of labeled data reaches 30%, CapMatch obtains F_1 values of no lower than 88.00% on the datasets above, which is better than several classical supervised algorithms, e.g., decision tree and KNN.**

***Index Terms*—Capsule Network, Contrastive Learning, Human Activity Recognition, Knowledge Distillation, Semi-supervised Learning, Similarity Learning, Wearable Sensors**

I. Introduction

HUMAN activity recognition (HAR) identifies people's actions based on their observations and environmental surroundings [1]. HAR has been widely used in various real-world domains, such as electroencephalography (EEG) analysis [2], gesture detection [3], and healthcare system [4]. With the prevalence of mobile devices, e.g., smartphones and watches, wearable HAR data collection has become accessible and convenient. Thus, wearable sensor-based HAR has grown into one of the mainstream research topics in HAR [5]. Wearable HAR data is a series of time-ordered data points collected by wearable sensor(s), e.g., triaxial accelerometer owns thee sensors producing X-, Y-, and Z-axis data simultaneously. Such a series is associated with a single or multiple time-dependent variables, i.e., univariate and multivariate [6]. A HAR algorithm captures the local and global patterns from a given time series, e.g., those associated with one variable and those across multiple variables [7], [8].

Manuscript received XX, XX; revised XX, XX (Corresponding author: H. Xing).

Z. Xiao, H. Xing, S. Luo, Z. Zhu, and F. Song are with the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu 611756, China (Email: xiao1994zw@163.com; hxx@home.swjtu.edu.cn; sxluo@swjtu.edu.cn; zzhu@swjtu.edu.cn; fhs@my.swjtu.edu.cn).

H. Tong is with College of Economic and Management, Nanjing Tech University, Puzhu Road(S), Nanjing 211816, China (Email: huagangtong@gmail.com).

R. Qu is with the School of Computer Science, University of Nottingham, Nottingham NG7 2RD 455356, UK (Email: rong.qu@nottingham.ac.uk)

Over the years, a large number of algorithms have been developed to address wearable sensor-based HAR problems, mainly through traditional and deep learning techniques [5], [6], [7]. Traditional algorithms are usually statistical or machine learning method based, which focus on capturing shallow features from HAR data. For example, Zhu and Sheng [9] introduced a hierarchical hidden Markov model for context-based recognition. Chen *et al.* [10] proposed a HAR system with coordinate transformation and principal component analysis (PCA) and online support vector machine (SVM). In contrast, deep learning ones are able to extract the intrinsic connections among representations by constructing the internal representation hierarchy of data [11], e.g., Al-qaness *et al.* [12] designed a multilevel residual network with attention for HAR feature extraction. Xia *et al.* [13] put forward a multiple-level domain adaptive learning model that used a single inertial measurement unit sensor to obtain accurate activity recognition. Shu *et al.* [14] presented a graph long short-term memory (LSTM)-in-LSTM method for group activity recognition, where person-level actions and group-level activity were modeled simultaneously. Unfortunately, all the algorithms above heavily relied on labeled data that usually consumed an incredible amount of human resource cost for raw data annotation.

Semi-supervised learning (SSL) leverages a small amount of labeled data to capture features from a dataset with a large amount of unlabeled data [15]. To mitigate the dependency on labeled data, SSL-based HAR has attracted increasingly more research interests. The SSL algorithms for HAR can be roughly classified into three categories: graph-based, self-labeled, and self-supervised. Graph-based algorithms use graph techniques to learn the similarity between the feature maps obtained from the HAR data, e.g., the multi-graph-based SSL [16], shared structure discovery SSL [17], and dynamic graph-based SSL [18]. Self-labeled ones usually adopt a supervised classifier to label instances with the unknown class without any specific input data suppositions, mainly relying on two methods: co-training [19], [20] and self-training [21], [22], [23]. Self-supervised algorithms consider a model's class prediction as a pseudo label of the training object, e.g., SelfHAR [24], CSSHAR [25], and ColloSSL [26]. In summary, the algorithms above usually use one or two of the four main SSL techniques, namely, entropy minimization, consistency regularization, pseudo-labeling, and generic regularization. Unfortunately, most SSL algorithms for HAR ignore integration of these techniques and thus have limited ability to capture rich representations from the data.

Recently, algorithms hybridizing multiple SSL techniques have become prevalent in the semi-supervised image classification community. For example, MixMatch used a single loss to integrate the entropy minimization, consistency regularization, and generic regularization [27]. ReMixMatch was an improved version of MixMatch, with distribution alignment and augmentation anchoring as two additional techniques [28]. FixMatch took the advantages of ReMixMatch and pseudo-labeling to capture sufficient representations from input data [29]. Zhang *et al.* [30] integrated curriculum pseudo-labeling into FixMatch to form FlexMatch for semi-supervised image classification.

The ensemble algorithms above, e.g., FixMatch and its variants (e.g., FlexMatch), usually consist of supervised learning based on a small amount of labeled data and unsupervised learning based on a large amount of unlabeled data. Their performance heavily depends on the representation learning on the unlabeled data, where lower- and higher-level semantic information is of significant importance [31]. However, most of the ensemble algorithms only emphasize the similarity learning on higher-level semantic information, ignoring the importance of lower-level semantic information on representation learning, which limits their ability of extracting abundant representations from unlabeled data. For example, the similarity learning in FixMatch only combines the output extracted from "weak" data and that from "strong" data through pseudo-labeling, where "weak" and "strong" are two augmentation versions of the same data. *Indeed, the performance of an algorithm is heavily dependent on the quality of lower- and higher-level semantic information obtained from the data through instance-level representation learning [31]. Therefore, it is crucial for an SSL algorithm to enhance its similarity learning on both lower- and higher-level semantic information, which ensures the algorithm's performance in unsupervised learning.*

Recently, feature-based knowledge distillation (KD), an effective form of similarity learning on lower-level semantic information, has emerged. This technique enables knowledge flow between the intermediate layers of a teacher and those of a student, helping the student obtain decent performance on instance-level representation learning [32]. On the other hand, contrastive learning (CL), a popular self-supervised learning method, studies the similarity between different views from the same sample and the similarity between the views from different samples, which improves the quality of the learned representations and thus provides rich semantic information for downstream tasks [33].

On the other hand, most SSL algorithms for HAR, e.g., ActSemiCNN [22], CSSHAR [25], and ColloSSL [26], usually use neural networks to capture features from the input. Neural networks, however, easily cause potential information loss of entities/objects due to the intrinsic translation invariance, e.g., Maxpooling. To overcome the drawback above, Sabour *et al.* [34] introduced a capsule network (CapNet) with dynamic routing mechanism to obtain entities' semantic information, e.g., location and orientation. It was reported that CapNet was quite effective in mining sufficient lower- and higher-level semantic information.

Based on FixMatch, we introduce the feature-based KD, CL and capsule-based methods to design a semi-supervised contrastive transformer capsule model for wearable HAR, called CapMatch. This model gracefully integrates supervised and unsupervised learning to mine rich representations from partially labeled data. Like most supervised capsule algorithms [35], [36], [37], CapMatch guides the prediction vectors towards the corresponding ground labels on the labeled data. On the other hand, CapMatch leverages data augmentation, pseudo-labeling, CL, and feature-based KD techniques to recognize the relationships among the features of classes obtained from the unlabeled data. CapMatch generates different views of the same sample by two data augmentation methods, namely "weak" and "timecut". Similarity learning on the lower- and higher-level semantic information extracted from the two types of augmented data is established in unsupervised learning. Meanwhile, CapMatch uses feature-based KD to transfer knowledge from the intermediate layers of weak-augmented model to those of the timecut-augmented model. The overview of CapMatch is shown in Figure 1.

Our significant contributions are summarized below.

- We propose a capsule transformer network with four capsule-based transformers and one routing layer as the CapMatch's feature extractor in Figure 1. Unlike the vanilla transformer [38], the capsule-based transformer considers the interaction rules among capsules, helping CapMatch mine abundant valuable connections and regularizations from the HAR data, e.g., the length of each capsule's vector is the capsule's entities' probability.
- CapMatch applies the pseudo-labeling, CL, and feature-based KD techniques to constructing similarity learning on the lower- and higher-level semantic information extracted from the weak and timecut versions of input data, resulting in high-quality feature extraction performance.
- Experimental results show that CapMatch outperforms 14 SSL algorithms on three widely used public HAR datasets: the smartphone-based recognition of human activities and postural transitions dataset (HAPT), wireless sensor data mining (WISDM), and University of California Irvine (UCI) HAR using smartphones (UCI_HAR) when the labeled data only takes up 10% of the training data. In particular, CapMatch overweighs a few supervised algorithms on these datasets in terms of F_1 value,

e.g., decision tree and k-nearest neighbor (KNN), when the labeled data accounts for 30% of the training data.

The rest of the paper is organized as follows. Section II reviews the related work. Section III describes the CapMatch's overall structure and its components. Section IV analyzes the experimental results, and Section V draws the conclusion.

II. Related Work

This section reviews the existing studies on wearable HAR, capsule network, CL, and KD.

A. Wearable HAR Algorithms

There have been many algorithms for addressing various wearable sensor-based HAR problems. These algorithms are either traditional or deep learning based [5], [6], [7]. Traditional algorithms usually use statistical or machine learning methods to mine shallow features from HAR data, such as PCA, SVM, KNN, Bagging, logistic regression (LR), Bayes algorithm, J48, Markov regression, collaboration algorithm, logic-based reasoning, and Fuzzy algorithm [9], [10], [39], [40], [41], [42].

On the other hand, deep learning algorithms can extract not only the shallow features but also the intrinsic regularizations and connections hidden in the data [11]. For example, Ravi *et al.* [43] introduced a temporal convolutional model for activity recognition on low-power smartphones. Zhang *et al.* [44] proposed a multi-head convolutional attention network to capture multi-scale features from HAR data. Besides, the stacked denoising autoencoder [45], graph-based LSTM-in-LSTM [14], multilevel residual network with attention [46], kernel density estimation-based model [47], Lego CNN [48], deformable convolutional network [49], multiple-level domain adaptive learning model [50], selective kernel convolution [51], and CNN-LSTM-based model [52] are all well-known HAR algorithms based on deep neural networks.

B. Capsule Network

Capsule network was developed to solve the problem of information loss of entities/objects due to translation invariance, e.g., Maxpooling [34]. In just a few years, capsule-based models have attracted increasingly more research efforts. For example, Chen *et al.* [35] proposed a contemporary novel neural network capsule architecture with multi-dimension and abundant spatial information for fault diagnosis. Feng *et al.* [36] presented a dual-routing capsule graph neural network to capture temporal and spatial features from video data. Xiao *et al.* [37] devised a multi-process collaborative capsule architecture for multi-scale feature extraction on time series classification. Saad and Chen [53] designed an efficient capsule network for Seismic Phase prediction. Sun *et al.* [54] put forward a capsule and gate recurrent unit network to recognize human activities.

C. Contrastive Learning

Recognized as one of the most effective SSL techniques, CL has been widely applied to tackle various real-world problems [33]. For instance, Feng *et al.* [55] adopted a CL-based monocular object detection model to distinguish 3-diminsional objects. In [56], an intra- and inter-Slice CL network was used to address OCT fluid segmentation problems. In [57], a CL-based joint learning framework was applied to accurate COVID-19 identification. In [58], a contrastive SSL method was utilized to capture the representations from remote sensing data. With the help of CL, pre-trained language models accelerated their fine-tuning phase and improved their generalization abilities [59].

D. Knowledge Distillation

KD encourages knowledge transfer form a cumbersome network (i.e., teacher) to a lightweight one (i.e., student). According to the knowledge form, researchers roughly divide the existing KD algorithms into three categories: response-based, feature-based, and relation-based [32]. The response-based method transfers the knowledge from the output (i.e., logits) of a teacher to that of a student [60], e.g., the multi-class KD object detection [61], fast pose distillation [62], specific expert learning [63], and collaborative teaching strategy [64]. The feature-based method enables knowledge sharing between intermediate layers of a teacher and its student instead of output-to-output, such as the FitNet [65] and knowledge representing method [66]. The relation-based method pays more attention to capturing the relationships among layers in the student and teacher models, such as the similarity-preserving approach [66], metric learning [67], and cross-layer mutual distillation [68].

III. CapMatch

This section first overviews the CapMatch's structure. Then it describes the problem formulation, capsule-based transformer, routing, data augmentation, CL, feature-based KD, and loss function one by one.

A. Overview

CapMatch contains supervised and unsupervised learning processes, as shown in Figure 1. The capsule transformer network, composed of four capsule-based transformers and one routing layer, is the model's feature extractor. CapMatch guides the prediction vectors towards the corresponding ground labels by a margin loss function on the labeled data in the supervised learning process. On the other hand, CapMatch leverages data augmentation, pseudo-labeling, CL, and feature-based KD techniques to recognize the relationships among the features of classes obtained from the unlabeled data in the unsupervised learning process. Two data augmentation methods, namely "weak" and "timecut", are used to generate different views of the same sample. CapMatch establishes similarity learning on the lower- and higher-level semantic information extracted from the weak and timecut versions of the data to enhance the instance-level representation learning

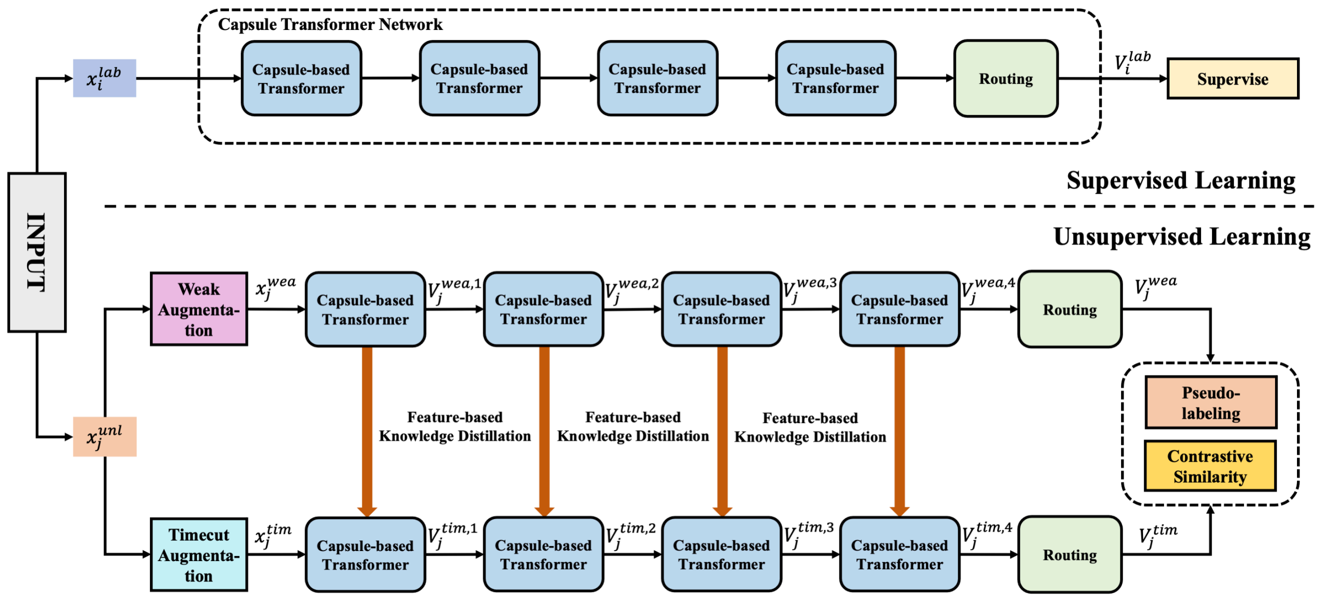


Fig. 1. The overview of CapMatch. The capsule transformer network consists of four capsule-based transformers and a routing layer.

in unsupervised learning. To be specific, CapMatch not only allows the weak-augmented artificial labels to supervise the timecut-augmented prediction vectors by a margin loss function but also combines the outputs of the weak- and timecut-augmented models to form similarity learning by a CL loss function. Meanwhile, CapMatch promotes the knowledge flow from the intermediate layers of weak-augmented model to those of the timecut-augmented model via feature-based KD.

B. Problem Formulation

Assume $x_i = \{\{x_{1,1}^{(i)}, ..., x_{1,d}^{(i)}\}, \ ..., \{\{x_{l,1}^{(i)}, ..., x_{l,d}^{(i)}\}\} \in \mathcal{X}$ is an arbitrary HAR time-series, where $\mathcal{X} \subseteq \mathbb{R}^{l\times d}$ is the input space, and l and d denote the length and dimension of x_i, respectively. $y_i \in \mathcal{Y}$ is a categorical variable associated with x_i, where $\mathcal{Y}$ is the target space. We aim at training a prediction model $\mathcal{M} : \mathcal{X} \mapsto \mathcal{Y}$ on an arbitrary dataset, $\mathcal{D} = \{\mathcal{D}_{train}, \mathcal{D}_{val}, \mathcal{D}_{test}\}$. $\mathcal{D}_{train} = \{\mathcal{D}_{train}^{lab}, \mathcal{D}_{train}^{unl}\}$, $\mathcal{D}_{val} = \{x_i, y_i\}_{i=1}^{n_{val}}$, and $\mathcal{D}_{test} = \{x_i, y_i\}_{i=1}^{n_{test}}$ are the data for training, validation, and testing, respectively, where $\mathcal{D}_{train}^{lab} = \{x_i^{lab}, y_i^{lab}\}_{i=1}^{n_{lab}}$ and $\mathcal{D}_{train}^{unl} = \{x_j^{unl}\}_{j=1}^{n_{unl}}$ are the labeled and unlabeled training data, respectively. n_{lab} and n_{unl} denote the sizes of labeled and unlabeled data, respectively. n_{val} and n_{test} are the sizes of validation and testing data, respectively.

C. Capsule-based Transformer

In the capsule transformer network, four capsule-based transformers are adopted to capture local and global patterns of the HAR data, providing rich representations with routing.

Each capsule-based transformer relates the representations at different locations of the input data to extract the intrinsic connections and regularizations among the representations obtained, as shown in Figure 2. The multi-head capsule attention, containing n_{att} capsule-based attention modules, is the core of each transformer. Each capsule-based attention module, e.g., $Attention_i$, transfers a query, $Query_i$, and its key-value pairs, Key_i-$Value_i$, to an output, V_i^{att}. Different from the vanilla attention [38], the capsule-based attention considers the interaction rules among capsules, e.g., the length of each capsule's vector is the capsule's entities' probability. V_i^{att} is defined as:

$$V_i^{att} = ||\frac{Query_i \cdot Key_i^T}{\sqrt{d_i}}|| \cdot Value_i \tag{1}$$

where, Key_i^T denotes the transpose of Key_i, d_i is the dimension of Key_i, and $||.||$ outputs the length of a given vector.

Let V_{mul} be the output of a multi-head capsule attention. V_{mul} fuses the n_{att} capsule-based attention through the CONCAT function, f_{concat}, to provide sufficient global features. V_{mul} is defined in Eq. (2).

$$V_{mul} = f_{concat}([V_1^{att}, V_2^{att}, ..., V_{n_{att}}^{att}]) \tag{2}$$

D. Routing

Following [34], [35], [36], [37], we adopt the routing layer to promote the interaction among capsules, which helps mine

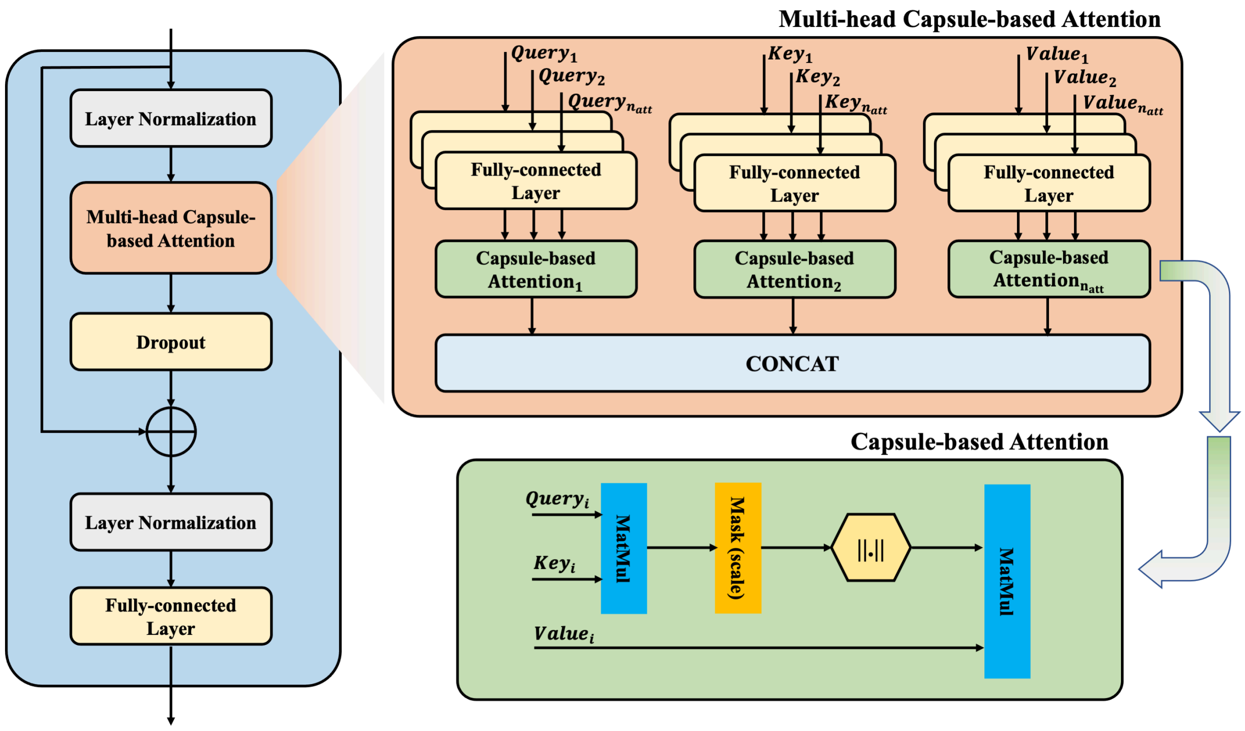


Fig. 2. The architecture of a capsule-based transformer. Note: ||.|| outputs the length of a given vector and "MatMul" represents the matrix multiplication operation.

Algorithm 1 Pseudo-code of Routing

```
1:  procedure ROUTING($s_j, n_{iter}$)    ▷ $n_{iter}$ denotes the number of iterations.
2:  Initialize weight matrix $W_{ij}$;
3:  Set $\hat{v}_{j|i} = W_{ij}s_j$ and $b_{ij} = 0$;
4:  Routing
5:      for $n_{iter}$ iterations do
6:          Obtain $k_{ij}$ using Eq. (4);
7:          Obtain $\hat{v}_{j|i}$ and $s_j$ using Eq. (3);
8:          Obtain $b_{ij}$ using Eq. (5);
9:      end for
10: return $v_j$;
11: end procedure
```

the relationships among them. Given capsule j, its input s_j is defined in Eq. (3).

$$s_j = \sum_i k_{ij}\hat{v}_{j|i}, \quad \hat{v}_{j|i} = W_{ij}v_i \tag{3}$$

where, the prediction vector, $\hat{v}_{j|i}$, is obtained by multiplying the output of capsule i in the previous layer, v_i, by a weight matrix, W_{ij}. k_{ij} is a coupling coefficient between all capsules in the current layer and capsule i in the previous layer calculated by a softmax function, $f_{softmax}$, through an iterative routing process [34], [35], [37]. k_{ij} is calculated as:

$$k_{ij} = f_{softmax}(b_{ij}) \tag{4}$$

where, b_{ij} is the log prior probabilities that capsules i and j couple. b_{ij} measures the "agreement" between the current output v_j and the prediction $\hat{v}_{j|v}$, where $\hat{v}_{j|v}$ is obtained by capsule i from the previous layer. b_{ij} is defined in Eqs. (5)-(7).

$$b_{ij} = b_{ij} + v_j \cdot \hat{v}_{j|v} \tag{5}$$

$$v_j = f_{squash}(s_j) \tag{6}$$

$$f_{squash}(x) = \frac{||x||^2}{1+||x||^2}\frac{x}{||x||} \tag{7}$$

where, v_j is output of capsule j by "squashing" its input, s_j, in the current layer. The pseudo-code of Routing is shown in Algorithm 1.

E. Data Augmentation

Data augmentation is a widely used regularization method to efficiently improve a model's robustness in deep learning [27], [28], [29]. CapMatch leverages two augmentation methods, namely "weak" and "timecut", to produce different views from the same sample as the input in the unsupervised learning process. Specifically, The weak augmentation is realized via a jitter-and-scale strategy, e.g., adding the Gaussian function to the raw data. The timecut version is a modified version of Cutout [69] that transforms a small piece of the raw unlabeled data without changing its overall trend. Figure 3 shows an example of raw data and its weak- and timecut-augmented data on the WISDM dataset.

F. Feature-based Knowledge Distillation

Feature-based KD encourages knowledge transfer between intermediate layers of the teacher and student models, improving the student's feature extraction ability [32]. Let $V_j^{wea,1}$,

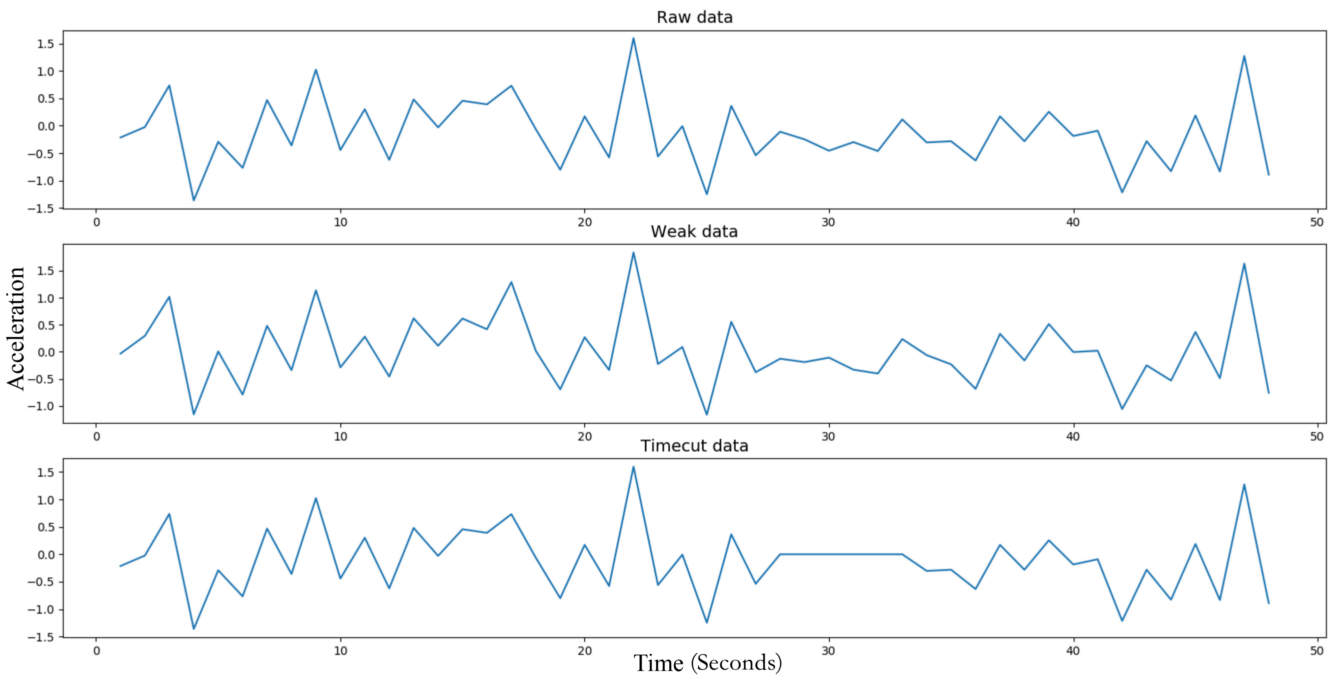


Fig. 3. Example raw data and its weak- and timecut-augmented data on the WISDM dataset.

$V_j^{wea,2}$, $V_j^{wea,3}$ and $V_j^{wea,4}$ denote the outputs of the four capsule-based transformers associated with the "weak" augmented data, x_j^{wea}, respectively; let $V_j^{tim,1}$, $V_j^{tim,2}$, $V_j^{tim,3}$ and $V_j^{tim,4}$ be the outputs of the four capsule-based transformers associated with the "timecut" augmented data, x_j^{tim}, respectively, in Figure 1.

The proposed KD loss, $\mathcal{L}_{KD}$, leverages an L_2 loss function to measure the difference between the features obtained from x_j^{wea} and those from x_j^{tim}. $\mathcal{L}_{KD}$ is written in Eq. (8)

$$\mathcal{L}_{KD} = \frac{1}{n_{unl}} \sum_{i=1}^{4} \sum_{j=1}^{n_{unl}} ||V_j^{wea,i}/t_{KD} - V_j^{tim,i}/t_{KD}||_2^2 \qquad (8)$$

where, t_{KD} is a temperature coefficient to generate a soft probability distribution over classes. In this paper, we set $t_{KD} = 1.0$ (More details can be found in Section IV.C).

G. Contrastive Learning

As aforementioned, CL distinguishes the similarity between different views from the same sample and that between the views from different samples via a CL loss function, $\mathcal{L}_{CL}$. Let V_j^{wea} and V_j^{tim} be the outputs of the capsule transformer network associated with x_j^{wea} and x_j^{tim}, respectively. As [59], [70] suggest, we define $\mathcal{L}_{CL}$ as:

$$\mathcal{L}_{CL} = -\sum_{j=1}^{n_{unl}} log \frac{exp(sim(V_j^{wea}, V_j^{tim})/t_{CL})}{\sum_{m=1}^{n_{unl}} \mathbb{1}_{[m \neq j]} exp(sim(V_j^{wea}, V_m^{tim})/t_{CL})} \qquad (9)$$

where,

$$sim(p, q) = \frac{p^T q}{||p|| ||q||},$$

and t_{CL} is a contrastive coefficient for $\mathcal{L}_{CL}$. This paper sets $t_{CL} = 1.0$ in our experiments (More details can be found in Section IV.C).

Algorithm 2 CapMatch

```
Input: $\mathcal{D} = (\mathcal{D}_{train}, \mathcal{D}_{val}, \mathcal{D}_{test})$;
Output: $\mathcal{Y}$;
 1: Initialize model parameters $\theta_0$;
 2: //Training and validation
 3: for $i = 1$ to $n_{ep}$ do    ▷ $n_{ep}$ is the number of training epochs.
 4:     Feedforward $\mathcal{D}_{train}$ into CapMatch;
 5:     Obtain $\mathcal{L}_{KD}$ using Eq. (8);
 6:     Obtain $\mathcal{L}_{CL}$ using Eq. (9);
 7:     Obtain $\mathcal{L}_{sup}$ using Eq. (10);
 8:     Obtain $\mathcal{L}_{CM}$ using Eq. (11);
 9:     Obtain $\mathcal{L}_{uns}$ using Eq. (12);
10:     Obtain $\mathcal{L}$ using Eq. (13);
11:     Update $\theta_i$ using $\theta_i = \theta_{i-1} - \eta \nabla_{\theta_{i-1}} \mathcal{L}(\theta_{i-1})$; ▷ $\eta$ is the learning rate, and $\theta_{i-1}$ and $\nabla_{\theta_{i-1}}$ denote the parameters and gradient at the ($i$-1)-th training epoch, respectively.
12:     if $i > 1$ then
13:         Validate CapMatch using $\mathcal{D}_{val}$;
14:     end if
15: end for
16: // Testing the model
17: Use the trained model to predict $\mathcal{Y}$ of $\mathcal{D}_{test}$.
```

H. Loss Function

The loss function, $\mathcal{L}$, includes a supervised loss, $\mathcal{L}_{sup}$, and an unsupervised loss, $\mathcal{L}_{uns}$. As the studies in [34], [35], [36], [37] suggest, $\mathcal{L}_{sup}$ uses the margin loss function to measure

TABLE I
DETAILS OF THE THREE HAR DATASETS.

Dataset	Sample Rate	Activities	Classes	Samples	CapMatch's Parameter (M)
HAPT	50Hz	Walking (Wk), Walking_Upstairs (Wu), Walking_Downstairs (Wd), Sitting (St), Standing (Sd), Laying (Ly), Stand-to-Sit (DtS), Sit-to-Stand (StD), Sit-to-Lie (StL), Lie-to-Sit (LtS), Stand-to-Lie(DtL), and Lie-to-Stand (LtD)	12	10,929	3.149857
WISDM	20Hz	Walking (Wk), Jogging (Jg), Upstairs (Us), Downstairs (Ds), Sitting (St), and Standing (Sd)	6	1,098,207	1.655185
UCI_HAR	50Hz	Walking (Wk), Walking_Upstairs (Wu), Walking_Downstairs (Wd), Sitting (St), Standing (Sd), and Laying (Ly)	6	10,299	1.739665

the average difference between the ground labels and their prediction vectors on labeled HAR data. $\mathcal{L}_{sup}$ is written as:

$$\begin{aligned}\mathcal{L}_{sup} = \frac{1}{n_{lab}}\sum_{i=1}^{n_{lab}}(y_i^{lab}max(0, m^+ - ||V_i^{lab}||)\\ + \lambda(1-y_i^{lab})max(0, ||V_i^{lab}|| - m^-))\end{aligned} \tag{10}$$

where, V_i^{lab} is the output of the proposed capsule transformer network associated with the labeled data, x_i^{lab}, and m^+, m^-, and λ are three coefficients for $\mathcal{L}_{sup}$. As the previous studies suggest [34], [35], [37], we set m^+ = 0.9, m^- = 0.1, and λ = 0.5.

$\mathcal{L}_{uns}$ consists of a KD loss function, $\mathcal{L}_{KD}$, a CL loss function, $\mathcal{L}_{CL}$, and a confidential marginal loss function, $\mathcal{L}_{CM}$. Similar to FixMatch [29], CapMatch leverages V_j^{wea} to generate an artificial label associated with V_j^{tim} when $max(V_j^{wea}) \geq t_{CM}$, where t_{CM} is a coefficient for $\mathcal{L}_{CM}$. $\mathcal{L}_{CM}$ is calculated by Eq. (11):

$$\begin{aligned}\mathcal{L}_{CM} = \frac{1}{n_{unl}}\sum_{j=1}^{n_{unl}}(y_j^{one}max(0, m^+ - ||V_j^{tim}||)\\ + \lambda(1-y_j^{one})max(0, ||V_j^{tim}|| - m^-))\end{aligned} \tag{11}$$

where,

$$y_j^{one} = \mathbb{1}(max(V_j^{wea}) \geq t_{CM})argmax(V_j^{wea}),$$

and $argmax(V_j^{wea})$ produces a valid one-hot probability distribution of V_j^{wea}. Following [29], we set t_{CM} = 0.95.

The unsupervised loss of CapMatch, $\mathcal{L}_{uns}$, is defined in Eq. (12).

$$\mathcal{L}_{uns} = \mathcal{L}_{CM} + \mathcal{L}_{KD} + \tau\mathcal{L}_{CL} \tag{12}$$

where, τ is a coefficient of $\mathcal{L}_{CL}$. Following the previous work in [71], we set $\tau = 0.1$.

Thus, the loss function of CapMatch, $\mathcal{L}$, is calculated as:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}_{sup} + \mathcal{L}_{uns} + \epsilon||\theta||_2^2\\ &= \mathcal{L}_{sup} + \mathcal{L}_{CM} + \mathcal{L}_{KD} + \tau\mathcal{L}_{CL} + \epsilon||\theta||_2^2\end{aligned} \tag{13}$$

where, θ represents the model parameters of CapMatch, and ϵ is a coefficient of $||\theta||_2^2$ (i.e., L_2 regularization). Following [37], we set ϵ = 0.0005 in our experiments. Besides, the CapMatch's pseudo-code is given in Algorithm 2.

IV. EXPERIMENTS

This section first describes the experimental setup, performance metrics, hyper-parameter sensitivity, and ablation study. Then, it verifies the CapMatch's overall performance and computational complexity.

A. Experimental Setting

1) Data Description: To evaluate the performance of CapMatch, we choose three widely used public HAR datasets, as follows:

- **HAPT**: the smartphone-based recognition of human activities and postural transitions dataset (HAPT) [72] was collected from 30 volunteers aged 19-48 years. The sensor signals, i.e., accelerometer and gyroscope with noise filters, were set to sample in fixed-width sliding windows of 2.56 sec and 50% overlap (128 readings/window). Each sample is a 561-feature vector with time and frequency domain variables. This dataset consists of six basic activities, i.e., standing, sitting, laying, walking, walking_downstairs and walking_upstairs, and six static postures, including stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie, and lie-to-stand.
- **WISDM**: the Wireless Sensor Data Mining (WISDM) [73] lab collected the accelerometer data every 50ms, where the signal sample rate was set to 20Hz. It has 1,098,207 multiple physical activities' examples with six attributes: user, activity, timestamp, x-acceleration, y-acceleration, and z-acceleration. This dataset considers six activities, namely, walking, jogging, upstairs, downstairs, sitting, and standing.
- **UCI_HAR**: the human activity recognition using smartphones dataset [74] in the University of California Irvine Machine Learning Repository (UCI_HAR) was collected from 30 volunteers aged 19-48 years. Each volunteer who wore a smartphone (Samsung Galaxy S II) on his/her waist performed six activities: walking, walking_upstairs, walking_downstairs, sitting, standing and laying. The 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz were used, and the signal sample rate was set to 20Hz.

The summary of the three datasets is shown in Table I. As the previous studies suggest [5], [6], [7], [44], [45], [46], we randomly divide each dataset into training and testing sets with a ratio of 7:3. Then, each training set is split into labeled and unlabeled data. Like [75], [76], [77], the ratio of labeled data to training data, $r = \frac{n_{lab}}{n_{lab}+n_{unl}}$ (s.t., $n_{lab} \ll n_{unl}$), is from 0.1 to 0.3, i.e., $r \in \{0.1, 0.2, 0.3\}$.

2) Implementation details: The hyper-parameter settings of the four capsule-based transformers are given in Table II. This paper uses RMSPropOptimizer as the optimizer, with the momentum term, initial learning rate, and decay value set to

TABLE II
HYPER-PARAMETER SETTINGS OF THE FOUR CAPSULE-BASED TRANSFORMERS.

Transformer No.	Fully-connected layer's units	n_{att}	Dropout Value
1	48	8	0.5
2	96	8	0.5
3	144	8	0.5
4	192	8	0.5

0.9, 0.001, and 0.9, respectively. We conduct the experiments with a computer with Ubuntu 18.04 OS, an Nvidia GTX 1080Ti GPU with 11GB, and an AMD R5 1400 CPU with 16G RAM.

B. Performance Metrics

As suggested in [5], [6], [7], [44], [45], [46], we use two commonly used metrics, i.e., *Accuracy* and F-measure, in performance comparison. These metrics are defined as:

$$\begin{aligned} Accuracy &= \frac{tp+tn}{tp+tn+fp+fn} \times 100\% \\ F_1 &= \frac{Precision \times Recall}{Precision + Recall} \\ Precision &= \frac{tp}{tp+fn} \times 100\% \\ Recall &= \frac{tp}{tp+tn} \times 100\% \end{aligned} \tag{14}$$

where, tp and tn are the numbers of true positive and negative samples, respectively. fp and fn represent the numbers of false positive and negative samples, respectively.

C. Hyper-parameter Sensitivity

We study the influence of hyper-parameter settings on the performance of CapMatch on the HAPT, WISDM, and UCI_HAR datasets.

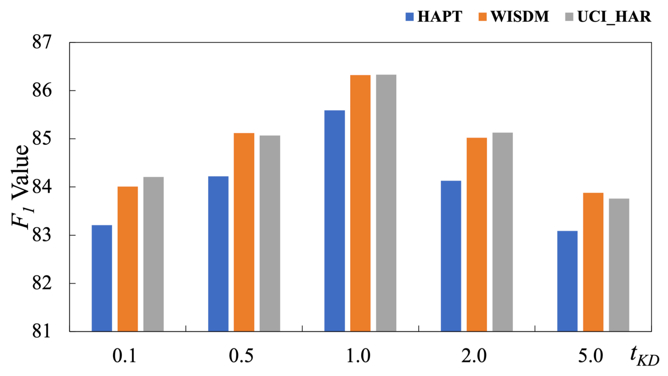


Fig. 4. F_1 results with different t_{KD} values when r = 0.1.

1) CapMatch with different t_{KD} values: t_{KD} is a scalable temperature coefficient to produce a soft probability distribution over classes. As shown in Figure 4, 1.0 is the best setting for t_{KD} because it helps CapMatch to obtain the highest F_1 value on each HAR dataset.

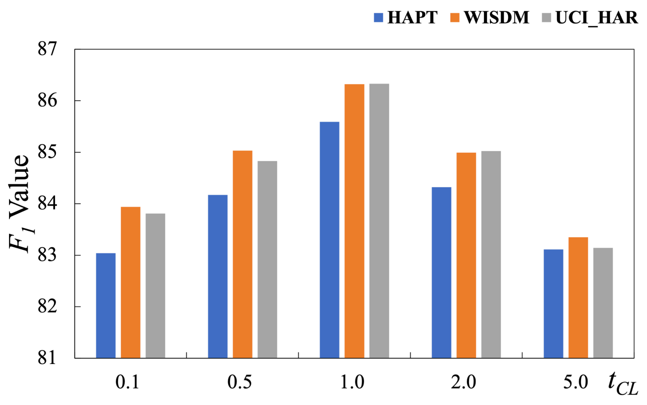


Fig. 5. F_1 results with different t_{CL} values when r = 0.1.

2) CapMatch with different t_{CL} values: t_{CL} is a threshold value for CapMatch to learn the similarity between different views from the same sample. Figure 5 shows the F_1 results obtained by CapMatch with different t_{CL} (i.e., $t_{CL} \in \{0.1, 0.5, 1.0, 2.0, 5.0\}$) values on three HAR datasets when r = 0.1. When t_{CL} = 1.0, CapMatch achieves the highest F_1 result on each dataset. That means t_{CL} = 1.0 helps CapMatch mine the connections and regularizations from the HAR data.

TABLE III
F_1 RESULTS OBTAINED BY CAPMATCH WITH DIFFERENT KD LOSSES WHEN r = 0.1. ABBREVIATIONS: L_1 – L_1 LOSS, CE – CROSS ENTROPY, HL – HUGE LOSS.

Dataset	L_1	L_2	KL	CE	HL
HAPT	85.04 ± 1.15	**85.59** ± 1.06	85.42 ± 1.12	84.69 ± 1.04	84.36 ± 1.17
WISDM	84.94 ± 0.79	**86.32** ± 0.82	84.99 ± 0.85	84.52 ± 0.86	84.15 ± 0.83
UCI_HAR	85.81 ± 1.58	**86.33** ± 1.73	85.32 ± 1.58	85.98 ± 1.65	84.92 ± 1.64

TABLE IV
F_1 RESULTS OBTAINED BY FIVE CAPMATCH VARIANTS WHEN r = 0.1.

Method	HAPT	WISDM	UCI_HAR
CapMatch w/o Routing	78.23 ± 1.13	75.05 ± 0.79	76.13 ± 1.62
CapMatch w/o CL	84.22 ± 1.12	84.15 ± 0.76	84.33 ± 1.58
CapMatch w/o FKD	83.98 ± 1.16	83.29 ± 0.74	83.53 ± 1.57
VanMatch	84.37 ± 1.04	84.68 ± 0.73	84.97 ± 1.35
CapMatch	**85.59** ± 1.06	**86.32** ± 0.82	**86.33** ± 1.73

3) CapMatch with different KD losses: It is crucial to choose an appropriate KD loss function to measure the knowledge difference between a teacher and its student. Table III shows the F_1 results obtained by CapMatch with different KD losses on three HAR datasets when r = 0.1. L_2 performs better than the other 4 losses. Hence, we choose the L_2 loss to promote the knowledge transfer within the model.

D. Ablation Study

To investigate the effects of different components on CapMatch, we compare it with four variants listed below:

- *CapMatch w/o Routing*: CapMatch without routing mechanism.
- *CapMatch w/o CL*: CapMatch without contrastive learning.

TABLE V
F_1 RESULTS OBTAINED BY VARIOUS SSL ALGORITHMS ON THREE HAR DATASETS WHEN $r = 0.1$.

Author	Method	HAPT (%)	WISDM (%)	UCI_HAR (%)
Chen *et al.* [77]	DTW-D	73.55 ± 1.41	74.25 ± 0.59	74.68 ± 1.45
Zhou *et al.* [78]	Self-labeled SVM	78.35 ± 1.29	77.25 ± 0.78	76.40 ± 1.52
	Self-labeled Clustering	81.08 ± 1.23	75.34 ± 0.36	79.32 ± 1.03
Han *et al.* [76]	SSSL	80.95 ± 1.27	80.92 ± 0.52	81.95 ± 1.47
Tang *et al.* [24]	SelfHAR	83.71 ± 1.15	82.29 ± 0.65	82.02 ± 1.22
Chen *et al.* [19]	SSRCA	–	–	81.43
Xie *et al.* [79]	UDA	81.21 ± 1.35	80.93 ± 0.73	81.94 ± 1.26
Guan *et al.* [20]	En-Co-Training	75.15 ± 1.28	77.12 ± 0.58	79.26 ± 2.04
Bhattacharya *et al.* [80]	Sparse-Coding	76.65 ± 1.25	75.93 ± 0.68	76.20 ± 2.15
Khaertdinov *et al.* [25]	SSCLHAR	82.74 ± 1.09	81.99 ± 0.46	83.32 ± 1.42
Bi *et al.* [22]	ActSemiCNN	83.54 ± 1.12	83.19 ± 0.59	82.99 ± 1.39
Berthelot *et al.* [27]	MixMatch	82.02 ± 1.21	82.72 ± 0.73	83.24 ± 1.06
Sohn *et al.* [29]	FixMatch	83.25 ± 1.13	83.82 ± 0.56	84.71 ± 1.18
Zhang *et al.* [30]	FlexMatch	83.98 ± 1.21	84.12 ± 0.61	84.99 ± 1.01
Ours	CapMatch	**85.59** ± 1.06	**86.32** ± 0.82	**86.33** ± 1.73

- *CapMatch w/o FKD*: CapMatch without feature-based KD.
- *VanMatch*: CapMatch with each capsule-based transformer replaced with the vanilla transformer [38].

Table IV shows the F_1 results obtained by five CapMatch variants on three HAR datasets when $r = 0.1$. First, we compare CapMatch with CapMatch w/o Routing to verify the contribution of routing. Clearly, the routing mechanism significantly improves the performance of CapMatch on F_1. Second, compared with CapMatch w/o CL, CapMatch results in a higher F_1 value on each dataset since CL helps enhance the quality of the representations learned by quantifying the difference between different views of the same sample. Third, we compare CapMatch with CapMatch w/o FKD. It is easily observed that the proposed CapMatch beats CapMatch w/o FKD on each dataset because feature-based KD improves knowledge transfer between intermediate layers of the model. Last but not least, we compare CapMatch with VanMatch. One can observe that CapMatch outperforms VanMatch on each dataset because the capsule-based transformer relates the capsules at different locations to mine rich connections and regulations hidden in HAR data. In summary, routing, CL, feature-based KD, and capsule-based transformer are all essential components for CapMatch.

E. Experimental Analysis

To evaluate the performance of CapMatch, we compare it with 14 SSL algorithms against F_1 value, as follows:

- *DTW-D*: a modified dynamic time warping algorithm for HAR [77].
- *Self-labeled SVM*: a modified self-labeled algorithm with SVM for HAR [78].
- *Self-labeled Clustering*: a modified self-labeled algorithm with clustering for HAR [78].
- *SSSL*: a combination of the shapelet method and pseudo-labeling for HAR [76].
- *SelfHAR*: a self-supervised learning algorithm for HAR [24].
- *SSRCA*: a semi-supervised recurrent convolutional attention algorithm for HAR [19].
- *UDA*: based on data augmentation and consistency regularization with the proposed capsule transformer network (see Figure 1) as its feature extractor [79].
- *En-Co-Training*: an SSL algorithm with co-training for HAR [20].
- *Sparse-Coding*: a sparse-coding SSL framework for HAR [80].
- *SSCLHAR*: a contrastive SSL algorithm for HAR [25].
- *ActSemiCNN*: an active semi-supervised CNN for HAR [22].
- *MixMatch*: a modified MixMatch [27] adapted to HAR, with the proposed capsule transformer network (see Figure 1) as its feature extractor.
- *FixMatch*: a modified FixMatch [29] adapted to HAR, with the proposed capsule transformer network (see Figure 1) as its feature extractor.
- *FlexMatch*: a modified FlexMatch [30] adapted to HAR, with the proposed capsule transformer network (see Figure 1) as its feature extractor.

Table V shows the F_1 results with 15 SSL algorithms on three HAR datasets when $r = 0.1$. One can easily observe that CapMatch performs the best among all compared SSL algorithms on each dataset, e.g., CapMatch obtains the highest F_1 value on the WISDM dataset, namely 86.32%. FixMatch takes the second position among all algorithms while DTW-D leads to the worst performance. The F_1 value of CapMatch is at least 1.3% higher than that of FixMatch in average with the three datasets considered.

The following explains our observations above. Based on the capsule transformer structure, CapMatch gracefully hybridizes pseudo-labeling, CL, and future-based KD, being able to capture as many intrinsic connections among the obtained representations of classes in the unlabeled data as possible. Thanks to the consistency regularization and curriculum pseudo-labeling techniques, FlexMatch can mine valuable information from the unlabeled data and achieves

TABLE VI
F_1 RESULTS OBTAINED BY CAPMATCH WITH $r = 0.3$ AND 16 EXISTING SUPERVISED ALGORITHMS ON THREE HAR DATASETS.

Training Scheme	Author	Method	HAPT (%)	WISDM (%)	UCI_HAR (%)
Semi-superved Learning	Ours	CapMatch with $r = 0.3$	88.00 ± 1.03	89.14 ± 0.89	90.02 ± 1.21
Supervised Learning (Traditional Algorithms)	Serpush *et al.* [5] Wang *et al.* [7]	SVM	94.14	88.26	94.00
		KNN	87.01	82.29	90.00
		GradientBoosting	90.00	79.71	93.90
		Random Forest	83.74	82.38	90.17
		Decision Tree	72.52	73.17	85.86
		J48	–	85.21	–
Supervised Learning (Deep Learning Algorithms)	Gu *et al.* [45]	stacked denoising autoencoder	–	94.01	–
	Zhang *et al.* [44]	1DCNN	91.15	92.13	91.72
		2DCNN	90.98	91.89	92.6
		Multi-head Convolutional Attention	94.79	95.68	95.4
	Abbaspour *et al.* [52]	CNNLSTM	93.15	94.92	94.89
		CNNBiLSTM	94.02	95.83	95.37
	Tang *et al.* [48]	LegoCNN	94.59	97.31	95.41
	Xiao *et al.* [6]	Perceptive Extraction Network	95.31	98.97	**96.33**
	Xu *et al.* [49]	Deformable CNN	–	**99.21**	–
	Gao *et al.* [51]	Selective Kernel Convolution	96.11	98.13	–

decent performance regarding F_1 value on three HAR datasets. On the other hand, DTW-D cannot explore rich features and regularizations from the unlabeled data via the DTW technique only.

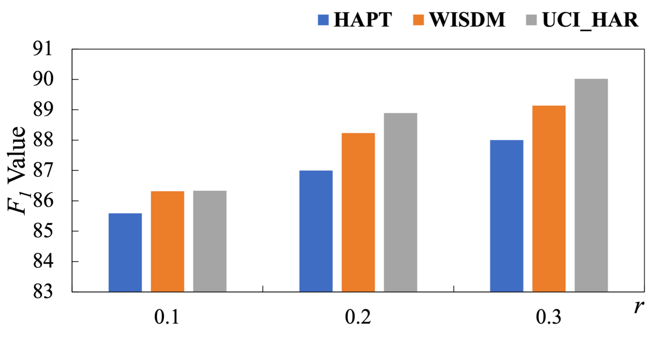


Fig. 6. F_1 results obtained by CapMatch with r = 0.1, 0.2, and 0.3 on three datasets.

Second, to study the impact of r on the performance of CapMatch, this paper shows the F_1 results obtained by CapMatch with r = 0.1, 0.2, and 0.3 on three HAR datasets in Figure 6. With more labeled data, more additional prior knowledge is brought to CapMatch, helping it mine richer relationships and regularizations from the data. That is why the CapMatch's performance is gradually enhanced as the amount of labeled data increases.

Finally, we compare CapMatch with r = 0.3 with 16 supervised algorithms on three HAR datasets and collect the F_1 results in Table VI. One can easily see that CapMatch performs worse than all deep learning-based supervised algorithms but it performs better than several traditional supervised algorithms. To be specific, CapMatch outperforms KNN, Random Forest, and Decision Tree on HAPT, performs the best on WISDM, and performs better than Decision Tree on UCR_HAR. For example, the F_1 value of CapMatch is 89.14% while that of KNN is 82.29% on the WISDM dataset. Therefore, CapMatch has the potential to address various SSL tasks in the HAR domain. Besides, to visualize the performance of CapMatch with r = 0.3, we show its confusion matrices on three datasets in Figure 7.

TABLE VII
THE NUMBER OF PARAMETERS AND RUN TIME RESULTS WITH VARIOUS ALGORITHMS ON THE WISDM AND UCI_HAR TESTING SETS.

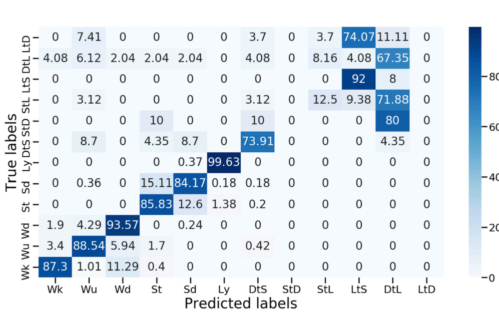
Method	WISDM			UCI_HAR		
	Parameters (M)	With CPU (s)	With GPU (s)	Parameters (M)	With CPU (s)	With GPU (s)
SVM	–	5.8365	–	–	0.2059	–
BAGGING	–	6.2953	–	–	1.1045	–
Random Forest	–	9.2659	–	–	3.4596	–
KNN	–	9.4432	–	–	0.9346	–
Multilayer Perceptron	0.089638	7.3983	1.1658	0.164396	1.0758	0.1964
1DCNN	0.772960	19.0011	8.2335	0.920145	15.2322	2.1262
2DCNN	1.124334	21.7324	7.8746	1.155719	31.6259	2.8849
LSTM	0.204324	16.8543	5.0002	1.364487	3.1376	0.9435
CNNLSTM	2.837317	69.6823	17.2398	2.913863	37.7875	7.0921
Multi-head Convolutional Attention	2.771270	22.3498	11.8345	2.896913	35.8435	5.8934
Perceptive Extraction Network	0.223558	17.9467	7.4801	0.819911	7.4029	1.4934
Deformable CNN	6.640000	–	–	–	–	–
Adaptive Deep Network	5.591000	–	–	–	–	–
Selective Kernel Convolution	0.360000	–	–	0.45	–	–
CapMatch	1.865496	21.8735	9.4529	2.637324	32.3743	4.9354

F. Computational Complexity

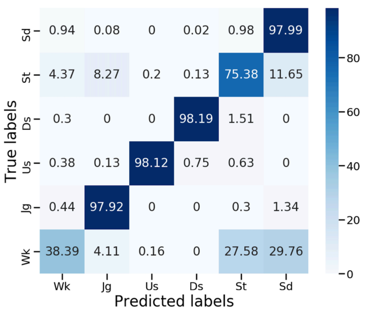
To evaluate the efficiency of CapMatch, we compare it with a number of machine and deep learning algorithms regarding the number of parameters and run time on the WISDM and UCI_HAR testing sets, as shown in Table VII. One can easily observe that CapMatch is slower than 4 machine learning algorithms, including SVM, BAGGING, Random Forest, and KNN. When compared with deep learning algorithms, CapMatch is faster than CNNLSTM and Multi-head Convolutional Attention, but slower than the others.

V. CONCLUSION

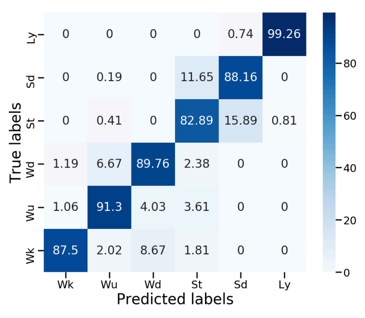
CapMatch gracefully integrates supervised learning and unsupervised learning into the proposed capsule transformer network, being able to extract abundant representations from partially labeled HAR data, where pseudo-labeling, contrastive learning, and feature-based knowledge distillation are adopted



(a) HAPT



(b) WISDM



(c) UCI_HAR

Fig. 7. Confusion matrixes of CapMatch with $r = 0.3$ on three HAR datasets.

to establish similarity learning on the lower- and higher-level semantic information extracted. As the feature extractor of CapMatch, the capsule transformer network can capture sufficient local and global patterns of HAR data. With only 10% of data labeled, CapMatch performs the best among 15 semi-supervised algorithms, achieving an F_1 value of 85.59% on HAPT, 86.32% on WISDM, and 86.33% on UCI_HAR. With 30% of data labeled, CapMatch performs even better than a number of classical supervised algorithms, achieving an F_1 value of 88.00% on HAPT, 89.14% on WISDM, and 90.02% on UCI_HAR. In particular, on the WISDM dataset, CapMatch outperforms all classical supervised algorithms for comparison, including SVM, KNN, GradientBoosting, Random Forest, Decision Tree, and J48. That reflects the potential of CapMatch to be applied to various real-world HAR problems.

References

[1] D. Anguita, L. O. A. Ghio, X. Parra, and J. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," *in Proc. ESANN 2013*, pp. 1–13, 2013.

[2] Y. Zhang, T. Zhou, W. Wu, H. Xie, H. Zhu, G. Zhou, and A. Cichocki, "Improving eeg decoding via clustering-based multitask feature learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3587–3597, 2022.

[3] Q. Li, Z. Luo, and J. Zheng, "A new deep anomaly detection-based method for user authentication using multichannel surface emg signals of hand gestures," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022.

[4] Q. Zhai, X. Han, Y. Han, J. Yi, S. Wang, and T. Liu, "A contactless on-bed radar system for human respiration monitoring," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022.

[5] F. Serpush *et al.*, "Wearable sensor-based human activity recognition in the smart healthcare system," *Computational Intelligence and Neuroscience*, 2022.

[6] Z. Xiao, X. Xu, H. Xing, F. Song, X. Wang, and B. Zhao, "A federated learning system with enhanced feature extraction for human activity recognition," *Knowledge-based Systems*, vol. 229, pp. 1–14, 2021.

[7] Y. Wang, S. Cang, and H. Yu, "A survey on wearable sensor modality centred human activity recognition in health care," *Expert Systems with Applications*, vol. 127, pp. 167–190, 2018.

[8] H. Xing, Z. Xiao, R. Qu, Z. Zhu, and B. Zhao, "An efficient federated distillation learning system for multitask time series classification," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[9] C. Zhu and W. Sheng, "Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 41, no. 3, pp. 569–573, 2011.

[10] Z. Chen, Q. Zhu, Y. C. Soh, and L. Zhang, "Robust human activity recognition using smartphone sensors via ct-pca and online svm," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3070–3080, 2017.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, pp. 436–444, 2015.

[12] M. A. A. Al-qaness, A. Dahou, M. A. Elaziz, and A. M. Helmi, "Multi-resatt: Multilevel residual network with attention for human activity recognition using wearable sensors," *IEEE Transactions on Industrial Informatics*, 2022.

[13] S. Xia, L. Chu, L. Pei, Z. Zhang, W. Yu, and R. C. Qiu, "Learning disentangled representation for mixed-reality human activity recognition with a single imu sensor," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 70, no. 2514314, pp. 1–14, 2021.

[14] X. Shu, L. Zhang, Y. Sun, and J. Tang, "Host–parasite: Graph lstm-in-lstm for group activity recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 663–674, 2021.

[15] J. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, pp. 373–440, 2020.

[16] M. Stikic, D. Larlus, and B. Schiele, "Multi-graph based semi-supervised learning for activity recognition," *In Proc. IEEE Int. Symp. Wearable Comput. (ISWC)*, pp. 85–92, 2009.

[17] L. Yao, F. Nie, Q. Z. Sheng, T. Gu, X. Li, and S. Wang, "Learning from less for better: Semi-supervised activity recognition via shared structure discovery," *In Proc. ACM Ubicomp*, pp. 13–24, 2016.

[18] W. Liu, S. Fu, Y. Zhou, Z. Zha, and L. Nie, "Human activity recognition by manifold regularization based dynamic graph convolutional networks," *Neurocomputing*, vol. 444, pp. 217–225, 2021.

[19] K. Chen, L. Yao, D. Zhang, X. Wang, X. Chang, and F. Nie, "A semisupervised recurrent convolutional attention model for human activity recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 5, pp. 1747–1756, 2020.

[20] D. Guan, W. Yuan, Y.-K. Lee, A. Gavrilov, and S. Lee, "Activity recognition based on semi-supervised learning," *In Proc. 13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, pp. 469–475, 2007.

[21] M. Stikic, K. V. Laerhoven, and B. Schiele, "Exploring semi-supervised and active learning for activity recognition," *In Proc. IEEE Wearable Comput. (ISWC)*, pp. 81–88, 2008.

[22] H. Bi, M. Perello-Nieto, P. Santos-Rodriguez, P. Flach, and I. Craddock, "An active semi-supervised deep learning model for human activity recognition," *Journal of Ambient Intelligence and Humanized Computing*, 2022.

[23] H. Qian, S. Pan, and C. Miao, "Distribution-based semi-supervised learning for activity recognition," *In Proc. AAAI*, 2019.

[24] C. Tang *et al.*, "Selfhar: Improving human activity recognition through self-training with unlabeled data," *In Proc. the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 5, no. 36, pp. 1–30, 2021.

[25] B. Khaertdinov, E. Ghaleb, and S. Asteriadis, "Contrastive self-supervised learning for sensor-based human activity recognition," *In Proc. 2021 IEEE International Joint Conference on Biometrics (IJCB)*, pp. 1–8, 2021.

[26] Y. Jain *et al.*, "Collossl: Collaborative self-supervised learning for human

activity recognition," *In Proc. the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 6, no. 17, pp. 1–28, 2022.

[27] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *In Proc. NeurIPS 2019*, pp. 5050–5060, 2019.

[28] D. Berthelot, N. Carlini, E. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "Remixmatch: Semi-supervised learning with distribution alignment and augmentation," *In Proc. ICLR*, pp. 1–13, 2020.

[29] K. Sohn, D. Berthelot, C. Li, Z. Zhang, N. Carlini, E. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *In Proc. NeurIPS*, pp. 1–12, 2020.

[30] B. Zhang, Y. Wang, W. Hou, H. Wu, J. Wang, M. Okumura, and T. Shinozaki, "Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling," *In Proc. NeurIPS*, 2021.

[31] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[32] J. Gou, B. Yu *et al.*, "Knowledge distillation: A survey," *Int. J. Comput. Vis.*, vol. 129, pp. 1789–1819, 2021.

[33] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Intell.*, vol. 43, no. 11, pp. 4037–4058, 2021.

[34] S. Sabour, N. Frosst, and G. Hinton, "Dynamic routing between capsules," *In Proc. NeurIPS*, pp. 3856–3866, 2017.

[35] L. Chen, N. Qin, X. Dai, and D. Huang, "Fault diagnosis of high-speed train bogie based on capsule network," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 6, pp. 6203–6211, 2020.

[36] Y. Feng, J. Gao, and C. Xu, "Learning dual-routing capsule graph neural network for few-shot video classification," *IEEE Transactions on Multimedia*, 2022.

[37] Z. Xiao, X. Xu, H. Zhang, and E. Szczerbicki, "A new multi-process collaborative architecture for time series classification," *Knowledge-Based Systems*, vol. 220, pp. 1–11, 2021.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kariser, and I. Polosukhin, "Attention is all you need," *In Proc. NeurIPS*, pp. 5998–6008, 2017.

[39] A. Gupta, A. Kembhavi, and L. S. Davis, "Observing human-object interactions: Using spatial and functional compatibility for recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1775–1789, 2009.

[40] S. Xia, L. Chu, L. Pei, Z. Zhang, W. Yu, and R. C. Qiu, "Learning disentangled representation for mixed- reality human activity recognition with a single imu sensor," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021.

[41] D. Tao, L. Jin, Y. Wang, and X. L, "Rank preserving discriminant analysis for human behavior recognition on wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 813–823, 2014.

[42] R. A. Hamad, A. S. Hidalgo, M. R. Bouguelia, M. E. Estevez, and J. M. Quero, "Efficient activity recognition in smart homes using delayed fuzzy temporal windows on binary sensors," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 2, pp. 387–395, 2020.

[43] D. Ravi, C. Wong, B. Lo, and G. Yang, "Deep learning for human activity recognition: a resource efficient implementation on low-power devices," *In Proc. IEEE 13th ICWIBSN*, pp. 71–76, 2016.

[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 Things J.*, vol. 7, no. 2, pp. 1072–1080, 2020.

[45] F. Gu, K. Khoshelham, S. Valaee, J. Shang, and R. Zhang, "Locomotion activity recognition using stacked denoising autoencoders," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2085–2093, 2018.

[46] M. A. A. Al-qaness, A. Dahou, M. A. Elaziz, and A. M. Helmi, "Multi-resatt: Multilevel residual network with attention for human activity recognition using wearable sensors," *IEEE Transactions on Industrial Informatics*, 2022.

[47] Y. Dong, J. D. X. Li, M. O. Khyam, M. Noor-A-Rahim, and S. S. Ge, "Dezert-smarandache theory-based fusion for human activity recognition in body sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 11, pp. 7138–7149, 2020.

[48] 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," *ArXiv Preprint arXiv:2005.03948*, 2020.

[49] S. Xu, L. Zhang, W. Huang, H. Wu, and A. Song, "Deformable convolutional networks for multimodal human activity recognition using wearable sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–14, 2022.

[50] S. Xia, L. Chu, L. Pei, Z. Zhang, W. Yu, and R. C. Qiu, "Learning disentangled representation for mixed- reality human activity recognition with a single imu sensor," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–14, 2021.

[51] W. Gao, L. Zhang, W. Huang, F. Min, J. He, and A. Song, "Deep neural networks for sensor-based human activity recognition using selective kernel convolution," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–13, 2021.

[52] S. Abbaspour, F. Fotouhi, A. Sedaghatbaf, H. Fotouhi, M. Vahabi, and M. Linden, "A comparative analysis of hybrid deep learning models for human activity recognition," *Sensors*, vol. 20, no. 19, pp. 1–14, 2020.

[53] O. M. Saad and Y. Chen, "Capsphase: Capsule neural network for seismic phase classification and picking," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, no. 5904311, pp. 1–11, 2022.

[54] X. Sun *et al.*, "Capsganet: Deep neural network based on capsule and gru for human activity recognition," *IEEE Systems Journal*, 2022.

[55] D. Feng, S. Han, H. Xu, X. Liang, and X. Tan, "Point-guided contrastive learning for monocular 3-d object detection," *IEEE Transactions on Cybernetics*, 2022.

[56] X. He, L. Fang, M. Tan, and X. Chen, "Intra- and inter-slice contrastive learning for point supervised oct fluid segmentation," *IEEE Transactions on Image Processing*, vol. 31, pp. 1870–1881, 2022.

[57] Z. Wang, Q. Liu, and Q. Dou, "Contrastive cross-site learning with redesigned net for covid-19 ct classification," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 10, pp. 2806–2813, 2020.

[58] H. Jung, Y. Oh, S. Jeong, C. Lee, and T. Jeon, "Contrastive self-supervised learning with smoothed representation for remote sensing," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, no. 8010105, pp. 1–5, 2022.

[59] Y. Su *et al.*, "Css-lm: A contrastive framework for semi-supervised fine-tuning of pre-trained language models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2930–2941, 2021.

[60] G. Hinton, O. Vinyals, and J. Dean, "Distillation the knowledge in a neural network," *arXiv preprint arXiv: 1503.02531*, 2015.

[61] G. Chen, W. Choi, X. Yu, T. Han, and M. Chandraker, "Learning efficient object detection models," *In Proc. NeurIPS*, pp. 742–751, 2017.

[62] F. Zhang, X. Zhu, and M. Ye, "Fast human pose estimation," *In Proc. IEEE CVPR*, pp. 3512–3521, 2019.

[63] W. C. Kao, H. X. Xie, C. Y. Lin, and W. H. Cheng, "Specific expert learning: Enriching ensemble diversity via knowledge distillation," *IEEE Transactions on Cybernetics*, 2022.

[64] H. Zhao, X. Sun, J. Dong, C. Chen, and Z. Dong, "Highlight every step: Knowledge distillation via collaborative teaching," *IEEE Transactions on Cybernetics*, vol. 52, no. 4, pp. 2070–2081, 2022.

[65] A. Romero, N. Ballas, S. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnet: hints for thin deep nets," *In Proc. ICLR*, 2015.

[66] J. Liu, D. Wen, H. Gao, W. Tao, T. Chen, K. Osa, and M. Kato, "Knowledge representing: efficient, sparse, representation of prior knowledge for knowledge distillation," *In Proc. IEEE CVPR 2019 Workshop*, 2019.

[67] L. Yu, V. Yazici, X. Liu, J. Weijer, Y. Chen, and A. Ramisa, "Learning metrics from teachers: compact networks for image embedding," *In Proc. IEEE CVPR*, pp. 2902–2911, 2019.

[68] A. Yao and D. Sun, "Knowledge transfer via dense cross-layer mutual-distillation," *In Proc. ECCV*, pp. 294–311, 2020.

[69] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[70] Y. Zhang, T. Xiang, T. Hospedales, and H. Lu, "Deep mutual learning," *in Proc IEEE CVPR*, pp. 4320–4328, 2018.

[71] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," *In Proc. IEEE CVPR*, 2021.

[72] J.-L. Reyes-Ortiz, L. Oneto, A. Sam, X. Parra, and D. Anguita, "Transition-aware human activity recognition using smartphones," *Neurocomputing*, vol. 171, pp. 754–767, 2016.

[73] J. R. Kwapisz, G. M. Weiss, and S. Moore, "Activity recognition using cell phone accelerometers," *In Proc. the Fourth International Workshop on Knowledge Discovery from Sensor Data (at KDD-10)*, 2010.

[74] 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 Proc. WAAL*, 2012.

[75] M. Gonźalez, C. Bergmeir, I. Triguero, Y. Rodíguerz, and J. Benítez, "Self-labeling techniques for semi-supervised time series classification: an empirical study," *Knowl. Inf. Syst.*, vol. 55, pp. 493–528, 2018.

[76] H. Wang, Q. Zhang, J. Wu, S. Pan, and Y. Chen, "Time series feature learning with labeled and unlabeled data," *Pattern Recognition*, vol. 89, pp. 55–66, 2019.
[77] Y. Chen, B. Hu, E. Keogh, and G. E. Batista, "Dtw-d: Time series semi-supervised learning from a single example," *In Proc. SIGKDD*, 2013.
[78] Y. Zhou, Q. She, Y. Ma, W. Kong, and Y. Yang, "Transfer of semi-supervised broad learning system in electroencephalography signal classification," *Neural Computing and Applications*, vol. 33, pp. 10 597–10 613, 2021.
[79] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," *arXiv preprint arXiv:1904.12848*, 2019.
[80] S. Bhattacharya, P. Nurmi, N. Hammerla, and T. Plötz, "Using unlabeled data in a sparse-coding framework for human activity recognition," *Pervasive and Mobile Computing*, vol. 5, pp. 242–262, 2014.

Zhiwen Xiao (M), received his B. Eng. degree in network engineering from Chengdu University of Information Technology in 2019. He is currently studying at Southwest Jiaotong University. His research interests are deep learning, federated learning, representation learning, data mining, and computer vision.

Huagang Tong, received his Ph.D. degree in management sciences from Nanjing University of Aeronautics and Astronautics. He is with College of Economic and Management, Nanjing Tech University, Puzhu Road(S), Nanjing 211816, China. His research interests are knowledge discovery, operation research, and data mining.

Rong Qu (SM'12) is an Associate Professor at the School of Computer Science, University of Nottingham. She received her B.Sc. in Computer Science and Its Applications from Xidian University, China in 1996 and Ph.D. in Computer Science from The University of Nottingham, U.K. in 2003. Her research interests include the modelling and optimisation for logistics transport scheduling, personnel scheduling, network routing, portfolio optimization and timetabling problems by using evolutionary algorithms, mathematical programming, constraint programming in operational research and artificial intelligence. These computational techniques are integrated with knowledge discovery, machine learning and data mining to provide intelligent decision support on logistic fleet operations at SMEs, workforce scheduling at hospitals, policy making in education, and cyber security for connected and autonomous vehicles.

Dr. Qu is an associated editor at IEEE Computational Intelligence Magazine, IEEE Transactions on Evolutionary Computation, Journal of Operational Research Society and PeerJ Computer Science. She is a Senior IEEE Member since 2012 and the Vice-Chair of Evolutionary Computation Task Committee since 2019 and Technical Committee on Intelligent Systems Applications (2015-2018) at IEEE Computational Intelligence Society. She has guest edited special issues on the automated design of search algorithms and machine learning at the IEEE Transactions on Pattern Analysis and Machine Intelligence and IEEE Computational Intelligence Magazine.

Huanlai Xing (M), received his Ph.D. degree in computer science from the University of Nottingham (advisor: Dr Rong Qu), Nottingham, UK, in 2013. He was a visiting scholar with the CISA lab led by Dr. Haibo He at the University of Rhode Island, Kingston, USA, in 2020. Huanlai Xing is an Associate Professor with the School of Computing and Artificial Intelligence, Southwest Jiaotong University. His research interests include representation learning, data mining, reinforcement learning, machine learning, network function virtualization, and software defined networking.

Shouxi Luo (M), received his bachelor's degree in communication engineering and the Ph.D. degree in communication and information systems from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2016, respectively.,He is a Lecturer with the School of Information Science and Technology, Southwest Jiaotong University, Chengdu. His research interests include data center networks, software-defined networking, and networked systems.

Zonghai Zhu received his B.Sc. degree in the Department of Information, Mechanical and Electrical Engineer, Shanghai Normal University, China, in 2010, and received his Ph.D. degree from the Department of Computer Science and Engineering, East China University of Science and Technology, China, in 2021. He is now an Assistant Professor in the School of Computing and Artificial Intelligence, Southwest Jiaotong University, China. His research interests include imbalanced problems, kernel-based methods, and graph-structured data.

Fuhong Song, received the M.E. degree from Southwest Jiaotong University, Chengdu, China, in 2018. He is currently pursuing the Ph.D. degree in the School of Information Science and Technology, Southwest Jiaotong University, Chengdu, China.His research interests include edge computing, multi-objective optimization, and reinforcement learning.