Self-Supervised Learning in Human Activity Recognition

Xiaobing Chen
Dec. 1, 2021

Outlines

1. Introduction

- Human Activity Recognition (HAR)
- Feature Extraction
- Self-Supervised Learning (SSL)

2. Self-supervised learning in HAR

- Five recent methods

3. Experiments

- Dataset Collection
- Implements
- Results

4. Conclusions and Discussion

1 Introduction

Introduction-Feature Extraction

- **Human Activity Recognition**: utilizing sensory data to recognize human activities
- Sensory data come from visual images/videos or **motion sensors (accelerometer, gyroscope)**
- **Common Activity Recognition Chain**: signal processing + feature representation + classifier
- Signal processing is done at data collection phase, like denoising.
- Classification performance highly depends on the **feature representation**.
- **Machine learning based classifiers** present the best performance, however, depending on more labelled data.
- **Main Feature Extraction Techniques**:
- Statistical features: based on heuristics, using mean, standard deviation, frequencies.
- **Learned features**: that derive representations directly from raw sensor data themselves

How to learn good features using unlabeled data?

Introduction-self-supervised learning

- **Self-supervised learning (SSL)**: leverage the data's inherent co-occurrence relationships as the self-supervision. [1]
- **Relationship with unsupervised learning**: SSL is a **subset** of unsupervised learning and it utilizes a portion of the input as a supervisory signal.
- **Motivation**: 1. tremendous available unlabeled data 2. save time and resource 3. privacy concerns
- **Two stages**: 1. pretrain encoders in **pretext tasks** 2. apply frozen encoder to downstream tasks
- **Pretext tasks in CV**: image rotation prediction; masked patch reconstruction; relative position prediction
- **Pretext tasks in NLP**: next-word prediction

Self-supervised learning is a **feature extraction** scheme using unlabeled data.

Introduction-self-supervised learning

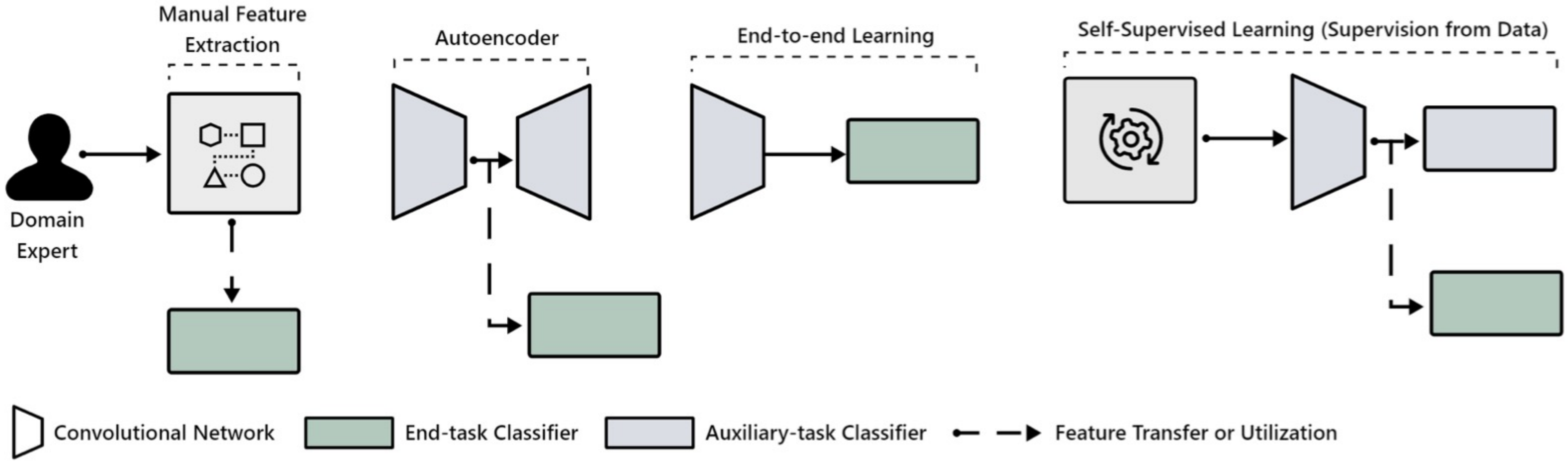

Fig. 1. Evolution of feature learning approaches from hand-crafted methods towards task discovery for self-supervision. [2]

2

Self-Supervised Learning in HAR

Self-Supervised Learning

- **Key Question**: how to get meaningful representation in the pretraining stage? How to design the **pretext tasks** (Open Question !)
- Some solutions: GAN, autoencoder, **contrastive learning**

- **Contrastive learning is one of the major and promising solutions!**
- **Goal:** to train the encoder making the similar (positive) pairs of data points closer and dissimilar (negative) ones orthogonal in the laten space
- **Commonly used Objective function** – InfoNCE:

$$\mathcal{L} = \mathbb{E}_{x,x^+,x^k}\left[-\log\left(\frac{e^{f(x)^T f(x^+)}}{e^{f(x)^T f(x^+)} + \sum_{k=1}^{K} e^{f(x)^T f(x^k)}}\right)\right]$$

Where x and x+ form a positive pair.
Let positive pair be closer and dissimilar pair far away.

Method 1: Multi-task SSL

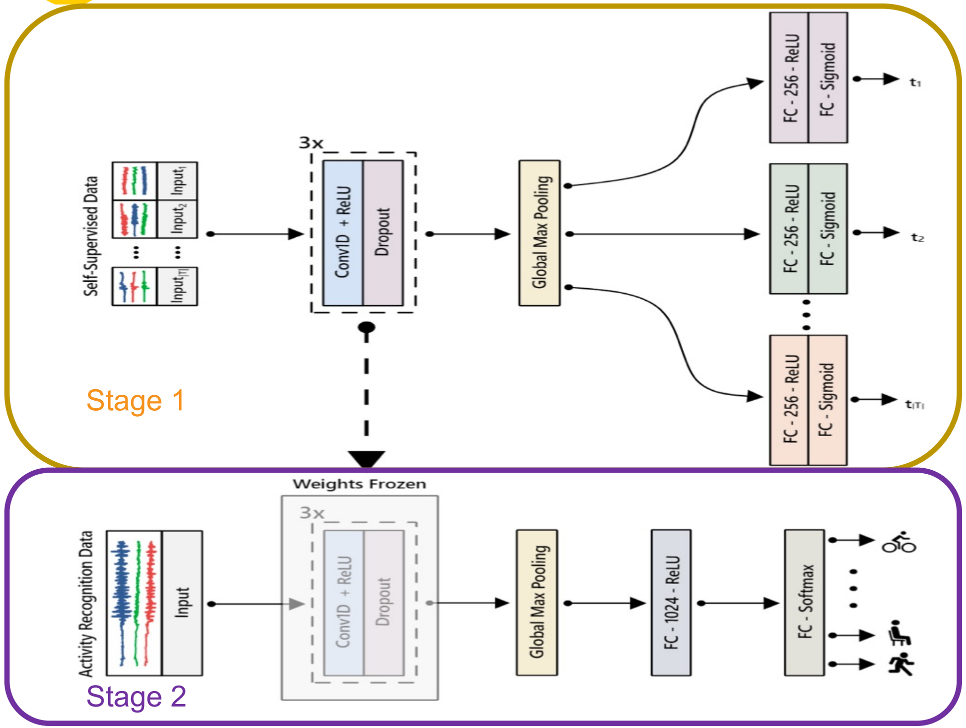

Fig. 2. The overall architecture of multi-task self-supervised learning. [2]

- In the **pretraining stage**, it is a **multi-task learning**. Each task is to detect whether the input is performed by a specific transformation.
- There are |T| detectors corresponding to |T| different transformations, like adding noise, scaling the magnitude, flipping the time-series signals.
- All tasks share the same encoder consisting of three convolution layers.
- In the **downstream task**, frozen encoder from the pretraining stage is fine-tuned to predict activities.

Method 2: Convolution Autoencoder (CAE)

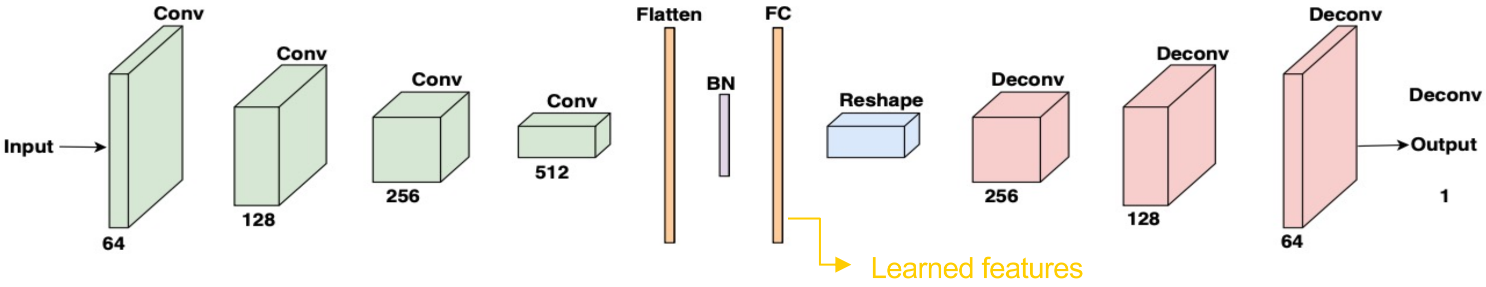

Fig. 3. Overview of the convolutional autoencoder . [3]

- **Input** consists of individual frames (1s) from six sensors and is considered as **a single channel image** by the autoencoder.
- The **encoder** contains four convolution blocks followed by a full-connected layer (FC).
- Vectors outputted by FC is **learned features** that will be used in the downstream task.

Method 3: Masked Reconstruction SSL

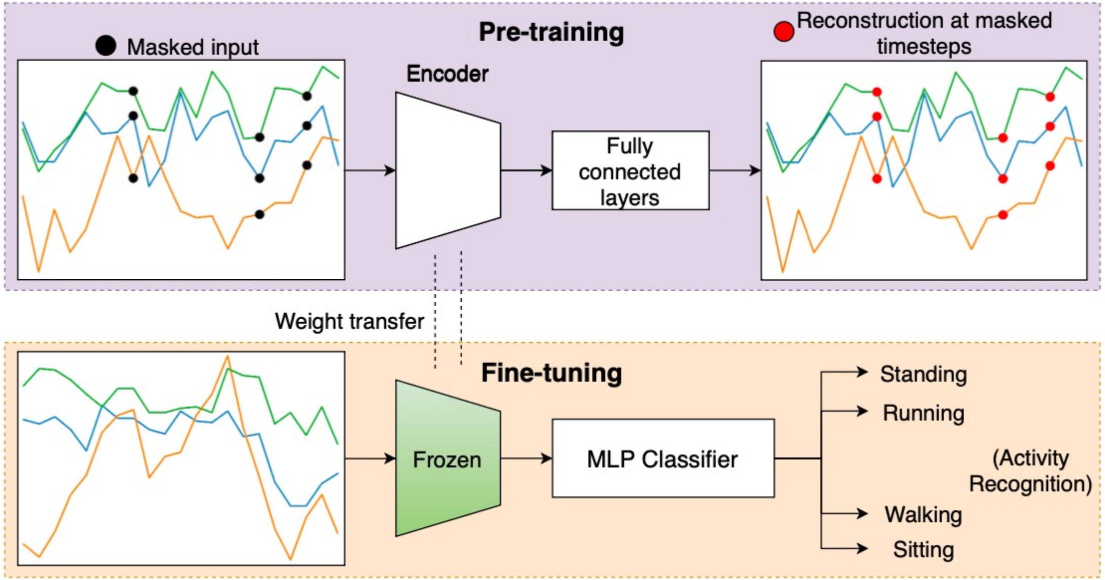

Fig. 4. The pipeline of masked reconstruction based self-supervision. [4]

- In the **pretraining stage**, an input is one frame that contains T consecutive N-dimensional sensor readings extracted by a sliding window.
- 10% of the samples in every frame are randomly masked for all dimensions.
- The task is to reconstruct the masked out parts and thus, to learn temporal patterns from context.
- Encoder here uses Transformer to obtain representations at each timestep, and than max-pooling layer generates the representation for the frame.

Method 4: Constrastive Predictive Coding (CPC)

- Input: raw time-series signals from accelerometer and gyroscope
- Positive pair: predicted vector w_1, w_2, w_3, w_4 and encoded vector of feature samples z_{t+1}, z_{t+2}, z_{t+3},

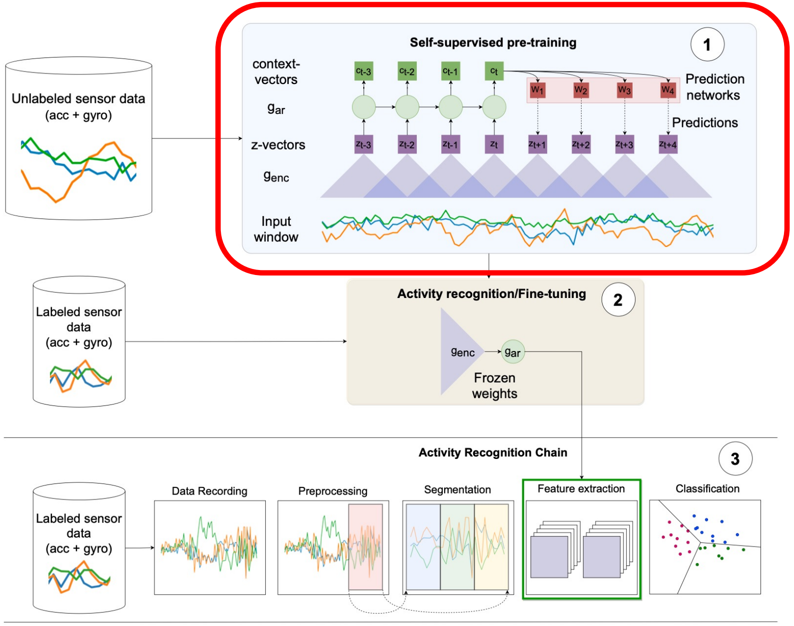

Fig. 5. Overview of CPC. [5]

Fig. 6. Architecture of pretraining model. (ct, w1), (ct, w2), (ct, w3), (ct, w4) are positive paires.

Method 5: Contrastive Self-supervised Learning (CSSHAR)

- Input: raw time-series signals from accelerometer and gyroscope
- Positive pair: apply two different augmentations to the same data

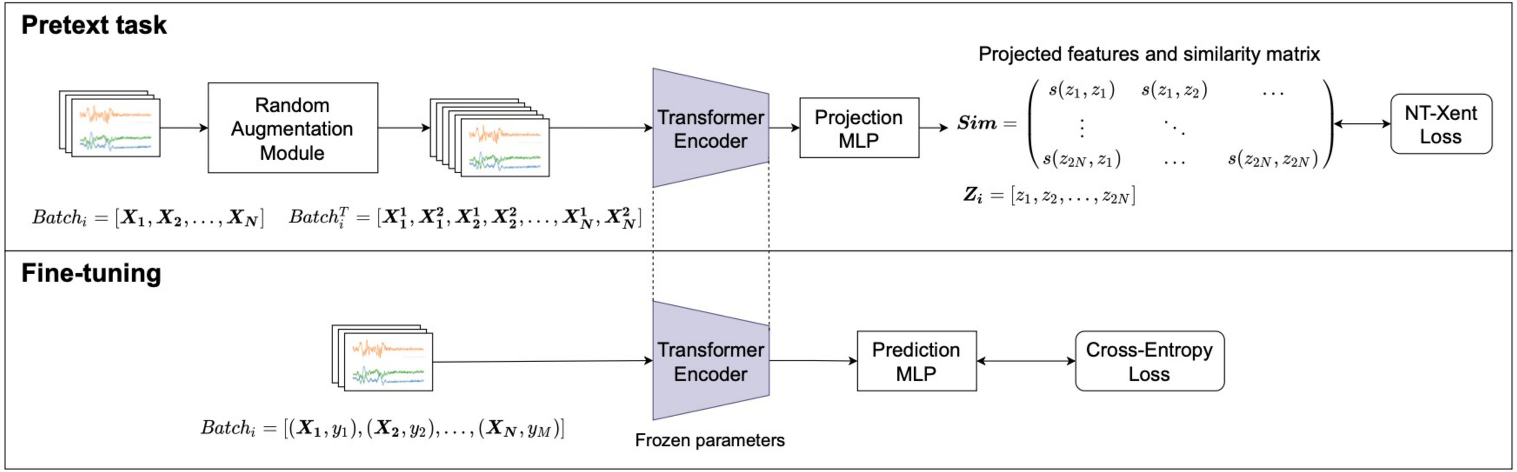

Fig. 7. Overview of CSSHAR. Applying two transformations to raw data X_1 produces a positive pair (X_1^1, X_1^2). [6]

3 Experiments

Dataset Collection

Dataset	# Users	# Activities	Sampling Rate (Hz)	Raw features	Train-Val-Test Split
UCI-HAR [7]	30	6 {standing, sitting, laying down, walking, downstairs and upstairs }	50	Acc_{x,y,z}, Gyro_{x,y,z}	Test: 20% of subjects in dataset Validation: 20% of the remaining subjects Trian: 80% of the remaining subjects
USC-HAD [8]	14	12 {walking–forward, left, right, upstairs, and downstairs–, running forward, jumping, sitting, standing, sleeping, and riding the elevator up and down }	100		
MobiAct [9]	61	11 {sitting, walking, jogging, jumping, stairs up, stairs down, stand to sit, sitting on a chair, sit to stand, car step-in, and car step-out }	200		

Data preprocessing: raw accelerometer and gyroscope signals are downsampled to 30Hz and segmented into 50% overlapping time-windows of 1 second length.

Experiment Settings

- **Performance Metric**: mean f1-score, given by

$$F_m = \frac{2}{|c|} \sum_{c} \frac{prec_c \times recall_c}{prec_c + recall_c}$$

where |c| corresponds to the number of classes while pre_c and $recall_c$ are the precision and recall for each class.

- Except aforementioned five SSL methods, two supervised methods are included in the comparison. **DeepConvLSTM** [] and **Transformer** (the encoder of CSSHAR). They are directly trained in the supervised-manner using all labelled data.
- Experimental procedure for SSL methods: 1. pretrain the encoder with unlabelled data, 2. freeze encoder and add a MLP as the trainable classifier, 3. compare the mean f1 score for the test dataset.

Results-Classification Performance

same architecture

Method	Type	Mean F1-Score MobiAct	UCI-HAR	USC-HAD
DeepConvLSTM	Sup.	82.4	82.83	44.83
Transformer	Sup.	83.92	95.26	60.56
Multi-task SSL	SSL	75.41	80.2	45.37
CAE	SSL	79.58	80.26	48.82
Masked Reconstruction	SSL	76.81	81.89	49.31
CPC	SSL	80.97	81.65	52.01
CSSHAR	SSL	**81.13**	**91.14**	**57.76**

Fig. 8. F1-scores for the baseline activity recognition task. Sup. denotes training the model in supervised manner; SSL denotes training the model in self-supervised manner.

Results-Classification Performance

- Given the same encoder and enough labelled data, the performance of supervised learning decides the upper bound of the performance of the self-supervised learning.
- CSSHAR is the SOA SSL method, which improves 9% and 5.75% from previous methods.
- Compared to supervised model DeepConvLSTM, CSSHAR also outperforms it in two datasets, which means SSL methods are capable of extracting robust feature embeddings without using data labels.

Results-Semi-supervised experiments

- Semi-supervised experiments:
  - 1. greatly reduce the size of training dataset: k ∈ {1, 2, 5, 10, 25, 50, 100} labeled examples per class are randomly sampled from the training set.
  - 2. use it to train the supervised and SSL models and compare their performance.
  - 3. repeat the procedure 10 times.

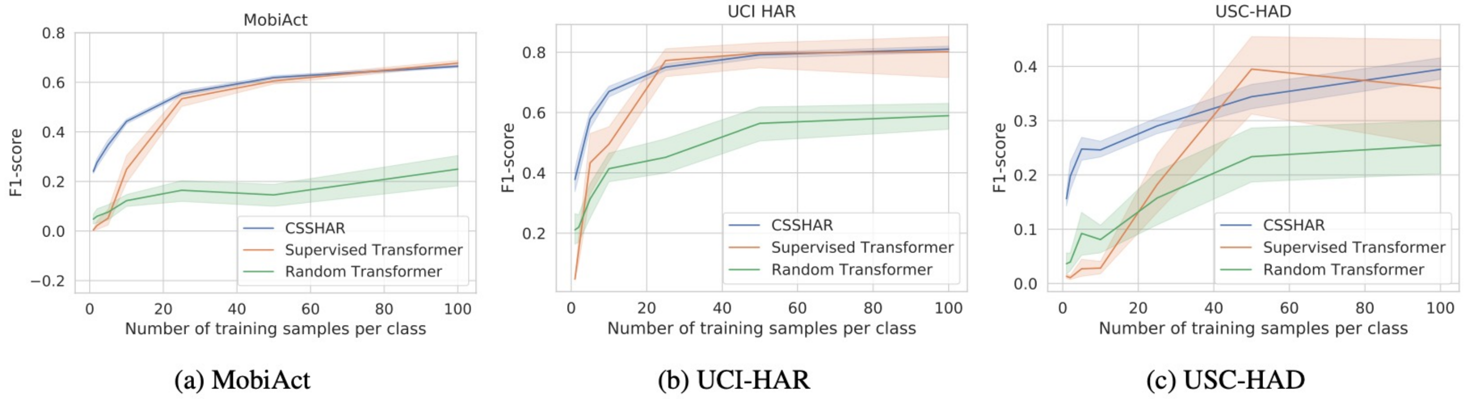

Fig. 9. Average F1-scores comparison among CSSHAR and the same architecture in supervised learning and in random frozen parameters.

Results- Semi-supervised experiments

- Both supervised transformer and the SSL transformer (CSSHAR) show about the same performance when $k > 25$, but CSSHAR is more robust.
- Unlike CSSHAR, the supervised transformer model completely fails when only very limited data ($k < 10$) is available.

4 Conclusions & Discussions

Conclusions & Discussions

- Applying SSL in Federated Learning manner can reduce the communication overhead and allow **personalized classifiers** for each user.
- The performance of supervised learning decides the **upper bound** of the performance of the self-supervised learning.
- Self-supervised learning is significantly helpful for the training in the **small-size datasets**.
- What is the best way to preprocess the sensors' signals remains to be further exploited, like the optimal length of time windows.
- The goodness of self-supervised learning **cannot** be directly compared but evaluated by the performance of downstream tasks, which means it is hard to compare the validity and rationales behind those methods.

Reference

[1]. Liu, Xiao, et al. "Self-supervised learning: Generative or contrastive." *IEEE Transactions on Knowledge and Data Engineering* (2021).

[2]. A. Saeed, T. Ozcelebi, and J. Lukkien. Multi-task Self- Supervised Learning for Human Activity Detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):1–30, 2019.

[3]. H. Haresamudram, D. V. Anderson, and T. Plo ̈tz. On the role of features in human activity recognition. In *Proceedings of the 23rd International Symposium on Wearable Computers*, pages 78–88, 2019.

[4]. H. Haresamudram, A. Beedu, V. Agrawal, P. L. Grady, I. Essa, J. Hoffman, and T. Plo ̈tz. Masked reconstruction based self-supervision for human activity recognition. *Proceedings - International Symposium on Wearable Computers, ISWC*, pages 45–49, 2020.

[5]. Haresamudram, Harish, Irfan Essa, and Thomas Plötz. "Contrastive predictive coding for human activity recognition." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.2 (2021): 1-26.

[6]. Khaertdinov, Bulat, Esam Ghaleb, and Stylianos Asteriadis. " Contrastive Self-supervised Learning for Sensor-based Human Activity Recognition." *2021 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2021.

Reference

[7]. D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. Reyes. A public domain dataset for human activity recognition using smartphones. In *Proceedings of the 21th International European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, page 437–442, 2013.

[8]. M. Zhang and A. A. Sawchuk. Usc-had: A daily activity dataset for ubiquitous activity recognition using wear- able sensors. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp ’12, page 1036–1043, New York, NY, USA, 2012. Association for Computing Machinery.

[9]. G. Vavoulas., C. Chatzaki., T. Malliotakis., M. Pediaditis., and M. Tsiknakis. The mobiact dataset: Recognition of activities of daily living using smartphones. In *Proceedings of the International Conference on Information and Communication Technologies for Ageing Well and e-Health - Volume 1:* ICT4AWE, (ICT4AGEINGWELL 2016), pages 143–151. INSTICC, SciTePress, 2016.

Thanks

Appendix: Dataset Preprocessing

Table. Data preprocessing in adopted by each method

<table>
<tr><th>Method</th><th>Sampling Rate after downsampling (Hz)</th><th>Length of a Frame</th><th>Overlapping rate (%)</th><th>Time-Series Signals only</th></tr>
<tr><td>Multi-task SSL</td><td>--</td><td>400 timestamps</td><td rowspan="5">50</td><td rowspan="5">True</td></tr>
<tr><td>CAE</td><td>33</td><td rowspan="4">1 s</td></tr>
<tr><td>Masked Reconstruction</td><td>33</td></tr>
<tr><td>CPC</td><td>50</td></tr>
<tr><td>CSSHAR</td><td>30</td></tr>
</table>

Appendix: Constructing Positive Pairs

- Two Major Ways to Construct Positive Pairs:

Identity Recognition: let encoder learn the similarity between the original data and corresponding transformed one

Co-Occurence Relationship Recognition: let encoder learn the similarity between the target data and related data