Self-Supervised Few-Shot Time-Series Segmentation for Activity Recognition

Chunjing Xiao, Shiming Chen, Fan Zhou, *Member, IEEE*, and Jie Wu, *Fellow, IEEE*

Abstract—Extracting valuable activity segments from continuously received sensor data is a key step for many downstream applications such as activity recognition, trajectory prediction, and gesture recognition. Numerous unsupervised and supervised approaches have been proposed for activity segmentation. However, current unsupervised methods generally suffer from subject and environment-dependent problems, and supervised methods require a great many labeled data which is time-consuming and expensive to be collected. To address these issues, we propose a Self-supervised Few-shot Time-series Segmentation framework called SFTSeg, which introduces few-shot learning to conduct activity segmentation only relying on several labeled target samples. To be applicable to time-series data, we design a line-level data augmentation method to build a consistency regularization for the few-shot learning framework, which can augment limited labeled target samples to enhance generalization capacity of the model. Also, we devise a time series-specific pretext task to construct a self-supervised loss with adaptive weighting, which can adopt unlabeled target data to enable the model to learn characteristics of the target data and further improve segmentation performance. The experiments illustrated that SFTSeg achieves obvious gains compared to state-of-the-art methods.

Index Terms—Sensor data, wearable, activity recognition, segmentation

1 Introduction

Human activity recognition based on sensor data is considered a key aspect for many emerging Internet of Things applications, such as smart homes and health care. To conduct activity recognition, continuously received data should be segmented into subsequences, each of which is associated with a single activity [1], [2]. The segmented sequences will then be fed into a specific classification model, e.g., support vector machines or neural networks, to perform activity recognition. This means the segmentation results have a significant effect on the performance of activity recognition. Consequently, lots of studies have been initiated on activity segmentation, including unsupervised methods [3], [4] and supervised models [5], [6].

Although these approaches achieve remarkable success, they still suffer from one or more shortcomings. The unsupervised approaches have the advantages of avoiding collecting a lot of labeled data, but are confronted with subject and environment-dependent problems. Unsupervised methods mainly fall into three main genres: Change Point Detection (CPD)-based [3], [7], threshold-based [1], [8] and temporal shape-based [4], [9] approaches. Both CPD-based and threshold-based methods require a threshold value to discriminate activity boundaries [2], [10]. However, the optimal threshold value needs to be determined depending on subjective experiences and specific applications [5], [7]. Also, temporal shape-based approaches such as FLOSS [4] require specific information of the given problem to determine temporal constraint parameters, which is environment-dependent [9]. Supervised approaches can alleviate the subject and environment-dependent problems. However, these methods require a great many labeled data for model training to obtain expected performance, and it is time-consuming, expensive and sometimes even infeasible to collect enough annotated activity data in practical applications [8], [10].

To address these shortages, we first transform activity segmentation tasks into classification problems, and then introduce few-shot learning to conduct classification. Specifically, similar to the work [5], we first divide continuous sensor streams into equally sized windows according to average activity length. And then each window is assigned one state label from the four states: *static-state*, *start-state*, *motion-state* and *end-state*. Finally, the activities boundaries are identified according to the state labels. In this way, the task of activity segmentation becomes how to design an appropriate state inference model to infer the state label of each window. To address the difficulty of gathering many labeled data, we introduce few-shot learning as the basic state inference framework. Few-shot learning, which is widely used in the fields of image classification, sentiment classification and object recognition, can transfer knowledge from source domains to target domains and predict classes for new examples only depending on several labeled samples [11].

- *Chunjing Xiao and Shiming Chen are with the Henan Key Laboratory of Big Data Analysis and Processing, Henan University, Kaifeng, Henan 475004, China. E-mail: chunjingxiao@gmail.com, shimingchen@henu.edu.cn.*
- *Fan Zhou is with the School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, China. E-mail: fan.zhou@uestc.edu.cn.*
- *Jie Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122 USA. E-mail: jiewu@temple.edu.*

Manuscript received 14 January 2022; revised 8 June 2022; accepted 5 August 2022. Date of publication 16 August 2022; date of current version 3 October 2023.
This work was supported in part by the National Natural Science Foundation of China under Grant 62072077 and in part by the Science and Technology Foundation of Henan Province of China under Grant 212102210387.
(Corresponding author: Fan Zhou.)
Digital Object Identifier no. 10.1109/TMC.2022.3199015

However, when directly applying few-shot learning to activity segmentation tasks, there are two additional issues. (1) In typical few-shot learning models, the performance will significantly degrade when there exists a large shift between source and target domains [12], [13]. In fact, this situation happens frequently in practical scenarios of activity segmentation. For instance, the target data might be obtained under different sensor devices, persons, and environments from the source data, leading to different data styles and characteristics between them. (2) In few-shot learning tasks, the categories of target data are typically different from the ones of source data [11]. However, for activity segmentation, both source data and target data have the same categories (*static-state*, *start-state*, *motion-state* and *end-state*). Based on this fact, the general few-shot learning approaches should be advanced for better performance.

In this paper, we present a Self-supervised Few-shot Time-series Segmentation framework, SFTSeg, to segment activities on time-series data. In SFTSeg, we design a new consistency regularization loss and self-supervised loss with adaptive weighting for the metric learning-based few shot learning model. In particular, to alleviate the large shift between source and target domains, we introduce consistency regularization to exploit limited labeled target data for model training. Consistency regularization is one such method which enforces the model's prediction to be consistent if perturbations are added to the input data points, and widely adopted in image processing and natural language processing [14], [15], [16] but has not been studied in activity segmentation fields. However, general perturbation methods are not appropriate for time-series data. For instance, the perturbation approaches for images mainly aim at generating pixel-level changes, while time-series data requires line-level variations since its data is a kind of waveform along with time. Hence, we propose a line-level data augmentation method for time-series data, which shrinks source data as perturbations, and injects these perturbations into the target data as augmented data according to a warping path generated using Dynamic Time Warping (DTW) [17], [18]. Finally, we build a consistency regularization loss to force the augmented data and original data to be assigned the same state label for enhancing classifier smoothness.

To enhance generalization capacity to the target data, we explore unlabeled target data using self-supervised techniques to enable the model to capture characteristics of the target data. The paradigm of self-supervised learning defines an annotation-free pretext task to provide a surrogate supervision signal for feature learning [19], [20], and has shown remarkable success in unsupervised representation learning for image processing [21] and natural language processing [22]. However, the prevailing pretext tasks are inappropriate for activity segmentation. For example, the typical pretext task, image rotation [23], may not benefit activity segmentation, because the sequence with the *start-state* label will be quite similar to the one with the *end-state* label when the former rotates 180° (referring to Fig. 1 for visual inspection). Additionally, forcing the model to assign the same label to the rotated data and original data may degrade model performance. Therefore, we propose a time series-specific pretext task to extract training sample pairs from unlabeled target data for the self-supervised loss. Further we design an adaptive weighting to adjust the importance of different sample pairs during model training. This training process based on unlabeled target data can enable the model to learn more characteristics of the target data and further improve the classification performance. This framework can be applied to a variety of applications related to activity recognition under cross-domain scenarios, such as healthcare and smart homes.

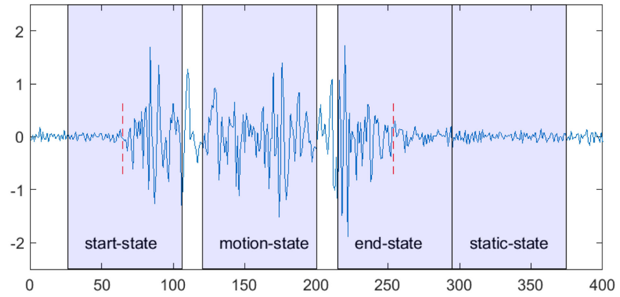


Fig. 1. Four kinds of states extracted from an activity.

We summarize the main contributions of this paper as follows:

- We present a self-supervised few-shot time-series segmentation framework, SFTSeg, which can address the subject and environment-dependent problems only relying several labeled target samples.
- We design a line-level data augmentation method for building the consistency regularization loss based on limited labeled target data, which can enhance generalization capacity of the model.
- We propose a time series-specific pretext task with adaptive weighting to construct a self-supervised loss based on unlabeled target data, which can boost the model in capturing characteristics of target data.
- Experiment results based on four datasets from different sensors illustrate that our framework outperforms the state-of-the-art approaches in various scenarios. The data and code are available online.[1]

2 Preliminaries

In this section, we present a preliminary overview of the problem concerning how to transform the activity segmentation task into the classification job and few-show learning. And these will serve as background or key design ingredients of our SFTSeg framework.

2.1 Problem Statement

Activity segmentation aims to identify when activities start and end, which is the first step in human activity recognition. Because of the difficulty of collecting labeled data from the target domain, unsupervised methods are widely adopted for activity segmentation, such as CPD-based [3], [7] and threshold-based [1], [8] approaches . However, these methods are confronted with subject and environment-dependent problems [5], [7]. Therefore, similar to the

[1] https://github.com/ChunjingXiao/SFTSeg

work [5], we transform the activity segmentation task into the classification problem by the following steps: (1) The continuous time series data is first discretized into equally sized windows according to the average activity length. (2) Each window is classified into one of the four state categories: *static-state*, *start-state*, *motion-state* and *end-state*. (3) Lastly, the start and end points of activities are detected according to these state labels. Here, the four state labels are defined as, *static-state*: the window is full of time-series data in the absence of activities, *start-state*: the window contains the start point of an activity, *motion-state*: the window is full of time-series data in the presence of activities, and *end-state*: the window contains the end point of an activity. Note that the window size should be far smaller than the activity length so that each window only contains one kind of state data. The examples of these four states extracted from an activity are visually shown in Fig. 1, which illustrates the first-order differences of WiFi Channel State Informatin (CSI) amplitudes of one subcarrier as a function of time. Here the dotted red lines are the real start and end points of the activity. As a result, the activity segmentation task becomes a classification problem of how to classify discretized data into the four states.

In this way, the segmentation results are heavily dependent on the state inference performance. Hence, the key problem of the activity segmentation becomes how to design an appropriate state inference model to predict state labels of discretized data from the target domain. Formally, assume that $D_{ls} = \{(x_{ls}^1, y^1), ..., (x_{ls}^{|D_{ls}|}, y^{|D_{ls}|})\}$ denotes the set of labeled source samples from the source domain, $D_{lt} = \{(x_{lt}^1, y^1), ..., (x_{lt}^{|D_{lt}|}, y^{|D_{lt}|})\}$ denotes the set of labeled target samples, $D_{ut} = \{x_{ut}^1, ..., x_{ut}^{|D_{ut}|}\}$ denotes the set of unlabeled target samples and $\mathcal{T}$ denotes the test set from the target domain, where $\mathcal{T} \cap D_{lt} = \emptyset$ and $\mathcal{T} \cap D_{ut} = \emptyset$. In practical applications, as the target data might be collected under different scenarios (e.g., persons, environments and sensor devices) from the source, their styles and characteristics can be quite different. Hence, the data distribution of D_{ls} is quite different from that of the target domain (e.g., D_{lt}, D_{ut} and $\mathcal{T}$). Due to the difficulty of collecting a large number of labeled target samples, we suppose it is feasible to gather several labeled target samples. Assuming that it is easy to collect a great many labeled source samples and unlabeled target samples, we have a large labeled source sample set D_{ls}, a limited labeled target sample set D_{lt} and a large unlabeled target sample set D_{ut}. The goal is to exploit these three kinds of data for training a classifier that can accurately infer the states of samples in $\mathcal{T}$.

2.2 Few-Shot Learning for State Inference

Few-shot learning aims at learning a classifier to predict classes of a set of unlabeled target samples (the query set) with only a small set of labeled target examples (the support set) from the target domain by reusing knowledge from existing models on relevant classes form the source domain [12], [24]. This problem is typically characterized as N_w way (number of categories in the query set) and N_s shot (number of labeled examples for each category). Different from traditional classification tasks, few-shot learning aims to classify novel classes after training. This requires that samples used for training and testing should come from disjoint label space.

Currently, significant progress has been made using different techniques for few-shot learning, including meta learning-based and metric learning-based methods. Among these approaches, metric-based methods have attracted remarkable attention because of their simplicity and effectiveness. This technique classifies samples in the query set based on the similarity between query samples and the support set. We realize metric-based few-shot learning by Siamese neural networks. A Siamese neural network consists of two twin networks, whose parameters are tied [25]. Each branch of the network uses the same building blocks, such as a Convolutional Neural Network (CNN). Generally, the Siamese network is trained by minimizing the contrastive loss based on the pairs of samples [26].

For state inference in activity segmentation, there is a extreme shift between the source and target domains, and both domains have the same category labels. Hence, the typical Siamese network should be improved to enhance the generalization ability to the target domain. Therefore, besides labeled source samples D_{ls}, we exploit labeled target samples D_{lt} and unlabeled target samples D_{ut} and propose new components to enable the model to obtain accurate feature representations of target samples and further enhance segmentation performance.

3 The Proposed Framework

In this section, we present the Self-supervised Few-shot Time-series Segmentation framework (SFTSeg). First, we illustrate an overview of the state inference model, SFTSeg. Next, we present how to boost the state inference model via consistency regularization based on labeled target data and via self-supervision based on unlabeled target data. Finally, we provide the steps of detecting the start and end points of activities according to the state labels.

3.1 SFTSeg Overview

To address the problems of subject and environment-dependence and shortage of labeled data, we introduce the few-shot learning model to forecast state labels of discretized data. The labels are further used to segment activities. However, different from general few-shot learning, in activity segmentation tasks there is a large shift between the source and target domains, and both domains have the same label space. Hence, we devise a Self-supervised Few-shot Time-series Segmentation framework, SFTSeg. The overall framework is described in Fig. 2.

In particular, as the source and target samples have the same categories, we first build a classification (cross-entropy) loss L_{cl} based on labeled source samples (x_{ls}^i, y^i). Second, as the model trained only based on the source data might not accurately capture characteristics of the target data, we develop a consistency regularization loss L_{cr} based on the limited labeled target samples (x_{lt}^i, y^i) to enhance model smoothness. For this purpose, we design a line-level data augmentation method, where labeled source sample x_{ls}^i is shrunk as the perturbation, which is injected into labeled target sample x_{lt}^i to generate augmented data $\hat{x}_{lt}^i$. Then they form sample pair $(x_{lt}^i, \hat{x}_{lt}^i)$ to build the loss L_{cr}.

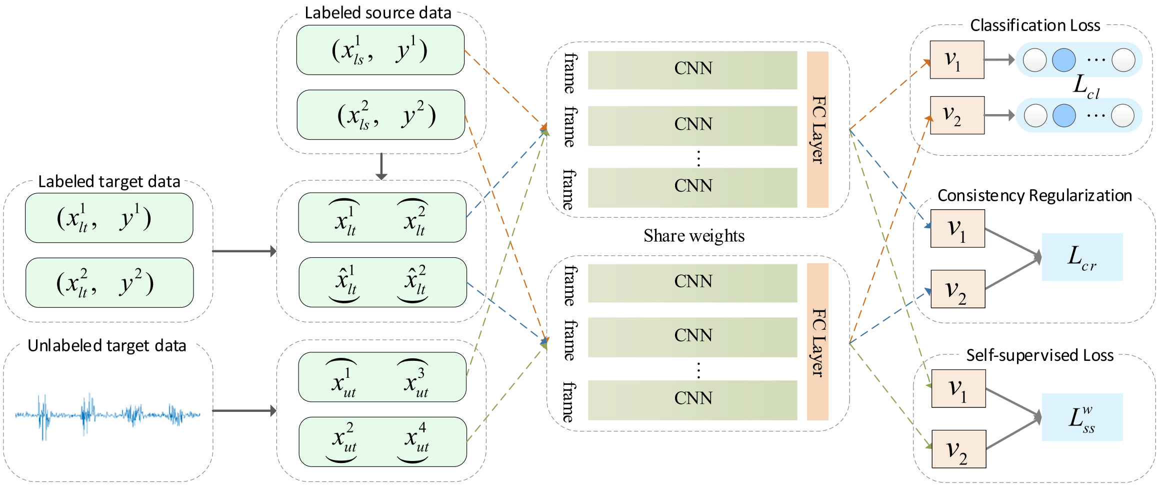


Fig. 2. SFTSeg Framework. Here (x_{ls}, y), (x_{lt}, y) and x_{ut} refer to the labeled source samples, labeled target samples and unlabeled target samples respectively. During the training process, (x_{ls}, y) is first used to build the classification loss L_{cl}. Second, x_{ls} is injected into x_{lt} to generate augmented sample $\hat{x}_{lt}$ using our designed line-level data augmentation method, and they further form sample pair $(x_{lt}^i, \hat{x}_{lt}^i)$ for constructing the consistency regularization loss L_{cr}. Third, (x_{ut}^i, x_{ut}^{i+1}) is extracted from unlabeled target samples to construct the self-supervised loss L_{ss}^w according to our designed time series-specific pretext task. In the testing process, the model takes testing samples and a few labeled target samples in each category as input, and outputs their feature representations. And then the state labels of testing samples are determined according to the similarity degree of their representations.

Third, to enhance generalization capacity to target data, we devise a time series-specific pretext task to extract sample pair (x_{ut}^i, x_{ut}^{i+1}) from unlabeled target data and build self-supervised loss L_{ss}^w. This loss is further enhanced by our proposed adaptive weighting.

3.2 Classification Loss on Labeled Source Data

Here we use the labeled source data to enhance the Siamese network model. Since labeled source data have the same category labels with the target data, we adopt the cross-entropy (classification) loss for model training.

Septically, different from typical few-shot learning tasks, for activity segmentation, both the source and target data have the same categories: *static-state*, *start-state*, *motion-state* and *end-state*. Hence, to take advantage of the labeled source data, we adopt the cross-entropy loss, rather than general losses of few-shot learning, to train the neural networks, which can enhance the classification capacity of the networks. Formally, let $D_{ls} = (x_{ls}^i, y^i)_{i=1}^N$ be the labeled source datasets, where y^i is the state label of x_{ls}^i. The classifier f is a function that maps the input feature space to the label space. Taking all of the labeled source data D_{ls} into account, the cross-entropy (classification) loss is as follows:

$$L_{cl} = -\frac{1}{|D_{ls}|}\sum_{i=1}^{|D_{ls}|}\sum_{j=1}^{C} y^{ij}\log f_j(x^i;\theta), \tag{1}$$

where θ refers to the model parameters, y^{ij} denotes the jth element of the one-hot encoded label of the sample x^i, and f_j is the jth element of f.

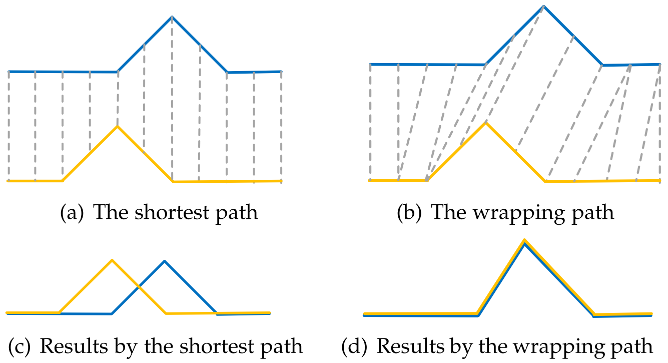
3.3 Consistency Regularization on Labeled Target Data

Due to the large shift between the source and target domains, the model trained only based on the source data may not fully capture characteristics of the target data, and correspondingly might not efficiently predict state labels of the target data. Hence, here we introduce consistency regularization to exploit the limited labeled target data to enhance generalization capacity of the model. In other words, we design a line-level data augmentation method for time-series data, which will generate augmented data for building a consistency regularization loss. This loss forces the augmented data and original data to be assigned the same label distribution to enhance model smoothness.

Consistency regularization aims to ensure that the classifier outputs the same class distribution for an example, even it is augmented by injecting semantic-preserving perturbations, which recently achieves remarkable success in image processing [27], [28] and natural language processing domains [16], [29]. Though widely used perturbation methods for consistency regularization, such as Gaussian noise and dropout noise, can provide supplementary superiority for image and natural language data, the effect is limited due to the intrinsic natures of time-series data. For example, perturbation methods for images mainly aim to generate pixel-level changes while time-series data needs line-level variations because time-series data are a kind of waveform along with time. Besides, the augmented data should be like real one with the style of the target data, which can benefit the inference model in learning characteristics of the target domain.

Hence, instead of injecting Gaussian noises like image processing, we shrink the real data from the source domain as the perturbation, and inject this perturbation into the labeled data from the target domain as augmented data. Then, the augmented and original samples are adopted to train the model by minimizing the mean-square error between their corresponding features.

Specifically, to yield proper augmented data, we process the labeled target sample following two rules: (i) The labeled source sample that will be shrunk as a perturbation should have the same category with the target sample. The reason is that a source sample with a different category might distort the waveform and undermine identification


(a) The shortest path

(b) The wrapping path

(c) Results by the shortest path

(d) Results by the wrapping path

Fig. 3. The difference between the wrapping & shortest path.

information of the augmented sample. (ii) the shrunk sample (perturbation) should be injected into the target sample based on a warping path instead of the default shortest path. Here, a warping path, produced using Dynamic Time Warping (DTW) [17], [18], maps the elements of two data sequences to minimize the distance between them. An example of the shortest path and warping path for two time-series samples is presented in Fig. 3. Here, the blue and orange line refer to waveforms of the two time-series samples, and the dotted gray lines represent the shortest path in Fig. 3a and wrapping path in Fig. 3b. When regarding the blue one as a perturbation, if adding this perturbation into the orange one by the shortest path, the waveform will be dramatically distorted, as shown in Fig. 3c. While, the result by the wrapping path in Fig. 3d can not only keep the basic shape, but also change the waveform a little.

Following this idea, the augmented sample $\hat{x}^i_{lt}$ is computed as follows:

$$\hat{x}^i_{lt} = Aggregate(x^i_{lt}, h(x^i_{ls})), \tag{2}$$

Where $Aggregate(x, x')$ sums the two sequence based on the warping path and $h(x)$ is a function to shrink the amplitude of the sensor data, e.g., $h(x) = \gamma * x$. Here $\gamma \in (0, 1)$ is a hyper-parameter to adjust the shrinking degree.

As a result, to penalize inconsistent predictions of original sample x^i_{lt} and augmented sample $\hat{x}^i_{lt}$, the loss about consistency regularization is defined by

$$L_{cr} = \sum_{i=1}^{|D_{lt}|} \|(f(x^i_{lt}) - f(\hat{x}^i_{lt}))\|^2, \tag{3}$$

where $f(x)$ refers to the feature vector outputted by the neural network and D_{lt} is the labeled target data set from the target domain.

3.4 Self-Supervision on Unlabeled Target Data

To enable the inference model to capture characteristics of the target data, here we explore a great deal of unlabeled target data for model training by incorporating the self-supervised technique into few-shot learning. For this purpose, we propose a time series-specific pretext task to build a self-supervised loss for time series data, and further enhance this loss by designing an adaptive weighting to adjust the importance of training data.

Self-Supervised Loss. To take advantage of unlabeled target data, we need to design an annotation-free pretext task for model training. Although there are a few effective pretext tasks in computer vision and natural language processing domains, such as image rotation, distortion and cropping [19], they are not applicable in continuous time-series data. For example, the image rotation task, widely used in computer vision, tries to assign the same label to both rotated image and original image. However, for activity segmentation, when the sequence with the *start-state* label is rotated by 180°, it can be easily regarded as the *end-state*, as shown in Fig. 1.

To this end, we propose a time series-specific pretext task, which builds many positive sample pairs and negative sample pairs based on unlabeled target data to train the neural network. Here, the positive pairs mean that both samples have the same state label, and the reverse is true for the negative pairs. Inspired by the work [30], we consider the two consecutive windows with the similar shape as the positive pairs and the two separated windows with the different shape as the negative pairs.

Specifically, we adopt a sliding window to discretize the time series into overlapping windows in a fixed size w, where the sliding step is l. The two windows are regarded as the positive pair if they adhere to the two constraints: (i) these two windows are adjacent; (ii) they contain the same number of change points, and the difference of both windows does not contain any change point. Correspondingly, two windows are regarded as the negative pair if they adhere to the two constraints: (i) these two windows are sufficiently separated from each other, i.e., their interval should be bigger than a given minimum temporal distance (such as $2 * w$); (ii) they contain the different number of change points, i.e., one window contains a change point and another has no change points. Here, the change point is a time point at which the behavior of a time series changes abruptly [31]. In time series data, activity breakpoints or transitions can be regarded as change points [10], [32]. Hence, if the two consecutive windows include the same number of change points, they should have the same state label and be regarded as the positive pair. And the two windows with the different number of change points should have different state labels and be regarded as the negative pair. We adopt a density ratio-based method, SEP [33], to detect change points. This approach employs the probability metrics and sensitive change scores to determine change points by comparing this score with a threshold value and achieves expected performance.

An example for exacting positive and negative sample pairs is presented in Fig. 4, where the dotted red lines are the real start and end points of an activity. As shown in this figure, the sample pair (x^1_{ut}, x^2_{ut}) is the positive one because they are two adjacent windows and both them have one change point, as well as the two differences, $\bar{x}^1_{ut} = x^1_{ut} - x^2_{ut}$ and $\bar{x}^2_{ut} = x^2_{ut} - x^1_{ut}$, have not any change point. Note that in this sample the dotted red lines are the change points as they are the transition points of activities. Correspondingly, the sample pair (x^1_{ut}, x^3_{ut}) is the negative one because they are well separated and have different number of change points.

Following these rules, a large number of positive pairs and negative pairs can be obtained from the unlabeled target samples. To improve the sample quality, we further

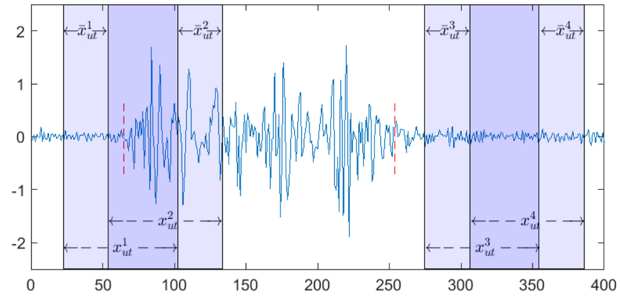


Fig. 4. Positive (negative) sample pairs extracted from activity data. Here, (x^1_{ut}, x^2_{ut}) is a positive pair because they are two adjacent windows and have the same number of change points. (x^1_{ut}, x^3_{ut}) is a negative pair because they are well separated and have different number of change points.

dropout the samples with low confidence. Specifically, for the positive pairs, a greater SEP score suggests that the probability of existing change points is bigger [33]. Since the difference sets of sample pairs, $\bar{x}^1_{t_u}$ and $\bar{x}^2_{t_u}$, are supposed to have no change points, we dropout the sample pairs with higher SEP scores in the difference sets. To this end, we first calculate the SEP scores of the difference sets of a sample pair, and then filter sample pairs according to their scores. For a difference set of a sample pair, we split this set into two parts evenly, x_{t-1} and x_t, each with length s, and then compute their density ratio as follows:

$$g_t(x) = \frac{f_{t-1}(x)}{f_t(x)}, \tag{4}$$

where $f_{t-1}(x)$ and $f_t(x)$ correspond to estimated probability densities of the two parts respectively. Then the SEP change point score is constructed as follows:

$$\hat{SEP} = \text{Max}\left(0, \quad \frac{1}{2} - \frac{1}{s}\sum_{i=1}^{s} g_t(x^i)\right), \tag{5}$$

In this way, the SEP scores of the difference sets $\bar{x}^j_{t_u}$ and $\bar{x}^{j+1}_{t_u}$, $\hat{SEP}_{\bar{x}_j}$ and $\hat{SEP}_{\bar{x}_{j+1}}$, can be computed. Then, to ensure the quality of these training samples and avoid over fitting, we dropout 10% of these positive sample pairs:

$$f_{drop} = \begin{cases} 1 & \hat{SEP}_{ave} > \varepsilon \\ 0 & otherwise \end{cases}, \tag{6}$$

where f_{drop} refers to the dropout condition, $\hat{SEP}_{ave} = (\hat{SEP}_{\bar{x}_j} + \hat{SEP}_{\bar{x}_{j+1}})/2$ is the average of both scores, and ε is a threshold determined by the SEP score ranking result of all the positive sample pairs and the dropout rate.

For the negative sample pairs, we expect that one sample has a change point while another has no change points. To meet this requirement, we filter sample pairs which have a smaller dissimilarity degree on change points. To this end, we design a dissimilarity score based on SEP scores to dropout negative sample pairs with lower confidence. In particular, for each sample from a negative pair, it is divided into h disjoint parts and then the SEP score of the two consecutive parts can be computed using Eq (5). Let $\hat{SEP}_j$ denote the SEP score of the jth and (j+1)-th parts. Then the maximum SEP score is:

$$\hat{SEP}_{max} = \max_{j\in[1,h-1]}\{\hat{SEP}_j\}. \tag{7}$$

As a result, each sample from a negative pair can be assigned a maximum SEP score, $\hat{SEP}_{max}$. Assume that $\hat{SEP}^{one}_{max}$ and $\hat{SEP}^{zero}_{max}$ are the maximum SEP scores of the samples with one and zero change points, respectively. Then, the dissimilarity score of this sample pair is computed as:

$$\hat{SEP}_{dis} = |\hat{SEP}^{one}_{max} - \hat{SEP}^{zero}_{max}|. \tag{8}$$

Then the negative sample pairs with lower dissimilarity scores will be eliminated, where Eq. (6) is still adopted as the dropout condition except replacing $\hat{SEP}_{ave}$ with $\hat{SEP}_{dis}$.

After dropping 10% positive and negative sample pairs with lower confidence, the remaining sample pairs is used to train the neural network by the following self-supervised loss:

$$L_{ss} = \frac{1}{2}\left((1-y).(d_e)^2 + y.(\max(0, m - d_e))^2\right), \tag{9}$$

where d_e is the euclidean distance between the representations $f(x^1_{t_u})$ and $f(x^2_{t_u})$ of a pair of input samples x^1_{ut} and x^2_{ut}, i.e., $d_e = ||f(x^1_{ut}) - f(x^2_{ut})||$. And y is a binary label assigned to this pair, i.e., $y = 0$ if x^1_{ut} and x^2_{ut} have the same state label, and y = 1 otherwise; and m is a hyper-parameter about the margin.

Adaptive Weighting. After filtering the sample pairs, the remaining positive sample pairs should have high confidence in belonging to the same state category, and the negative pairs to the different state categories. However, different sample pairs might provide various clues for learning data representation. Generally, the sample pairs without any activity data should contain less clues for learning data representation. Correspondingly, the ones including activity data should provide more clues and play a more important role for model training. An example about two kinds of sample pairs is shown in Fig. 4. In this figure, the positive sample pair (x^1_{ut}, x^2_{ut}) contains activity data and the pair (x^3_{ut}, x^4_{ut}) do not include activity data. Hence, the former is worth paying more attention during model training.

Since fluctuation ranges of amplitudes for sensor data when activities occur are much larger than that when no activity presents [2], the sample pairs with bigger fluctuation ranges probably contain activity data and should be paid more attention in model training as they are supposed to be more reliable. Hence, we adopt the amplitude variance of sample pairs to reflect fluctuation levels, which are regarded as weights to adjust their importance. Thus, the fluctuation level of a sample pair is computed as follow:

$$V_{pair} = \frac{1}{2}(V^1_{ut} + V^2_{ut}), \tag{10}$$

where V^1_{ut} and V^2_{ut} are the fluctuation variances of the sample pair, x^1_{ut} and x^2_{ut}, individually. Then V_{pair} is regarded as the weight of this pair to adjust its importance during model training. Taking this weight into account, the self-supervised loss in Eq. (9) becomes:

$$L^w_{ss} = \frac{1}{2}\left((1-y).V_{pair}.(d_e)^2 + y.(\max(0, m - V_{pair}.d_e))^2\right) \tag{11}$$

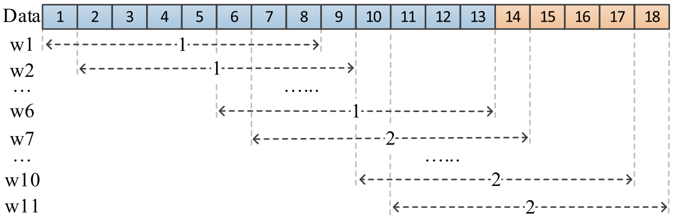


Fig. 5. An example about how to detect the start point.

Finally, after combining the classification loss in Eq. (1), consistency loss in Eq. (3), and self-supervised loss with weights in Eq. (11), the final loss function is illustrated as follows:

$$L_s = L_{ce} + L_{cr} + L_{ss}^{w}. \tag{12}$$

In this loss, the consistency regularization loss and self-supervised loss are based on the labeled target data and unlabeled target data from the target domain. Hence, the model trained using these losses can effectively capture the characteristics of the target data. The Adam algorithm [34] is adopted as the optimization method with the default hyper-parameters.

3.5 Activity Segmentation

After obtaining the trained state inference model, we predict the state label of a given sequence from the target domain, i.e., we compare the euclidean distance between the test sample and labeled target samples in each class, so that the minimum one indicates the predicted result. Then activities can be segmented according to the inferred state labels. In particular, similar to our previous work [5], we detect the start and end points of activities based on the following way. First, the continuous data streams are split into overlapping windows using a sliding window, each with length w, where the sliding step is 1. Second, the state label of each window is inferred using the state inference model. Finally, according to the state labels of windows, the start and end points of activities are identified by observing the changes of the mode of a set of window labels. Here, the mode is the number which appears most often in the set. In other words, if the mode changes from 1 (*static-state*) to 2 (*start-state*), the corresponding window is regarded as the start of an activity. If it changes from 4 (*end-state*) to 1 (*static-state*), this window is regarded as the end.

For visual inspection, Fig. 5 presents an example about how to detect the start point by observing the change of the mode, where the length of the list of window labels for mode calculation, m, is set to 10. In this figure, there are 18 data points in this sequence, and the points from 1 to 13 are the static data and the points from 14 to 18 are the motion data. To detect the start point, first, this sequence is split into 11 overlapping windows, each with length $w = 8$. Second, for each window, the trained state inference model is used to infer its state label. The state labels from w1 to w6 are 1 (*static-state*), and the ones from w7 to w11 are the 2 (*start-state*). Finally, each window is traversed to detect the start and end points of activities. When checking w10, the mode of the list of state labels from w1 to w10 is 1. When checking w11 where the index of the current data point i is 18, the mode from w2 to w11 becomes 2, which means that the mode changes from 1 (*static-state*) to 2 (*start-state*) and there is a start point. Note that when there are multiple values occurring equally frequently, the mode is set to the biggest of those values. Here, the start point, t_{start}, is set to $i - m/2 + 1$. Hence, when $i = 18$ and $m = 10$, t_{start} is equal to the real start point, 14, in this example. After segmenting human activities, these data will be used for activity classification.

4 Experimental Evaluation

In this section, we evaluate the effectiveness of SFTSeg on four datasets under the scenarios with different sensor devices, users and environments. Also, the performance of activity recognition and contributions of different components are investigated.

4.1 Experiment Data and Setup

Experiment Data. We conduct the experiments on the four activity recognition data sets, which are collected using different kinds of sensors, such as WiFi devices, smartphones, and RFID tags.

- *HandGesture* [35]: This dataset includes twelve hand movement activities performed by two subjects and captured by inertial measurement units. These data are continuous.
- *USC-HAD* [36]: This dataset consists of twelve human activities that are each recorded separately across fourteen subjects using a 3-axis accelerometer and a 3-axis gyrometer. Activities are repeated five times for each subject. Since these data are non-continuous, to conduct segmentation, they are manually stitched randomly.
- *RFID* [37]: This experiment dataset involves six persons which each performs postures between a wall and an RFID antenna, where nine passive RFID tags are placed on the wall. RFID data is a non-continuous dataset and is still manually stitched randomly for our experiments.
- *WiFiAction* [5]: The dataset is composed of ten activities performed by five persons collected by a WiFi devices with the CSI collection tool [38]. These data are continuous.

Since the source data and target data might be collected under different sensor devices, persons, and environments, in the following experiments, one dataset is selected as the source data and others are regarded as the target data. We mainly select WiFiAction and HandGesture data as the source data since they have a large amount of data. Besides, by default, three labeled samples per category from the target data are chosen for model training, i.e, the following experiments are conducted under the scenario of three-shot learning. And for the data from the target domain, 80% of their data are selected as unlabeled data for model training and the rest 20% as test set.

Evaluation Metrics. Similar to the work [9], we evaluate our proposed SFTSeg model as well as the baseline models using the two metrics: (i) *F1-score*: the F1-score is the mean of the Precision and Recall. A predicted segmentation is regarded as a true positive when it is located within a specified time window of the ground truth boundaries and a false negative when it falls outside the time window of the ground truth boundaries. According to the sampling rate of sensors, the specified time window is set to 0.3 and 0.5 seconds for the WiFiAction and HandGesture datasets respectively, and 2 seconds for the other datasets. (ii) *RMSE*: The Root Mean Square Error (RMSE) is computed between the ground truth boundary time and its nearest estimated boundary time. The RMSE is then normalized into the range of [0, 1] by dividing it by the time-series duration. The F1-score takes predicted segments as a whole, and provides an intuitive information about how many predicted segments approach the actual ones. However, the F1-score metric depends upon the selection of a threshold value. If the lengths of two predicted segments exceed the given threshold, the F1-score cannot distinguish their performance difference because both of them will be regarded as the same (False Negatives) no matter how different these two segments are. The RMSE will address this problem given its continuous metric space ensures that the predicted segment, which is closer to the actual start and end points, acquires a superior estimate. Hence, we adopt both F1-score and RMSE as evaluation metrics, which can provide a more comprehensive view.

TABLE 1
Segmentation Results

	Method	HandGesture	USC-HAD	RFID	WiFiAction
F1-score	WiAG	0.6082	0.7263	0.7520	0.6891
	Wi-Multi	0.6195	0.7355	0.7623	0.7123
	AR1seg	0.6132	0.6994	0.6987	0.6012
	SEPseg	0.6202	0.6781	0.6823	0.6054
	IGTS	0.3825	0.7333	0.9554	0.5123
	FLOSS	0.5379	0.3733	0.4106	0.4345
	ESPRESSO	0.6209	0.7467	0.9378	0.7075
	DeepSeg-Hand	-	0.8456	0.8701	0.7944
	DeepSeg-WiFi	0.7592	0.8534	0.8837	-
	SFTSeg-Hand	-	0.8897	**0.9576**	**0.8097**
	SFTSeg-WiFi	**0.7778**	**0.9031**	0.9565	-
RMSE	WiAG	0.2835	0.2188	0.1893	0.2227
	Wi-Multi	0.2722	0.2032	0.1532	0.1961
	AR1seg	0.2884	0.2397	0.2192	0.2632
	SEPseg	0.2882	0.2393	0.2130	0.3060
	IGTS	0.4270	0.1939	0.0401	0.3238
	FLOSS	0.3166	0.3267	0.3969	0.3285
	ESPRESSO	0.2764	0.1933	0.0692	0.2095
	DeepSeg-Hand	-	0.1396	0.0781	0.1771
	DeepSeg-WiFi	0.2310	0.1357	0.0765	-
	SFTSeg-Hand	-	0.1082	0.0329	**0.1486**
	SFTSeg-WiFi	**0.2059**	**0.1035**	**0.0326**	-

SFTSeg-Hand and SFTSeg-WiFi refer to the results of SFTSeg when taking the HandGesture and WiFiAction dataset as the source data individually, and DeepSeg-Hand and DeepSeg-WiFi mean the same.

Implementation Details. The models are optimized by Adam with learning rate 0.001, and the mini-batch size of data is 32. We optimize shrinking degree γ for computing $h(x)$ in Eq. (2) by conducting a grid search on the values between 0.01 and 0.5. Too big or small values result in low performance, and we set $\gamma = 0.05$ because it can obtain a better result. Similarly, the margin m in Eq. (9), and the window size w are empirically set to 1 and 120 respectively. The CNN architecture in the Siamese network is the same to our previous work [5].

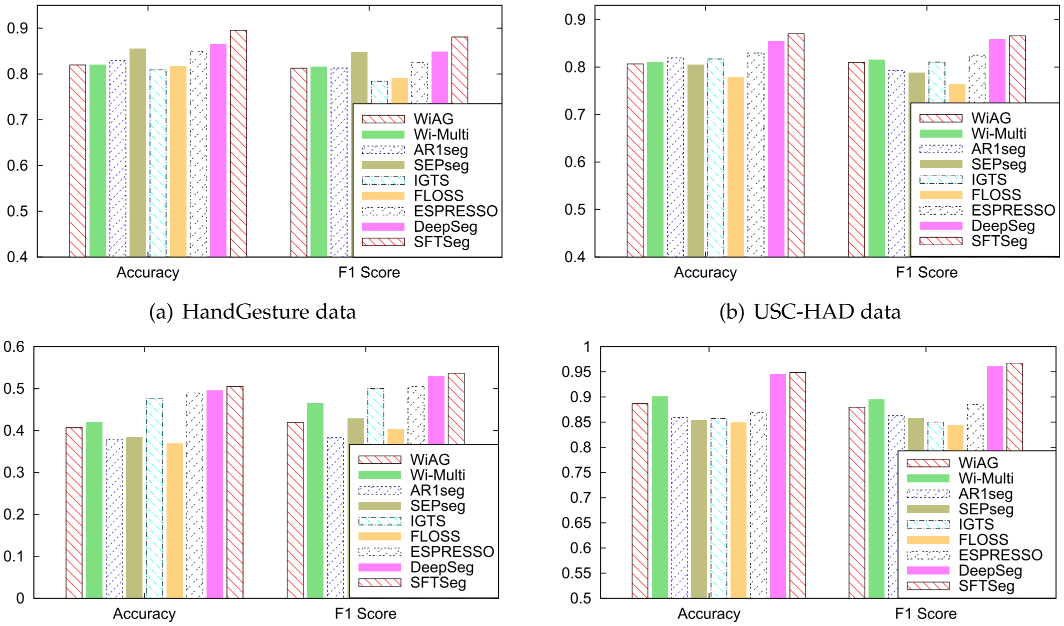
4.2 Baselines

To demonstrate the effectiveness and superiority of the proposed model, we select eight segmentation methods with widely used techniques as the baselines, including threshold-based [39], [40], CPD-based [41], [42], [43], temporal shape-based [4], [9], and supervised methods [5].

- *WiAG* [39]: A typical threshold-based segmentation method for gesture extraction. The method identifies the start and end of a gesture by comparing the amplitudes of the principal component streams with a given threshold.
- *Wi-Multi* [40]: A novel activity segmentation algorithm under noisy environment with multiple subjects. The algorithm can eliminate potential false detection by calculating the largest eigenvalues from amplitude and calibrated phase correlation matrices.
- *AR1seg* [41]: A typical change point detection method in the area of statistics. This approach estimates change-points using an autoregressive process of order 1.
- *SEPseg* [42]: An effective method to detect change points for time series data. This algorithm has been adopted to efficiently identify activity boundaries and recognizing human daily activities [10].
- *IGTS* [43]: an information gain based segmentation approach. This approach estimates action boundaries by using the dynamic programming approach to maximize the information gain of the constituent segments.
- *FLOSS* [4]: A shape-based segmentation method. This method segments activities based on the fact that patterns of similar shape should be each associated with the same segment class and occur within close temporal proximity of each other.
- *ESPRESSO* [9]: An entropy and shape aware time-series segmentation approach. This method exploits the entropy and temporal shape properties of time-series to conduct activity segmentation for multi-dimensional time-series.
- *DeepSeg* [5]: A supervised learning-based activity segmentation method. This framework adopts a CNN as the state inference model to predict state labels of discretized data and then identifies activity boundaries according to state labels.

4.3 Performance of Activity Segmentation

We report the performance of different segmentation approaches in Table 1, where the best is highlighted in bold. For the two supervised methods DeepSeg and SFTSeg, we present the results when taking the HandGesture/WiFiAction dataset as the source data, named as $\mathcal{X}$-Hand/$\mathcal{X}$-WiFi where $\mathcal{X}$ refers to the method DeepSeg/SFTSeg. And the results that the source and target data are from the same dataset are not shown as we mainly focus on the situation that they are from different distribution. The USC-HAD and RFID datasets are not taken as the source data for the



(a) HandGesture data

(b) USC-HAD data

(c) RFID data

(d) WiFiAction data

Fig. 6. The performance of activity recognition for the four datasets.

experiments because their sample numbers are relatively small, which cannot meet the assumption that there are a large number of source data. By analyzing the method performance, we have the following observations.

First, our proposed SFTSeg method consistently yields a better performance than previous segmentation approaches on all the datasets. Specifically, when regarding the WiFiAction dataset as the source data, SFTSeg-WiFi yields 2.45%, 5.82% and 8.23% improvement over the better baseline approach, DeepSeg-WiFi, in terms of F1-score on HandGesture, USC-HAD and RFID datasets, individually. Even compared to the optimal results selected from all the baselines for each dataset, our model still outperforms these optimal results. Similar trends can be found in the results when taking the HandGesture dataset as the source data. The results strongly indicate that our model can capture characteristics of the target data via our proposed consistency regularization and self-supervised loss, and further conduct accurately activity segmentation for the target data based on several labeled samples.

Second, the supervised method, DeepSeg, does not always exhibit significant superiority compared with unsupervised approaches, especially for RFID data that has a very small number of samples. The main reason is that DeepSeg is designed for the scenario that the source data and target data have the same distribution. However, in the case that there are only several labeled target samples, DeepSeg becomes less competitive. This explains the reason that most of works exploit the unsupervised way to segment activities when lacking labeled data. Whereas, SFTSeg can address the problem of limited labeled target data, and yields better performance than these supervised and unsupervised approaches.

Third, for unsupervised baselines, different datasets should adopt different unsupervised methods for obtaining better segmentation results. For example, for RFID data, IGTS outperforms other unsupervised methods, however, for HandGesture data, the segmentation result of IGTS is significantly worse than that of ESPRESSO. These results firmly support our mentioned description that the unsupervised segmentation approaches typically suffer from environment dependent problems. By contrast, our proposed SFTSeg can achieve consistently better performance on all the datasets.

4.4 Performance of Activity Recognition

We have demonstrated that SFTSeg can consistently obtain significant superiority about activity segmentation compared to the segmentation baselines. However, it is not clear how it benefits the final activity recognition. Here, we conduct the experiments to investigate activity classification performance when using our proposed SFTSeg and baseline segmentation methods to segment activities. A well-tuned CNN in the work [44] is used as the classification model. And accuracy and F1 score are regarded as the evaluation metrics. For each dataset, we randomly select 80% of their activity samples extracted using a given segmentation method as the training set, and the rest as the test set.

The results of activity classification are presented in Fig. 6. These figures clearly show that SFTSeg achieves better performance than all the baseline methods for the four datasets. For example, SFTSeg exhibits improvements of 3.03% and 8.62% compared to the best baseline and the worst baseline for HandGesture data, in terms of accuracy. This suggests that our proposed segmentation method can effectively benefit the activity recognition. Because it adopts the same activity classification model, the superiority of activity classification of our proposed SFTSeg should primarily be attributed to the segmentation results. Hence these results firmly prove our model's effectiveness, which can capture the characteristics of the target data and accurately segment activities.

TABLE 2
The Segmentation Performance When Incorporating Different Components

Method		WiFiAction as source data			HandGesture as source data		
		HandGesture	USC-HAD	RFID	WiFiAction	USC-HAD	RFID
F1-score	SFTSeg-Base	0.7429	0.8312	0.8795	0.7297	0.8233	0.8772
	SFTSeg-Consis	0.7478	0.8367	0.8971	0.7385	0.8356	0.8911
	SFTSeg-Self	0.7653	0.8633	0.9127	0.7533	0.8698	0.9268
	SFTSeg-Weight	0.7695	0.8812	0.9365	0.7817	0.8765	0.9331
	SFTSeg-Full	**0.7778**	**0.9031**	**0.9565**	**0.8097**	**0.8897**	**0.9536**
RMSE	SFTSeg-Base	0.2359	0.1608	0.0683	0.1686	0.1635	0.0705
	SFTSeg-Consis	0.2331	0.1139	0.0635	0.1669	0.1228	0.0698
	SFTSeg-Self	0.2159	0.1076	0.0438	0.1528	0.1168	0.0415
	SFTSeg-Weight	0.2094	0.1051	0.0371	0.1525	0.1133	0.0389
	SFTSeg-Full	**0.2059**	**0.1035**	**0.0326**	**0.1486**	**0.1082**	**0.0369**

4.5 Ablation Study

Here, we focus on studying the contributions of our designed essential components in the SFTSeg model, i.e. the consistency regularization loss, self-supervised loss and adaptive weighting. We investigate the role of taking into account different components: (i) **SFTSeg-Base** is the basic Siamese network model by optimizing the classification loss based on the labeled source data as given in Eq. (1). (ii) **SFTSeg-Consis** is the Siamese network model with our designed consistency regularization loss as shown in Eq. (3). (iii) **SFTSeg-Self** is the Siamese network model with the self-supervised loss but without the adaptive weights as presented in Eq. (9). (iv) **SFTSeg-Weight** is the Siamese network model with the self-supervised loss and adaptive weights as illustrated in Eq. (11). (v) **SFTSeg-Full** is our proposed model fully incorporating all of the components.

The segmentation results when taking WiFiAction/HandGesture as the source data are shown in Table 2 with the best results highlighted in boldface. We summarize the observations reported in this table as follows. First, *SFT-Seg-Full* achieves the highest performance. Meanwhile, *SFT-Seg-Base* performs the worst, which indicates that the efficiency of the main components we designed can substantially enhance the segmentation performance. Second, when incorporating the consistency regularization term, *SFT-Seg-Consis* obtains better results than *SFT-Seg-Base*. This is because that the limited labeled samples from the target domain are augmented by our designed method, which can benefit the model in improving generalization capacity on the target domain. Third, both *SFT-Seg-Self* and *SFT-Seg-Weight* outperform *SFT-Seg-Base* by a certain margin. The results prove the main motivation of our model, i.e., introducing the self-supervised loss with adaptive weights based on the unlabeled target data can enable the model to capture characteristics of the target domain and further significantly enhance segmentation performance.

4.6 Role of the Target Data Size

Our proposed model tries to take advantage of target data to alleviate the shift between the source and target domains. Hence, here we explore the role of the target data size on the segmentation performance. In particular, we investigate the results with one-shot, two-shot, and three-shot when the number of the unlabeled target data varies.

Fig. 7 illustrates the results about F1 score of using the WiFiAction dataset as the source data with different proportions of all the unlabeled target data. The results of using HandGesture as the source data and RMSE are not shown for they have the same trend. The figures show that SFTSeg acquires rising segmentation F1 score with the increase of the unlabeled data size for all the datasets. This suggests that the unlabeled data size plays a vital role on segmentation performance, and our designed Self-supervised pretext task can adopt unlabeled target data for enhancing model training. Meanwhile, our model requires a certain number of unlabeled data, such as 50% of all the unlabeled data for USC-HAD data, to obtain expected performance as many data does. However, it is easily affordable to collect unlabeled data from the target scenarios. Besides, the performance of three-shot is obviously higher than that of one-shot. The reason is that more labeled target samples can not only benefit consistency regularization in the training stage, but also benefit distance computation between the test sample and labeled target samples in the test stage. As a whole, the above results indicate that our model can effectively exploit labeled and unlabeled target data to improve the segmentation performance.

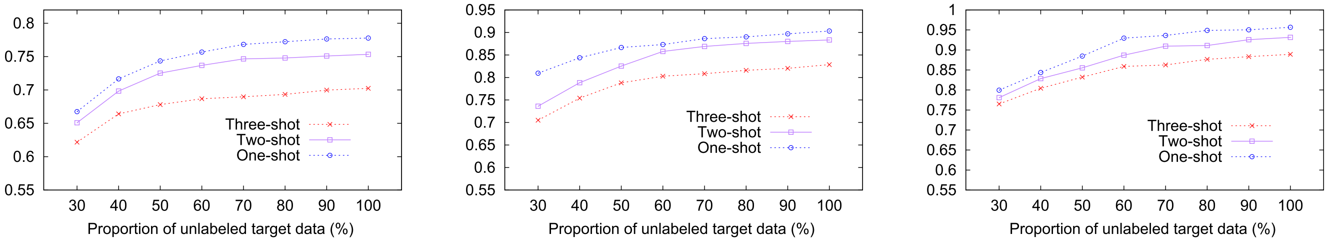
5 Related Work

This work is mainly related to two research areas: time-series segmentation and few-shot learning. Here, we will present an overview of the most closely related works in each area, and highlight the major differences between our study and these works.

5.1 Time-Series Segmentation

As a vital step for activity recognition, time-series segmentation has attracted considerable attention in the field of activity recognition. The studies on activity segmentation fall into two categories: supervised and unsupervised methods.

Supervised Segmentation Approaches. Supervised learning-based segmentation approaches mainly adopt classification models to identify transitions between activities or measure efficiency of data segmentation. For example, Martindale et al. [6] proposed a multi-task recurrent neural network



(a) HandGesture data

(b) USC-HAD data

(c) RFID data

Fig. 7. The segmentation F1 score with different size of target data using the WiFiAction dataset as the source data.

architecture which simultaneously segment and recognize activities and cycle phases for inertial sensor data. San-Segundo et al. [45] developed a human activity recognition and segmentation system using Hidden Markov Models to model inertial signals from a smartphone. Ma et al. [46] designed an adaptive sliding window algorithm which adopt a classifier to evaluate the efficiency of segmentation results and further expand or contract boundaries of activities to accurately extract the time windows of activities. Noor et al. [47] presented a novel adaptive sliding window technique for activity segmentation based on tri-axial accelerometer signals. In this method, the window size is adaptively adjusted based on signal information generated by a classifier to acquire the more effective window segmentation. Xiao et al. [5] proposed to adopt a CNN to infer the states of a given pieces of sensor data which indicates whether this piece includes a start point or end point. Then, the exact start and end points are determined according to piece states.

Unsupervised Segmentation Approaches. As it is costly and time-consuming to obtain labeled target samples for sensor-based activities, most works focus on unsupervised approaches, including CPD-based segmentation, threshold-based segmentation and temporal shape-based segmentation approaches.

CPD-based approaches regard activity breakpoints or transitions in time-series data as change points, and detect activity boundaries by finding these change points [7]. Due to the real-time requirement, only a few CPD techniques are appropriate for segmenting activities [10]. In particular, Alam et al. [48] applied the Relative unconstrained Least-Squares Importance Fitting (RuLSIF) algorithm to accelerometer data to detect gesture changes. Sadri [43] proposed an Information Gain-based Temporal Segmentation method (IGTS) to find the transition times in human activities and daily routines. Chakar et al. [41] proposed an AR1seg approach for estimating change-points using the AR(1) model based on the assumption of a common autocorrelation coefficient for all segments. Also, embedding models is used to represent sensor events for detecting activity transitions and conducting activity segmentation [32], and Hybrid fuzzy c-means is applied to change point detection for multi-resident activity segmentation [49]. Recently, density ratio-based techniques are enhanced to conduct change point detection by employing new probability metrics and more sensitive change scores, and change points are determined by comparing the score with a threshold value [10], [42].

Threshold-based approaches adopt thresholds to identify start and end times of activities [2], [8] as the statistical properties such as variance of sense data amplitudes in the presence of activities is much bigger than the one in the absence of activities. Some works pursue selecting a fixed threshold to segment activities. For example, Sheng et al. [50] presented a segmentation algorithm, which adopts their designed mean absolute differences and the pre-set start and end thresholds to identify activities. Bu et al. [51] adopted an optimal threshold calculated based on training set to detect boundaries of activities. Wang et al. [52] presented a two-phase segmentation algorithm, which first detects whether activities occur and then locates starting points of activity data. Meanwhile other research utilizes a dynamic threshold that can be changed based on experiment settings. For instance, Feng et al. [40] designed a two-step segmentation algorithm, where the slot length for variance computation changes for different scenarios. Wang et al. [53] introduced two thresholds to distinguish activity boundaries. They adopted an amplitude threshold to find potential activities, and a duration threshold to eliminate a short wave burst. Yan et al. [54] proposed an adaptive activity cutting algorithm, which updates the threshold based on the changes of waveforms in the current sliding window. Wang et al. [55] presented a dynamic thresholding algorithm for activity segmentation, where thresholds change along with the noise level.

Temporal shape-based approaches exploit the change of the temporal shape patterns in the stream data to detect the start and end points of activities. For instance, Gharghabi et al. [4] presented a Fast Low-Cost Semantic Segmentation (FLOSS) based on the finding that patterns of similar shape are each associated with the same segment class and occur within close temporal proximity of each other. The authors proposed a pattern-based primitive to discover a chain of similar patterns [56], [57]. Recently, Deldari et al. [9] proposed an entropy and shape aware time-series segmentation approach (ESPRESSO), which exploits the entropy and temporal shape properties of time-series to conduct activity segmentation for multi-dimensional time-series.

Although the above supervised and unsupervised methods have achieved great success in many segmentation applications, there exist limitations when applying to practical scenarios. The supervised methods require a lot of labeled samples from the target domain, which is difficult to be obtained, especially for terminal users who buys devices for activity detection. The unsupervised approaches generally do not require labeled data, however, these methods are generally subjective and experiment dependent. For example, seeking optimal settings for CPD techniques is hard since the values may be application dependent and change over time [7]. The threshold-based methods and shape-based approaches also requires having detailed knowledge about the particular problem to determine the optimal thresholds and parameters [5], [9]. Here, we introduce few-shot learning to conduct activity segmentation, which can alleviate the difficulty of obtaining a lot of labeled data and reduce the dependence on experiences. And we also design the consistency regularization loss, self-supervised loss and adaptive weighting for time series data to enhance the few-show model.

5.2 Few-Shot Learning

Few-shot learning aims to learn representations that generalize well to the novel classes where only few samples are available. Few-shot learning has been quickly adopted in the field of activity recognition and image classification.

Activity Recognition. Recently, few-shot learning has been introduced to address the problem of limited labeled data for activity recognition. For instance, Ding et al. [58] leveraged the one-shot learning strategy to realize adaptive initial state sensing, which can obviate the need for the collection of massive samples with different initial poses. Gong et al. [59] designed a meta-learning framework that can leverage seen conditions in training data to adapt to an unseen condition based on a few labeled samples. Wang et al. [60] proposed an augment few shot learning-based human activity recognition framework, which can fine-tune the model parameters using a small amount of samples to recognize new categories. Ding et al. [61] proposed a meta-learning framework for one-shot activity recognition using Radio-Frequency (RF) signals. This framework involves a parametric RF-specific module for training a powerful distance metric, and a dual-path base network to exploit high-dimensional features in the RF signal matrix. Shi et al. [62] designed a human activity recognition scheme using matching network, which can perform one-short learning to recognize activities in a new environment. Yang et al. [63] developed an intelligent few-shot learning model based on Siamese networks for Internet of Things applications, which adopts two self-attention models to extract the sentiment features and use Mahalanobis distance to measure the similarity between the feature vectors of different categories. Feng et al. [64] proposed a few-shot human activity recognition method which leverages a deep learning model for feature extraction and classification and implements knowledge transfer in the manner of model parameter transfer. Yang et al. [65] presented a deep Siamese representation learning architecture for one-shot gesture recognition. This model extends the capacity of spatio-temporal pattern learning by incorporating convolutional and bidirectional recurrent neural networks, and ameliorates the transferable pairwise loss of the Siamese framework.

Few-Shot Classification. Most existing studies about few shot classification focus on the image processing, which aims to classify images with a limited number of labeled examples in each class. For example, Gidaris et al. [66] adopted the self-supervised technique as a pretext task for the few-shot framework, which can benefit extracting informative and transferable feature representations but only adopting several labeled samples. Sun et al. [67] presented a novel semi-supervised meta-learning method, which explores unlabeled samples to meta-learn how to cherry-pick and annotate such unsupervised data for improving accuracy. Das et al. [68] proposed a few-shot learning framework using Mahalanobis distance. They first trained a multi-layer neural network using samples from the basic classes, and then classified samples by calculating the Mahalanobis distance to the average class representation. Ye et al. [69] proposed a task optimizer for adaptive initialization. This method integrates the task context when the model is initialized, which not only simplifies the optimization process of the model, but also effectively obtains the high-performance model parameters. Li et al. [70] developed an adaptive margin principle to improve the generalization ability of metric-based meta-learning approaches for few-shot learning problems. Guo et al. [24] proposed the broader study of cross-domain few-shot learning benchmark, consisting of image data from a diverse assortment of image acquisition methods.

Compared to the above works, we focus on the scenario that there is a larger drift between source and target domains on time series data. We introduce self-supervised learning into the Siamese framework and design a novel time series-specific pretext task with adaptive weighting to boost few-shot learning in capturing characteristics of the target domain. Also, we build a new consistency regularization for time series data to enhance model performance.

6 Conclusion

In this paper, we proposed SFTSeg, a Self-supervised Few-shot Time-series Segmentation framework for activity recognition on time-series data. In this framework, we explored few-shot learning to overcome the problems of subject and environment dependence and shortage of annotation data. And we designed a line-level data augmentation method for building a consistency regularization loss, and presented a time series-specific pretext task to build a self-supervised loss with adaptive weighting, which can boost few-shot learning in capturing characteristics of the target data, and further enhance segmentation performance. Based on four datasets from different sensor devices, experimental results illustrate that SFTSeg significantly outperforms the state-of-the-art baselines. In the future, we plan to enhance this model or explore other deep learning techniques, such as generative adversarial networks, to reduce the usage of labeled data. Also, we will apply this framework to activity recognition tasks and design a real-time activity recognition system for healthcare services.

References

[1] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, “Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities,” *ACM Comput. Surv.*, vol. 54, no. 4, pp. 1–40, 2021.

[2] Y. Ma, G. Zhou, and S. Wang, “WiFi sensing with channel state information: A survey,” *ACM Comput. Surv.*, vol. 52, no. 3, pp. 1–36, 2019.

[3] C. Truong, L. Oudre, and N. Vayatis, “Selective review of offline change point detection methods,” *Signal Process.*, vol. 167, pp. 1–20, 2020.

[4] S. Gharghabi et al., “Domain agnostic online semantic segmentation for multi-dimensional time series,” *Data Min. Knowl. Discov.*, vol. 33, no. 1, pp. 96–130, 2019.

[5] C. Xiao, Y. Lei, Y. Ma, F. Zhou, and Z. Qin, “DeepSeg: Deep-learning-based activity segmentation framework for activity recognition using WiFi,” *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5669–5681, Jul. 2021.

[6] C. F. Martindale, V. Christlein, P. Klumpp, and B. M. Eskofier, "Wearables-based multi-task gait and activity segmentation using recurrent neural networks," *Neurocomputing*, vol. 432, pp. 250–261, 2021.

[7] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowl. Informat. Syst.*, vol. 51, no. 2, pp. 339–367, 2017.

[8] L. Chen, X. Chen, L. Ni, Y. Peng, and D. Fang, "Human behavior recognition using Wi-Fi CSI: Challenges and opportunities," *IEEE Commun. Mag.*, vol. 55, no. 10, pp. 112–117, Oct. 2017.

[9] S. Deldari, D. V. Smith, A. Sadri, and F. Salim, "ESPRESSO: Entropy and shape aware time-series segmentation for processing heterogeneous sensor data," *Proc. ACM InterAct. Mobile Wearable Ubiquitous Technol.*, vol. 4, no. 3, 2020, Art. no. 77.

[10] S. Aminikhanghahi and D. Cook, "Enhancing activity recognition using CPD-based activity segmentation," *Pervasive Mobile Comput.*, vol. 53, pp. 75–89, 2019.

[11] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, 2020, Art. no. 63.

[12] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. Wang, and J.-B. Huang, "A closer look at few-shot classification," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–16.

[13] Y. Guo et al., "A broader study of cross-domain few-shot learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 124–141.

[14] A. Mustafa and R. K. Mantiuk, "Transformation consistency regularization–A semi-supervised paradigm for image-to-image translation," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 599–615.

[15] Q. Zhou et al., "Uncertainty-aware consistency regularization for cross-domain semantic segmentation," *Compt. Vis. Image Understanding*, vol. 221, p. 103448, 2022.

[16] B. Zheng et al., "Consistency regularization for cross-lingual fine-tuning," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2021, pp. 3403–3417.

[17] Z. Zhang, R. Tavenard, A. Bailly, X. Tang, P. Tang, and T. Corpetti, "Dynamic time warping under limited warping path length," *Informat. Sci.*, vol. 393, pp. 91–107, 2017.

[18] T.-T.-H. Phan, E. P. Caillault, A. Lefebvre, and A. Bigand, "Dynamic time warping-based imputation for univariate time series data," *Pattern Recognit. Lett.*, vol. 139, pp. 139–147, 2020.

[19] X. Liu et al., "Self-supervised learning: Generative or contrastive," *IEEE Trans. Knowl. Data Eng.*, early access, Jun. 22, 2021, doi: 10.1109/TKDE.2021.3090866.

[20] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1422–1430.

[21] L. Zhang, G.-J. Qi, L. Wang, and J. Luo, "AET vs. AED: Unsupervised representation learning by auto-encoding transformations rather than data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2547–2555.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 4171–4186.

[23] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. Int. Conf. Learn. Representations*, 2018.

[24] Y. Guo et al., "A broader study of cross-domain few-shot learning," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 124–141.

[25] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proc. Int. Conf. Mach. Learn. Deep Learn. Workshop*, 2015, pp. 1–8.

[26] K. Musgrave, S. Belongie, and S.-N. Lim, "A metric learning reality check," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 681–699.

[27] A. Abuduweili, X. Li, H. Shi, C.-Z. Xu, and D. Dou, "Adaptive consistency regularization for semi-supervised transfer learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 6923–6932.

[28] H. Zhang, Z. Zhang, A. Odena, and H. Lee, "Consistency regularization for generative adversarial networks," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–19.

[29] H. Seo, L. Yu, H. Ren, X. Li, L. Shen, and L. Xing, "Deep neural network with consistency regularization of multi-output channels for improved tumor detection and delineation," *IEEE Trans. Med. Imag.*, vol. 40, no. 12, pp. 3369–3378, Dec. 2021.

[30] S. Deldari, D. V. Smith, H. Xue, and F. D. Salim, "Time-series change point detection with self-supervised contrastive predictive coding," in *Proc. Web Conf.*, 2021, pp. 3124–3135.

[31] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Process.*, vol. 167, 2020, Art. no. 107299.

[32] U. Bermejo, A. Almeida, A. Bilbao-Jayo, and G. Azkune, "Embedding-based real-time change point detection with application to activity segmentation in smart home time series data," *Expert Syst. Appl.*, vol. 185, 2021, Art. no. 115641.

[33] S. Aminikhanghahi, T. Wang, and D. J. Cook, "Real-time change point detection with application to smart home time series data," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 1010–1023, May 2019.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.

[35] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.s*, vol. 46, no. 3, pp. 1–33, 2014.

[36] M. Zhang and A. A. Sawchuk, "USC-HAD: A daily activity dataset for ubiquitous activity recognition using wearable sensors," in *Proc. ACM Conf. Ubiquitous Comput.*, 2012, pp. 1036–1043.

[37] L. Yao, Q. Z. Sheng, W. Ruan, X. Li, S. Wang, and Z. Yang, "Unobtrusive posture recognition via online learning of multidimensional RFID received signal strength," in *Proc. IEEE 21st Int. Conf. Parallel Distrib. Syst.*, 2015, pp. 116–123.

[38] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: Gathering 802.11n traces with channel state information," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 1, pp. 53–53, 2011.

[39] A. Virmani and M. Shahzad, "Position and orientation agnostic gesture recognition using WiFi," in *Proc. 15th Annu. Int. Conf. Mobile Syst. Appl. Serv.*, 2017, pp. 252–264.

[40] C. Feng, S. Arshad, S. Zhou, D. Cao, and Y. Liu, "Wi-multi: A three-phase system for multiple human activity recognition with commercial WiFi devices," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 7293–7304, Aug. 2019.

[41] S. Chakar, E. Lebarbier, C. Lvy-Leduc, and S. Robin, "A robust approach for estimating change-points in the mean of an $\mathrm{AR}(1)$ process," *Bernoulli*, vol. 23, no. 2, pp. 1408–1447, 05 2017.

[42] S. Aminikhanghahi, T. Wang, and D. J. Cook, "Real-time change point detection with application to smart home time series data," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 1010–1023, May 2019.

[43] A. Sadri, Y. Ren, and F. D. Salim, "Information gain-based metric for recognizing transitions in human activities," *Pervasive Mobile Comput.*, vol. 38, pp. 92–109, 2017.

[44] C. Xiao, D. Han, Y. Ma, and Z. Qin, "CsiGAN: Robust channel state information-based activity recognition with GANs," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 191–10 204, Jun. 2019.

[45] R. San-Segundo, J. Lorenzo-Trueba, B. Martinez-Gonzalez, and J. M. Pardo, "Segmenting human activities based on HMMs using smartphone inertial sensors," *Pervasive Mobile Comput.*, vol. 30, pp. 84–96, 2016.

[46] C. Ma, W. Li, J. Cao, J. Du, Q. Li, and R. Gravina, "Adaptive sliding window based activity recognition for assisted livings," *Informat. Fusion*, vol. 53, pp. 55–65, 2020.

[47] M. H. M. Noor, Z. Salcic, and K. I.-K. Wang, "Adaptive sliding window segmentation for physical activity recognition using a single TRI-axial accelerometer," *Pervasive Mobile Comput.*, vol. 38, pp. 41–59, 2017.

[48] M. A. U. Alam, N. Roy, A. Gangopadhyay, and E. Galik, "A smart segmentation technique towards improved infrequent non-speech gestural activity recognition model," *Pervasive Mobile Comput.*, vol. 34, pp. 25–45, 2017.

[49] D. Chen, S. Yongchareon, E. M. K. Lai, J. Yu, and Q. Z. Sheng, "Hybrid fuzzy C-means CPD-based segmentation for improving sensor-based multiresident activity recognition," *IEEE Internet Things J.*, vol. 8, no. 14, pp. 11 193–11 207, Jul. 2021.

[50] B. Sheng, F. Xiao, L. Sha, and L. Sun, "Deep spatial-temporal model based cross-scene action recognition using commodity WiFi," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3592–3601, Apr. 2020.

[51] Q. Bu, G. Yang, X. Ming, T. Zhang, J. Feng, and J. Zhang, "Deep transfer learning for gesture recognition with WiFi signals," *Pers. Ubiquitous Comput.*, no. 1, pp. 1617–4917, 2020.

[52] H. Wang, D. Zhang, Y. Wang, J. Ma, Y. Wang, and S. Li, "RT-fall: A real-time and contactless fall detection system with commodity WiFi devices," *IEEE Trans. Mobile Comput.*, vol. 16, no. 2, pp. 511–526, Feb. 2017.

[53] F. Wang, W. Gong, and J. Liu, "On spatial diversity in WiFi-based human activity recognition: A deep learning based approach," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2035–2047, Apr. 2019.

[54] H. Yan, Y. Zhang, Y. Wang, and K. Xu, "WiAct: A passive WiFi-based human activity recognition system," *IEEE Sensors J.*, vol. 20, no. 1, pp. 296–305, Jan. 2020.

[55] W. Wang, A. X. Liu, and M. Shahzad, "Gait recognition using WiFi signals," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2016, pp. 363–373.

[56] S. Wang, Y. Yuan, and H. Li, "Discovering all-chain set in streaming time series," in *Proc. Adv. Knowl. Discov. Data Mining*, 2019, pp. 306–318.

[57] Y. Zhu, M. Imamura, D. Nikovski, and E. Keogh, "Matrix profile VII: Time series chains: A new primitive for time series data mining," in *Proc. IEEE Int. Conf. Data Mining*, 2017, pp. 695–704.

[58] X. Ding, T. Jiang, Y. Zhong, S. Wu, J. Yang, and W. Xue, "Improving WiFi-based human activity recognition with adaptive initial state via one-shot learning," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2021, pp. 1–6.

[59] T. Gong, Y. Kim, R. Choi, J. Shin, and S.-J. Lee, "Adapting to unknown conditions in learning-based mobile sensing," *IEEE Trans. Mobile Comput.*, early access, Feb. 23, 2021, doi: 10.1109/TMC.2021.3061130.

[60] Y. Wang, L. Yao, Y. Wang, and Y. Zhang, "Robust CSI-based human activity recognition with augment few shot learning," *IEEE Sensors J.*, vol. 21, no. 21, pp. 24 297–24 308, Nov. 2021.

[61] S. Ding, Z. Chen, T. Zheng, and J. Luo, "RF-net: A unified meta-learning framework for RF-enabled one-shot human activity recognition," in *Proc. 18th Conf. Embedded Netw. Sensor Syst.*, 2020, pp. 517–530.

[62] Z. Shi, J. A. Zhang, Y. D. R. Xu, and Q. Cheng, "Environment-robust device-free human activity recognition with channel-state-information enhancement and one-shot learning," *IEEE Trans. Mobile Comput.*, vol. 21, no. 2, pp. 540–554, Feb. 2022.

[63] L. Yang, Y. Li, J. Wang, and N. N. Xiong, "FSLM: An intelligent few-shot learning model based on siamese networks for IoT technology," *IEEE Internet Things J.*, vol. 8, no. 12, pp. 9717–9729, Jun. 2021.

[64] S. Feng and M. F. Duarte, "Few-shot learning-based human activity recognition," *Expert Syst. Appl.*, vol. 138, 2019, Art. no. 112782.

[65] J. Yang, H. Zou, Y. Zhou, and L. Xie, "Learning gestures from WiFi: A siamese recurrent convolutional architecture," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10 763–10 772, Dec. 2019.

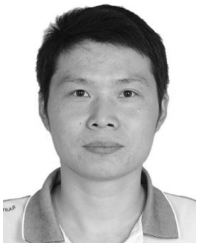
[66] S. Gidaris, A. Bursuc, N. Komodakis, P. Perez, and M. Cord, "Boosting few-shot visual learning with self-supervision," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 8058–8067.

[67] Q. Sun, X. Li, Y. Liu, S. Zheng, T.-S. Chua, and B. Schiele, "Learning to self-train for semi-supervised few-shot classification," in *Proc. Annu. Conf. Neural Inform. Process. Syst.*, 2019, pp. 10276–10286.

[68] D. Das and C. S. G. Lee, "A two-stage approach to few-shot learning for image recognition," *IEEE Trans. Image Process.*, vol. 29, pp. 3336–3350, 2020.

[69] H.-J. Ye, X.-R. Sheng, and D.-C. Zhan, "Few-shot learning with adaptively initialized task optimizer: A practical meta-learning approach," *Mach. Learn.*, vol. 109, no. 3, pp. 643–664, 2020.

[70] A. Li, W. Huang, X. Lan, J. Feng, Z. Li, and L. Wang, "Boosting few-shot learning with adaptive margin loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12573–12581.



Chunjing Xiao received the PhD degree from the University of Electronic Science and Technology of China, Chengdu, China. He is currently an Associate Professor with the School of Computer and Information Engineering, Henan University, Kaifeng, China. He was a visiting scholar with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL. His current research interests include recommender systems, representation learning, and Internet of Things.

Shiming Chen is currently working toward the master's degree with the School of Computer and Information Engineering, Henan University, Kaifeng, China. His research interests include deep learning, data analytics, and wireless sensing.

Fan Zhou (Member, IEEE) received the BS degree in computer science from Sichuan University, Chengdu, China, in 2003, and the MS and PhD degrees from the University of Electronic Science and Technology of China, Chengdu, in 2006 and 2012, respectively. He is currently an associate professor with the School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests include machine learning, neural networks, spatio-temporal data management, graph learning, recommender systems, and social network data mining.

Jie Wu (Fellow, IEEE) is the director of the Center for Networked Computing and Laura H. Carnell professor with Temple University. He also serves as the director of international affairs with the College of Science and Technology. He served as chair of Department of Computer and Information Sciences from the summer of 2009 to the summer of 2016 and associate vice provost for international affairs from the fall of 2015 to the summer of 2017. Prior to joining Temple University, he was a program director with the National Science Foundation and was a distinguished professor with Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Service Computing*, *Journal of Parallel and Distributed Computing*, and *Journal of Computer Science and Technology*. He was a general co-chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, ACM MobiHoc 2014, ICPP 2016, and IEEE CNS 2016, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). He is a fellow of the AAAS. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.